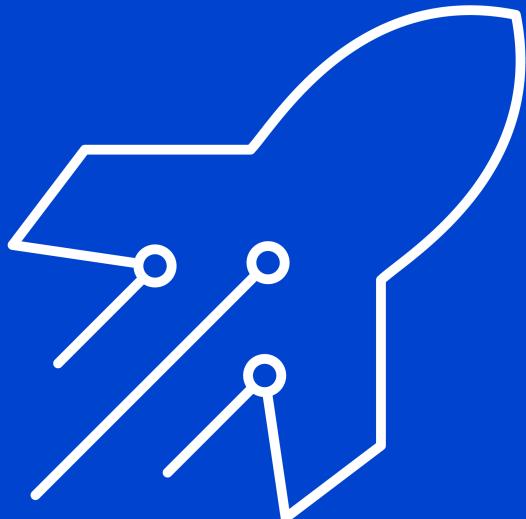


# Lab Guide

## CP4Data and Z Integration HOL

Session: EZS66T316BH

Maggie Lin [meichi@us.ibm.com](mailto:meichi@us.ibm.com), Watson Machine Learning for z/OS Development





## Table of Contents

Legal Disclaimer .....	4
Introduction.....	5
Lab Overview .....	5
Lab Instructions .....	6
CP4D integration with WMLz.....	6
Log in to CP4D.....	6
1. Model Training in CP4D .....	9
Log in to IBM Watson Machine Learning for z/OS.....	30
2. Model Management and Deployment in WMLz .....	34
Section 1. Dashboard.....	34
Section 2. Model Management .....	34
Section 3. Deployment Model to Scoring Server .....	39
We Value Your Feedback!.....	46

## Legal Disclaimer

© IBM Corporation 2020. All Rights Reserved. The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

## Introduction

IBM Watson Machine Learning for z/OS (WMLz) is an enterprise machine learning platform that provides end to end management of the entire machine learning workflow. WMLz helps to simplify and significantly reduce the time to train and deploy machine learning models by:

- Integrating all the tools and functions needed for machine learning and automating the machine learning workflow.
- Providing a platform for better collaboration across different personas including Data Scientist, Data Engineer, Business Analyst and Application Developers, for a successful machine learning project.
- Infusing cognitive capabilities into the machine learning workflow to help determine when model results deteriorate and need to be tuned and provide suggestions for updates or changes.

## Lab Overview

For this hands-on lab, you have the opportunity to use CP4Data and Watson Machine Learning for z/OS (WMLz) to experience the full life cycle of predictive model development and management

- Train a model and save it in CP4D environment
- Import the CP4D trained model to WMLz, and deploy the model using the Model Management Dashboard in WMLz
- Test scoring of the deployed model in WMLz

## [Lab Instructions](#)

Each student should receive a lab worksheet. The worksheet provides information that is specific to each student to use throughout the lab.

Some of the instructions in the lab instruct the student to use a value that is marked with angle brackets (< >). The value the student should use is on their lab worksheet. Make sure to exclude the brackets when using the value for its intended purpose in the lab.

## [CP4D integration with WMLz](#)

This customer churn model is a simplified customer churn model to determine the likelihood of the customer leaves the phone carrier. Every user has their own profile and information like phone usage, payment method, and annual income. The value of adding Machine Learning to the customer churn is that ML generates a smart score for the customer with the information provided. The phone carrier can define a threshold for this score to help determine if there is additional action they can take to retain the customer. The customer churn is a critical metric because it is much less expensive to retain existing customers than it is to acquire new customers.

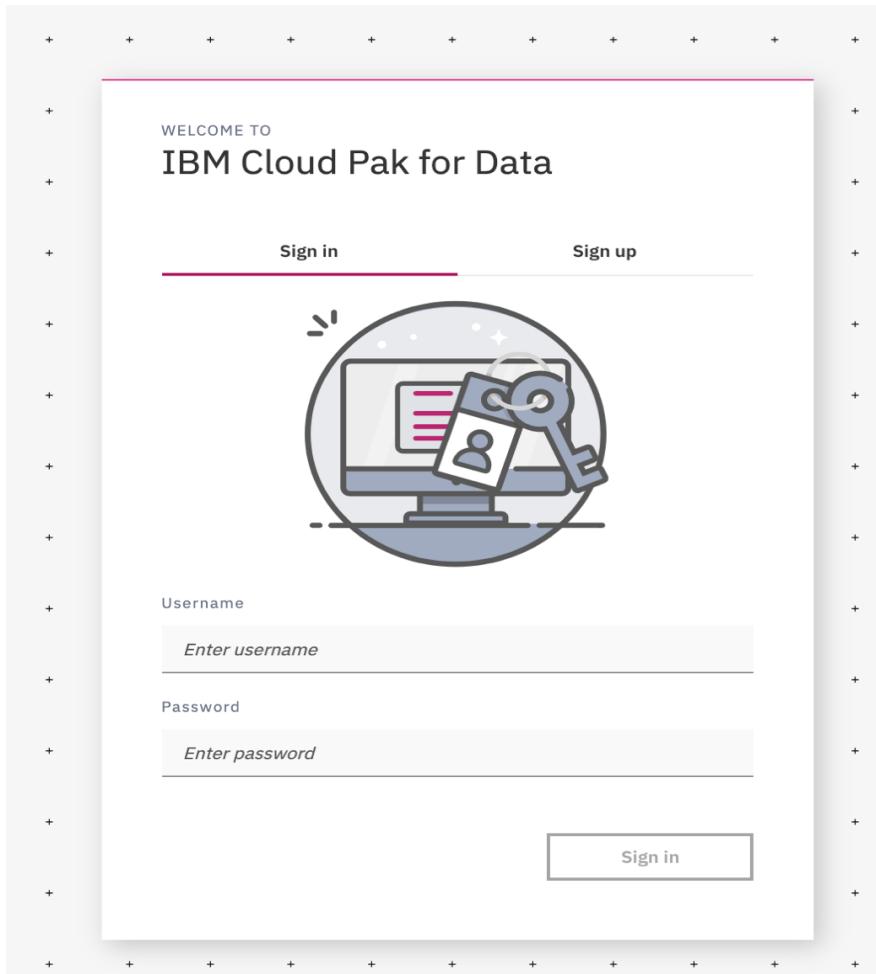
In this session, you will go through the process of building the customer churn model with CP4D, experiencing the flow to analyze data, develop model, and then import the model and deploy model with WMLz.

- Model Training with Jupyter notebook
- Model Management and Deployment

## [Log in to CP4D](#)

The URL and authentication credentials for the CP4D Web UI are provided on a separate worksheet. Contact the lab instructor if you did not receive a worksheet.

- 1. Start the Chrome or Firefox browser, and enter the URL assigned to you, e.g. <cp4d\_login\_url>.
- 2. Enter the <cp4d\_login\_userid> and <cp4d\_login\_password> credentials from your lab worksheet.



You should be presented with the CP4D Lets get started main page

IBM Cloud Pak for Data

All

Search

WELCOME, ctp!

# Let's get started!

Use these resources to make the most of your IBM Cloud Pak for Data experience.

Collect and organize

IBM Cloud Pak for Data:  
Collect and organize

0:00 / 17:44

Build your enterprise data catalog and ensure that your policies and rules.

- Explore business terms
- Explore policies
- Discover assets
- Explore catalogs

DATA JOURNEY

## From data to insight

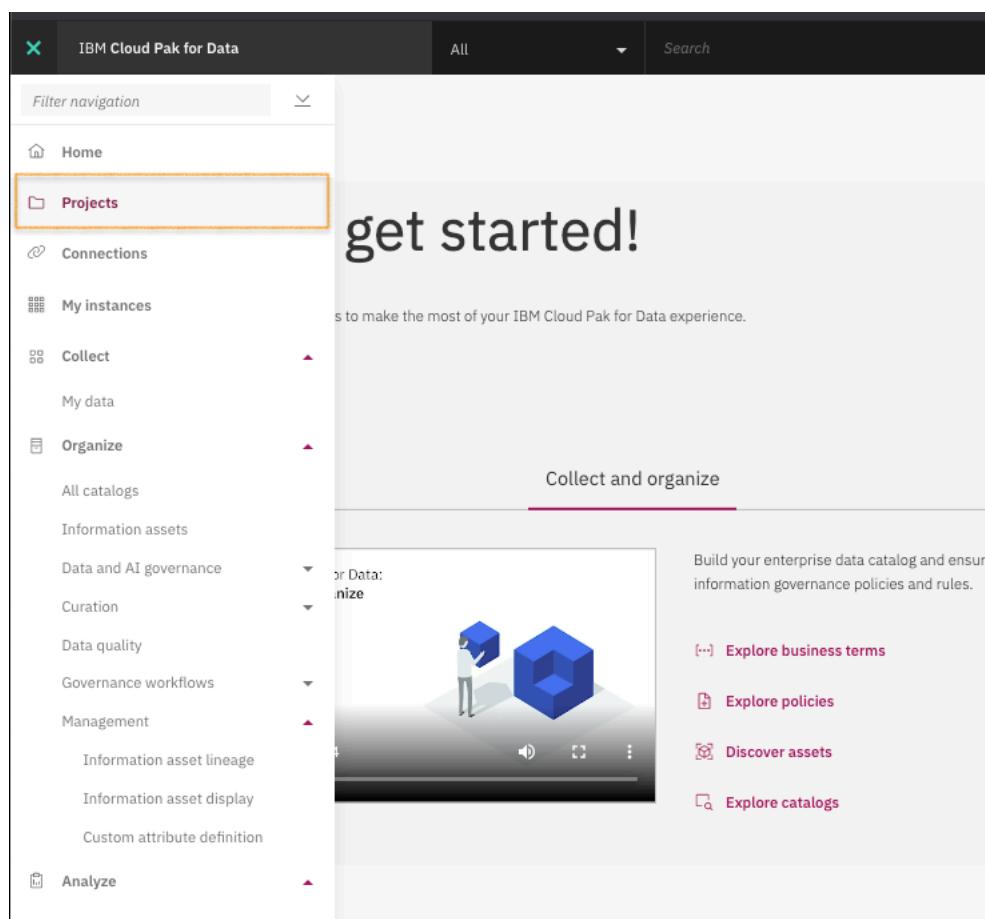
Understand the end-to-end data journey to see how your role fits into the story.

# 1. Model Training in CP4D

In this Lab, we will use the Web UI of CP4D to leverage machine learning to understand customer churn and determine which customer is likely to leave. You will learn how to use Jupyter notebook of CP4D to train a binary classification scikit learn (RandomForestClassifier) model.

1. Select a project to contain your notebook.

Click on the 3 short horizontal bars in the top left of the web page and choose “**Projects**”



Click “**New Project**” button to create one. Select to create an empty project, and name it “**<cp4d\_login\_userid>-IMLz\_HOL**

IBM Cloud Pak for Data

All

Search

Projects

Search

New project

Name	Project Type	User Role	Last Updated	Actions
sudesh-uob	Analytics	Admin	30 Dec 2019, 10:20 PM	
vjprj	Analytics	Admin	11 Dec 2019, 10:14 AM	
test2	Analytics	Admin	28 Nov 2019, 2:41 PM	
garyde	Analytics	Admin	14 Jan 2020, 12:56 PM	
Teleco_Churn_CH	Analytics	Admin	4 Dec 2019, 7:15 PM	

IBM Cloud Pak for Data

All

Search

← Back

## Create a project

Choose whether to create an empty project or to preload your project with data and analytical assets. Add collaborators and data, and then choose the right tools to accomplish your goals. Add services as necessary.



**Create an empty project**

Add the data you want to prepare, analyze, or model. Choose tools based on how you want to work: write code, create a flow on a graphical canvas, or automatically build models.

**USE TO**

Prepare and visualize  
Analyze data in notes  
Train models

IBM Cloud Pak for Data    All    Search

## New project

### Define project details

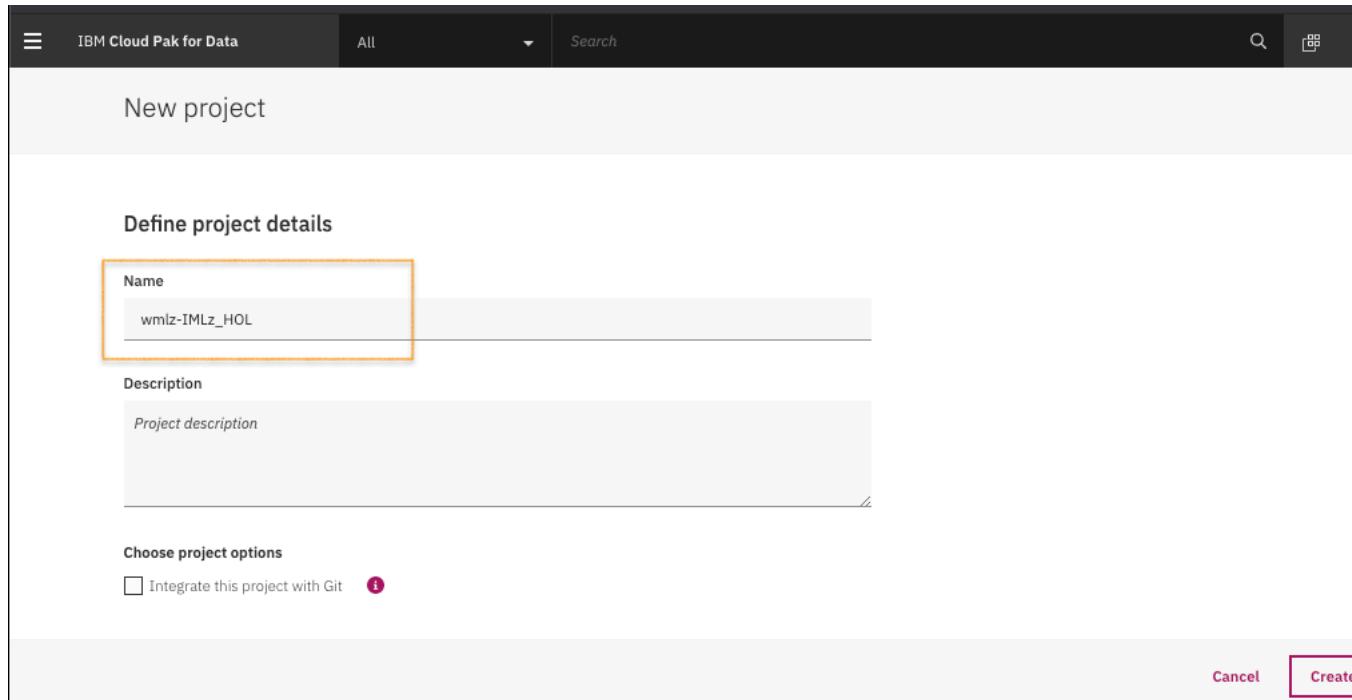
Name  
wmlz-IMLz\_HOL

Description  
*Project description*

Choose project options

Integrate this project with Git i

Cancel Create



You should be now at the project overview page.

The screenshot shows the project overview for 'wmlz-IMLz\_HOL'. At the top, there's a navigation bar with 'IBM Cloud Pak for Data', a search bar, and various project management icons. Below the header, the project name 'wmlz-IMLz\_HOL' is displayed, along with a 'Last Updated: 16 Jan, 2020' timestamp. A prominent '0' indicates zero assets. The main content area is divided into two columns: 'Recent activity' (which is currently empty) and a detailed section for 'Date created' (16 Jan, 2020), 'Description' (No description available), 'Storage' (File System, 0 Byte used), 'Collaborators' (ctp, Admin), and 'Associated deployment space' (No deployment space associated). A note says 'Alerts related to this project will show here when the project is active.'

- 2. From the project overview, click on “Asset” to load 2 asset files **customers.csv** and **MyLabelEncoder-1.2.0.401.post201912301918.tar.gz**  
Download the 2 files from the HOL GitHub link if you have not :  
[https://github.com/meichilin/IBMWMLz/tree/master/FastStart\\_2020](https://github.com/meichilin/IBMWMLz/tree/master/FastStart_2020)

IBM Cloud Pak for Data

All

Search

My Projects > wmlz-IMLz\_HOL

Overview Assets Environments Jobs Access Control Settings

What assets are you looking for?

**▼ Data assets**

NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
You don't have any Data assets yet.				

IBM Cloud Pak for Data

All

Search

My Projects > forMaggie

Overview Assets Environments Jobs Access Control Settings

What assets are you looking for?

**▼ Data assets**

0 asset selected.

<input type="checkbox"/>	NAME	TYPE	CREATED BY	LAST MODIFIED
<input type="checkbox"/>	customers.csv	Data Asset	ctp	10 Jan 2020, 12:56:15
<input type="checkbox"/>	MyLabelEncoder-1.2.0.401.post201912301918.tar.gz	Data Asset	ctp	10 Jan 2020, 12:56:01

3. Click “Add to project” to add “Notebook” asset to the project.

IBM Cloud Pak for Data

All

Search

My Projects > wmlz-IMLz\_HOL

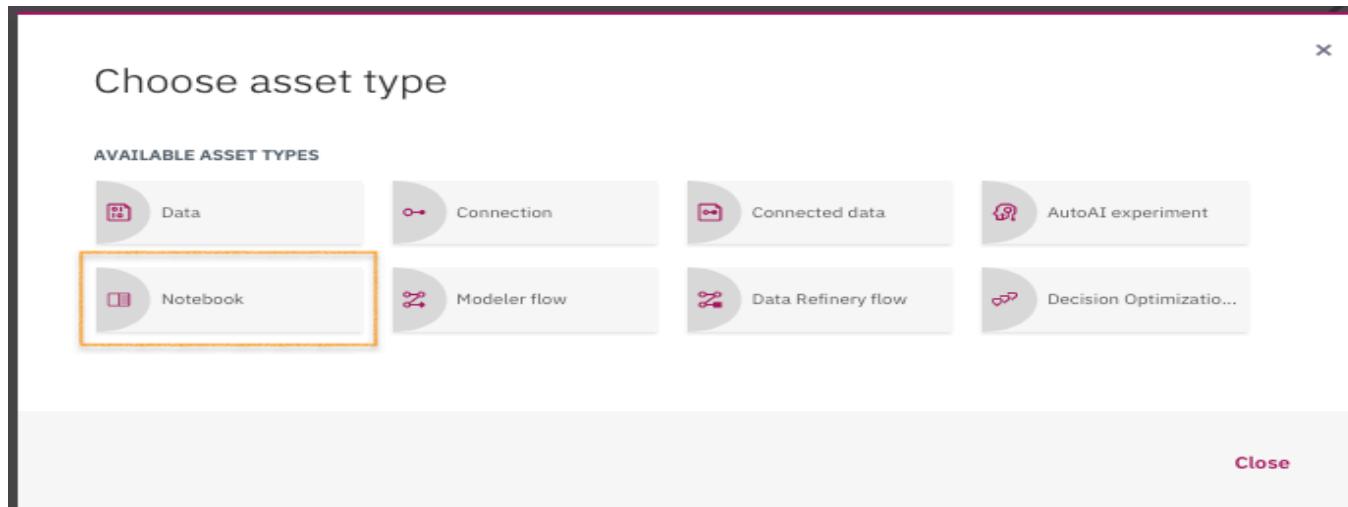
Overview Assets Environments Jobs Access Control Settings

wmlz-IMLz\_HOL

Last Updated: 16 Jan, 2020

2 Assets

**Add to project**



- 4. Create a new notebook “From file”. Import **ChurnScikit\_CP4D\_WMLz.ipynb** from the HOL GitHub link if you have not : [https://github.com/meichilin/IBMWMLz/tree/master/FastStart\\_2020](https://github.com/meichilin/IBMWMLz/tree/master/FastStart_2020). Leave runtime to Default Python 3.6 (1 vCPU and 2 GB RAM). Then click “Create Notebook” button.

IBM Cloud Pak for Data

All

Search

My Projects > wmlz-IMLz\_HOL > Add Notebook

### New notebook

Blank   **From file**   From URL

Name  
Type Notebook Name here  
40 characters remaining

Description (optional)  
Type your Description here  
500 characters remaining

Select runtime  
Default Python 3.6 (1 vCPU and 2 GB RAM)

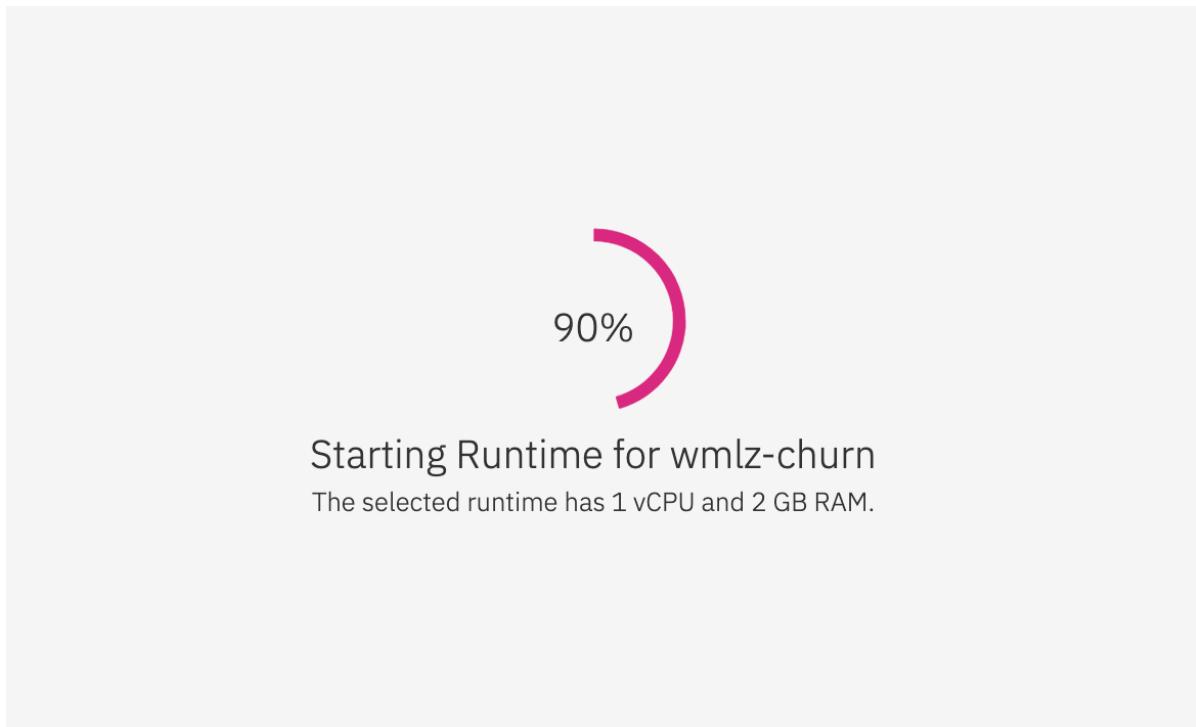
Notebook file  
**Choose file**

Import a notebook file (.ipynb) from your local device.

Cancel   **Create Notebook**

You will see the following spinner indicating a runtime container is being created for the Jupyter notebook.

This could take a minute or two for the first time you open a notebook container in your project.



A message indicates the Jupyter notebook is being launched. Shortly afterward, you will see the ChurnScikit\_CP4D\_WMLz.ipynb loaded.

The screenshot shows a Jupyter notebook interface within the IBM Cloud Pak for Data environment. The top navigation bar includes 'IBM Cloud Pak for Data', 'All' (dropdown), and a 'Search' bar. Below the navigation is a breadcrumb trail: 'My Projects > forMaggie > ChurnScikit\_CP4D\_WMLz'. The toolbar below the breadcrumb includes File, Edit, View, Insert, Cell, Kernel, Widgets, Help, and various cell type icons (Code, Markdown, etc.). A 'Run' button is also present. The main area displays the following code and its execution results:

```
In [1]: # In this Notebook we shall create a Machine Learning Model using Scikit Learn 0.19
There are several different approaches and frameworks for predicting the likelihood of a customer to churn. In this notebook, we purpose.

In [110]: import sklearn
print(sklearn.__version__)
0.19.1

In [3]: !pip install --user scikit-learn==0.19.1
Collecting scikit-learn==0.19.1
  Downloading https://files.pythonhosted.org/packages/3d/2d/9fbc7baa5f44bc9e88ffb7ed32721b879b
earn-0.19.1-cp36-cp36m-manylinux1_x86_64.whl (12.4MB)
| ################################## | 12.4MB 10.7MB/s eta 0:00:01
Installing collected packages: scikit-learn
Successfully installed scikit-learn-0.19.1

In [4]: !pip install /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz
Processing /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz
Building wheels for collected packages: MyLabelEncoder
  Building wheel for MyLabelEncoder (setup.py) ... done
    Created wheel for MyLabelEncoder: filename=MyLabelEncoder-1.2.0.401.post202001102058-cp36-m
3acb3efb7417c46fc8531c544b18e381cb940cb66f0de2d243dce2a6
    Stored in directory: /home/wsuser/.cache/pip/wheels/2b/69/0b/0d1a542a512bea2988b3dcba44df82c
Successfully built MyLabelEncoder
Installing collected packages: MyLabelEncoder
Successfully installed MyLabelEncoder-1.2.0.401.post202001102058

In [111]: import os
import numpy as np
```

Next, we read in a dataset that we will use to develop a Machine Learning model.

We can read the data here in various ways. We are showing here how to read data from a CSV file.

## 5. Now to train the model, execute each notebook cell step by step.

First, make sure we are using python scikit learn package 0.19.1 version, and install the customized transformer **MyLabelEncoder**.

Execute the first cell to know the current scikit learn package version. If it is already at 0.19.1 version, you can proceed on to the next step to install the customized transformer **MyLabelEncoder**.

The screenshot shows the IBM Cloud Pak for Data Jupyter Notebook interface. The top navigation bar includes 'IBM Cloud Pak for Data', a search bar, and a search icon. Below the navigation is a breadcrumb trail: 'My Projects > wmlz-IMLz\_HOL > ChurnScikit\_CP4D\_WMLz'. On the right side of the header are icons for sharing, running, and other actions. The main menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. A status bar on the right says 'Not Trusted'. The toolbar below the menu includes icons for new notebook, run cell, format, and nbdiff. The main content area features a large bold title: 'In this Notebook we shall create a Machine Learning Model using Scikit Learn 0.19'. Below the title is a text block: 'There are several different approaches and frameworks for predicting the likelihood of a customer to churn. In this notebook, we illustrate how scikit learn can be used for this purpose.' A code cell is shown with the input: 'In [4]: import sklearn print(sklearn.\_\_version\_\_)' and the output: '0.20.3'. This code cell is highlighted with an orange border.

Pip install the scikit learn package to 0.19.1 version.

Pip install the customized transformer **MyLabelEncoder**.

In [4]:

```
import sklearn
print(sklearn.__version__)
0.20.3
```

In [3]:

```
!pip install --user scikit-learn==0.19.1
Collecting scikit-learn==0.19.1
  Downloading https://files.pythonhosted.org/packages/3d/2d/9fbc7baa5f44bc9e88ffb7ed32721b879bfa416573e85031e16f52569bc9/scikit_learn-0.19.1-py3-none-any.whl (12.4MB)
    #####
    12.4MB 10.7MB/s eta 0:00:01
Installing collected packages: scikit-learn
Successfully installed scikit-learn-0.19.1
```

In [4]:

```
!pip install /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz
Processing /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz
Building wheels for collected packages: MyLabelEncoder
  Building wheel for MyLabelEncoder (setup.py) ... done
    Created wheel for MyLabelEncoder: filename=MyLabelEncoder-1.2.0.401.post202001102058-cp36-none-any.whl size=1905 sha256=d6c65943acb8531c544b18e381cb940cb66f0de2d243dc2a6
    Stored in directory: /home/wsuser/.cache/pip/wheels/2b/69/0b/0dla542a512bea2988b3dcba44df82c09221a2c26ab00cba7c
Successfully built MyLabelEncoder
Installing collected packages: MyLabelEncoder
Successfully installed MyLabelEncoder-1.2.0.401.post202001102058
```

6. If your scikit learn package version is already in 0.19.1 version, skip this step.

Otherwise, exit the notebook to restart the kernel to let the new the scikit learn package version to take effect in the notebook session.

Exit out of the notebook by clicking on your project name to leave the page.

In [5]:

```
import sklearn
print(sklearn.__version__)
0.20.3
```

In **Assets** page, **Notebooks** section, locate the ChurnScikit\_CP4D\_WMLz notebook.

The screenshot shows the 'Assets' tab selected in the navigation bar. In the 'Notebooks' section, the row for 'ChurnScikit\_CP4D\_WMLz' is highlighted with an orange border. The 'Action' column for this row contains a stop icon, which is the target for stopping the kernel.

NAME	SHARED	SCHEDULED	STATUS	LANGUAGE	LAST EDITOR	LAST MODIFIED
ChurnScikit_CP4D_WMLz	○			Python 3.6	ctp	16 Jan 2020

Under **Action**, click on it to **Stop Kernel**.

The screenshot shows the IBM Cloud Pak for Data interface. At the top, there's a navigation bar with 'IBM Cloud Pak for Data' and a search bar. Below the navigation bar, the 'Assets' tab is selected. A search bar at the top of the main content area contains the placeholder 'What assets are you looking for?'. Under the 'Data assets' section, there are two entries:

	NAME	TYPE	CREATED BY	LAST MODIFIED
<input type="checkbox"/>	CSV customers.csv	Data Asset	ctp	16 Jan 2020, 11:14:24 am
<input type="checkbox"/>	gz MyLabelEncoder-1.2.0.401.post201912301918.tar.gz	Data Asset	ctp	16 Jan 2020, 11:14:02 am

Below the data assets, the 'Notebooks' section is expanded, showing one notebook entry:

NAME	SHARED	SCHEDULED	STATUS	LANGUAGE	LAST EDITOR	LAST MODIFIED
ChurnScikit_CP4D_WMLz	<input checked="" type="checkbox"/>		<span>○</span>	Python 3.6	ctp	16 Jan 2020

A context menu is open on the right side of the 'ChurnScikit\_CP4D\_WMLz' notebook entry, with the 'Delete' option highlighted.

Once the kernel is stopped, go back into the **ChurnScikit\_CP4D\_WMLz** notebook to continue.

What assets are you looking for?

**Data assets**

0 asset selected.

<input type="checkbox"/>	NAME	TYPE	CREATED BY	LAST MODIFIED
<input type="checkbox"/>	CSV customers.csv	Data Asset	ctp	16 Jan 2020, 11:14:24 am
<input type="checkbox"/>	MyLabelEncoder-1.2.0.401.post201912301918.tar.gz	Data Asset	ctp	16 Jan 2020, 11:14:02 am

**Notebooks**

NAME	SHARED	SCHEDULED	STATUS	LANGUAGE	LAST EDITOR	LAST MODIFIED
ChurnScikit_CP4D_WMLz				Python 3.6	ctp	16 Jan 2020

Enable the notebook in edit mode for continuous notebook execution.

In [5]:

```
import sklearn
print(sklearn.__version__)
0.20.3
```

Lets make sure the python scikit learn package shows 0.19.1 version.

The screenshot shows a Jupyter Notebook interface within the IBM Cloud Pak for Data environment. The top navigation bar includes 'IBM Cloud Pak for Data', 'All', 'Search', and various project and notebook management icons. The main toolbar has options like File, Edit, View, Insert, Cell, Kernel, Widgets, Help, and a toolbar with icons for Run, Format, and Code. A status bar at the bottom right indicates 'Not Trusted'. The central area displays a title 'In this Notebook we shall create a Machine Learning Model using Scikit Learn 0.19' and a descriptive text about customer churn prediction. A code cell is highlighted with an orange border, containing the following Python code:

```
In [1]: import sklearn  
print(sklearn.__version__)  
0.19.1
```

— 7. Lets continue on to execute the notebook cell **one by one**.

Import numpy package.

Load the training data **customer.csv** and split training data.

IBM Cloud Pak for Data    All    Search

My Projects > wmlz-IMLz\_HOL > ChurnScikit\_CP4D\_WMLZ

File Edit View Insert Cell Kernel Widgets Help    Not Trusted

Format Code nbdiff

```
Collecting scikit-learn==0.19.1
  Downloading https://files.pythonhosted.org/packages/3d/2d/9fbc7baa5f44bc9e88ffb7ed32721b879bfa416573e85031e16f52569bc9/scikit_learn-0.19.1-py36-manylinux1_x86_64.whl (12.4MB)
    #####
    |████████████████████████████████| 12.4MB 5.2MB/s eta 0:00:01
Installing collected packages: scikit-learn
Successfully installed scikit-learn-0.19.1
```

In [4]: `!pip install /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz`

```
Processing /project_data/data_asset/MyLabelEncoder-1.2.0.401.post201912301918.tar.gz
Building wheels for collected packages: MyLabelEncoder
  Building wheel for MyLabelEncoder (setup.py) ... done
    Created wheel for MyLabelEncoder: filename=MyLabelEncoder-1.2.0.401.post202001161959-cp36-none-any.whl size=1906 sha256=724c8b512550eb350b1b0b0b95db52f20438668e11c95e2b24
    Stored in directory: /home/wsuser/.cache/pip/wheels/2b/69/0b/01da542a512bea2988b3dcba44df82c09221a2c26ab00cba7c
Successfully built MyLabelEncoder
Installing collected packages: MyLabelEncoder
Successfully installed MyLabelEncoder-1.2.0.401.post202001161959
```

In [2]: `import os  
import numpy as np`

Next, we read in a dataset that we will use to develop a Machine Learning model.

We can read the data here in various ways. We are showing here how to read data from a CSV file.

In [3]: `import pandas as pd  
df_data_1 = pd.read_csv('/project_data/data_asset/customers.csv')  
df_data_1.head()`

Out[3]:

ID	LONGDISTANCE	INTERNATIONAL	LOCAL	DROPPED	PAYMETHOD	LOCALBILLTYPE	LONGDISTANCEBILLTYPE	USAGE	RATEPLAN	CHURN	GENDER	STATUS	CHILDREN
0	1	23	0	206	0	CC	Budget	Intnl_discount	229	3	T	F	S
1	6	29	0	45	0	CH	FreeLocal	Standard	75	2	F	M	M
2	8	24	0	22	0	CC	FreeLocal	Standard	47	3	F	M	M
3	11	26	0	32	1	CC	Budget	Standard	59	1	F	M	S
4	17	12	0	46	4	CC	FreeLocal	Standard	58	1	F	M	M

Continue on the feature engineering work for model development.

IBM Cloud Pak for Data All Search

My Projects > forMaggie > ChurnScikit\_CP4D\_WMLz

File Edit View Insert Cell Kernel Widgets Help

In [111]: `import os  
import numpy as np`

Next, we read in a dataset that we will use to develop a Machine Learning model.

We can read the data here in various ways. We are showing here how to read data from a CSV file.

In [112]: `import pandas as pd  
df_data_1 = pd.read_csv('/project_data/data_asset/customers.csv')  
df_data_1.head()`

Out[112]:

ID	LONGDISTANCE	INTERNATIONAL	LOCAL	DROPPED	PAYMETHOD	LOCALBILLTYPE	LONGDISTANCEBILLTYPE
0	1	23	0	206	0	CC	Budget
1	6	29	0	45	0	CH	FreeLocal
2	8	24	0	22	0	CC	FreeLocal
3	11	26	0	32	1	CC	Budget
4	17	12	0	46	4	CC	FreeLocal

In [113]: `cmergedDF = df_data_1.copy()`

In [114]: `# drop the ID field  
cmergedDF.drop(['ID'], axis=1, inplace=True)`

In [115]: `cmergedDF.head()`

Out[115]:

	LONGDISTANCE	INTERNATIONAL	LOCAL	DROPPED	PAYMETHOD	LOCALBILLTYPE	LONGDISTANCEBILLTYPE
0	23	0	206	0	CC	Budget	Intl_discount
1	29	0	45	0	CH	FreeLocal	Standard
2	24	0	22	0	CC	FreeLocal	Standard
3	26	0	32	1	CC	Budget	Standard
4	12	0	46	4	CC	FreeLocal	Standard

Split the data for training and testing.

The screenshot shows a Jupyter Notebook interface within the IBM Cloud Pak for Data environment. The top navigation bar includes 'IBM Cloud Pak for Data', 'All', and a search bar. Below the menu bar, the current project path is 'My Projects > forMaggie > ChurnScikit\_CP4D\_WMLz'. The toolbar contains standard options like File, Edit, View, Insert, Cell, Kernel, Widgets, Help, and various execution and notebook management icons.

```

In [116]: cmergedDf.columns.tolist()
Out[116]: ['LONGDISTANCE',
 'INTERNATIONAL',
 'LOCAL',
 'DROPPED',
 'PAYMETHOD',
 'LOCALBILLTYPE',
 'LONGDISTANCEBILLTYPE',
 'USAGE',
 'RATEPLAN',
 'CHURN',
 'GENDER',
 'STATUS',
 'CHILDREN',
 'ESTINCOME',
 'CAROWNER',
 'AGE']

In [117]: from sklearn.model_selection import train_test_split
y = cmergedDf['CHURN']
x = cmergedDf.drop('CHURN', axis=1)
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)

In [118]: from sklearn.pipeline import Pipeline
#from sklearn.preprocessing import StandardScaler, OneHotEncoder, LabelEncoder
from sklearn.preprocessing import LabelEncoder

#categorical_transformer = Pipeline(steps=[('onehot', OneHotEncoder(handle_unknown='ignore'))])
categorical_features = cmergedDf.select_dtypes(include=['object']).drop(['CHURN'], axis=1)

In [119]: cat_indices = [cmergedDf.columns.get_loc(c) for c in categorical_features]

In [120]: df1 = cmergedDf.copy()

In [121]: df1.head()
Out[121]: LONGDISTANCE INTERNATIONAL LOCAL DROPPED PAYMETHOD LOCALBILLTYPE LONGDISTANCEBILLTYPE

```

8. Use RandomForestClassifier as the algorithm for the pipeline training.

IBM Cloud Pak for Data All Search

My Projects > forMaggie > ChurnScikit\_CP4D\_WMLZ

File Edit View Insert Cell Kernel Widgets Help

Run Format Code nbdiff

```
In [124]: from sklearn.ensemble import RandomForestClassifier
rf = Pipeline(steps=[ ('cat',categorical_transformer),
                      ('classifier', RandomForestClassifier())])

In [125]: print(rf._final_estimator)

RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=None, max_features='auto', max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                       oob_score=False, random_state=None, verbose=0,
                       warm_start=False)

In [126]: rf.fit(X_train, y_train)

Out[126]: Pipeline(memory=None,
                    steps=[('cat', Pipeline(memory=None,
                                           steps=[('encoder', <MyLabelEncoder.MyLabelEncoder object at 0x7ff6ba428f28>)]),
                                         trap=True, class_weight=None, criterion='gini',
                                         max_depth=None, max_features='auto', max_leaf_nodes=None,
                                         min_n_estimators=1,
                                         oob_score=False, random_state=None, verbose=0,
                                         warm_start=False))]

In [127]: y_pred = rf.predict(X_test)

In [128]: print(y_pred)

['F' 'F' 'F' 'T' 'F' 'T' 'T' 'T'
 'F' 'F' 'T' 'F' 'T'
 'T' 'T' 'F' 'T' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'T' 'F' 'T' 'F' 'F'
 'F' 'F' 'F' 'F' 'T' 'F' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'T' 'F' 'F' 'T' 'F' 'F'
 'T' 'T' 'T' 'F' 'T' 'F' 'F' 'T' 'T' 'F' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'T' 'T' 'F'
 'T' 'F' 'T' 'T' 'F' 'F' 'T' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'T' 'F' 'F' 'T' 'F'
 'T' 'F' 'T' 'T' 'F' 'F' 'T' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'T' 'F' 'F' 'T' 'F'
 'F' 'F' 'F' 'T' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'F' 'F' 'F' 'T' 'F'
 'F' 'F' 'F' 'F' 'T' 'F' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F' 'F' 'T' 'F'
 'T' 'T' 'F' 'F' 'F' 'T' 'F' 'T' 'F' 'T' 'F' 'T' 'F' 'F' 'F' 'T' 'F' 'F' 'T' 'F'
 'T' 'F' 'T' 'F' 'F'
```

Note, this is a deprecation warning. Nothing to worry. Just continue.

```
In [15]: from sklearn.ensemble import RandomForestClassifier
rf = Pipeline(steps=[ ('cat',categorical_transformer),
                      ('classifier', RandomForestClassifier())])

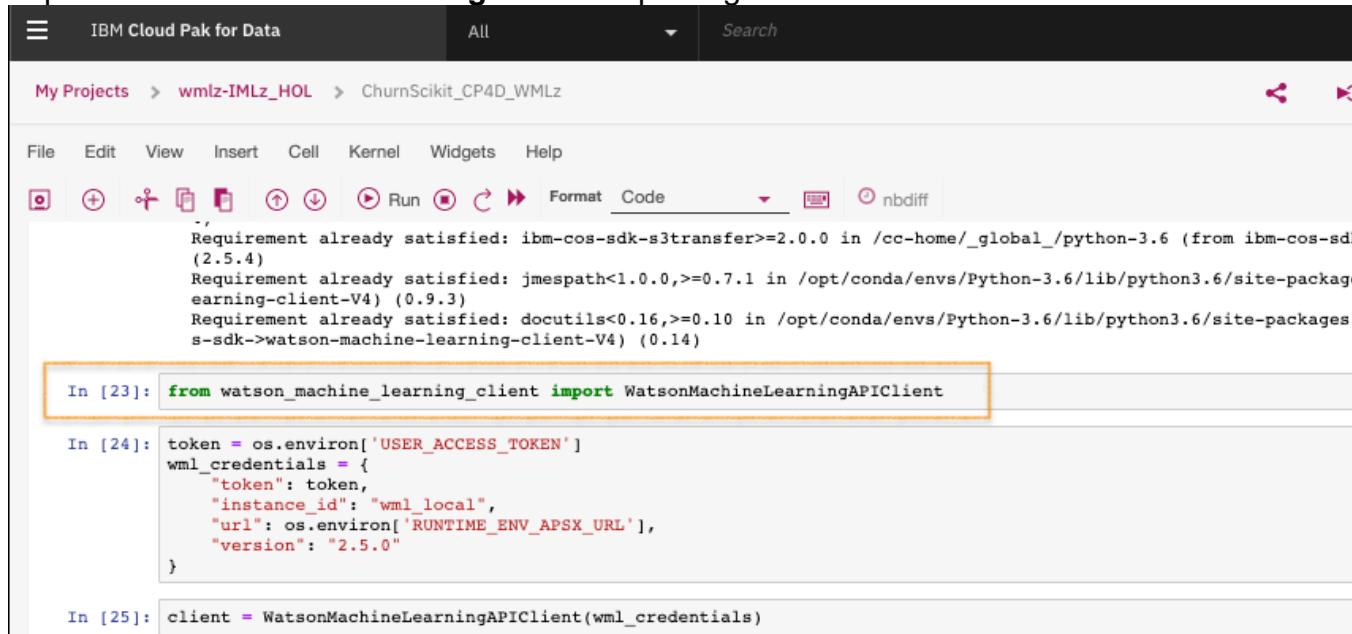
/home/wsuser/.local/lib/python3.6/site-packages/sklearn/ensemble/weight_boosting.py:29: DeprecationWarning:
umPy module and should not be imported. It will be removed in a future NumPy release.
  from numpy.core.umath_tests import inner1d
```

- 9. Last part to import **WatsonMachineLearningAPIClient** package, use default space id to save the model.

Pip install **watson-machine-learning-client-V4** package if it is not there.

```
In [22]: !pip install watson-machine-learning-client-V4
Requirement already satisfied: watson-machine-learning-client-V4 in /cc-home/_global_/python-3.6 (1.0.55)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.6/lib/python3.6/site-packages (from watson-machine-learning-client-V4)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.6/lib/python3.6/site-packages (from watson-machine-learning-client-V4)
```

Import **WatsonMachineLearningAPIClient** package.



The screenshot shows a Jupyter Notebook interface within the IBM Cloud Pak for Data environment. The top navigation bar includes 'IBM Cloud Pak for Data', 'All', and a search bar. Below the header, the path 'My Projects > wmlz-IMLz\_HOL > ChurnScikit\_CP4D\_WMLz' is visible. The toolbar contains standard options like File, Edit, View, Insert, Cell, Kernel, Widgets, Help, and various execution and format buttons. The main area displays three code cells:

```
In [23]: from watson_machine_learning_client import WatsonMachineLearningAPIClient
In [24]: token = os.environ['USER_ACCESS_TOKEN']
wml_credentials = {
    "token": token,
    "instance_id": "wml_local",
    "url": os.environ['RUNTIME_ENV_APXS_URL'],
    "version": "2.5.0"
}
In [25]: client = WatsonMachineLearningAPIClient(wml_credentials)
```

**IMPORTANT** : Update the <cp4d\_login\_userid> to your assigned HOL cp4d userid.

```
In [28]: your_HOL_USERID = "<cp4d_login_userid>"
space_name="Space@CP4D_By_" + your_HOL_USERID
space_id = client.spaces.store(meta_props={client.spaces.ConfigurationMetaNames.NAME: space_name})
```

Now save your trained model.

Your CP4D Space name : **Space@CP4D\_By\_<your\_cp4d\_login\_userid>**  
 Your CP4D Model name : **Skmodel-trained@CP4D\_By\_<your\_cp4d\_login\_userid>**

## cp4d login userid>

The screenshot shows a Jupyter Notebook interface within the IBM Cloud Pak for Data environment. The notebook has the following structure:

- Cell 79:** Imports the WatsonMachineLearningAPIClient module.
- Cell 80:** Sets up WML credentials using environment variables and defines a dictionary for the client.
- Cell 81:** Creates a client object using the WML credentials.
- Cell 82:** A comment block indicating the goal is to obtain the UID of the space.
- Cell 83:** Sets the user ID and constructs the space name by appending it to "Space@CP4D\_By\_". It also stores the space ID.
- Cell 84:** Sets the default space to the stored space ID.
- Cell 85:** Prints the metadata for the stored model, including repository details and space ID.
- Cell 86:** Stores the model in the repository, specifying the space name, model type, runtime, and space ID, and prints confirmation messages.

The output for Cell 86 shows the successful creation of a model named "Skmodel-trained@CP4D\_By\_HOLUserid2" in the "Space@CP4D\_By\_HOLUserid2" space.

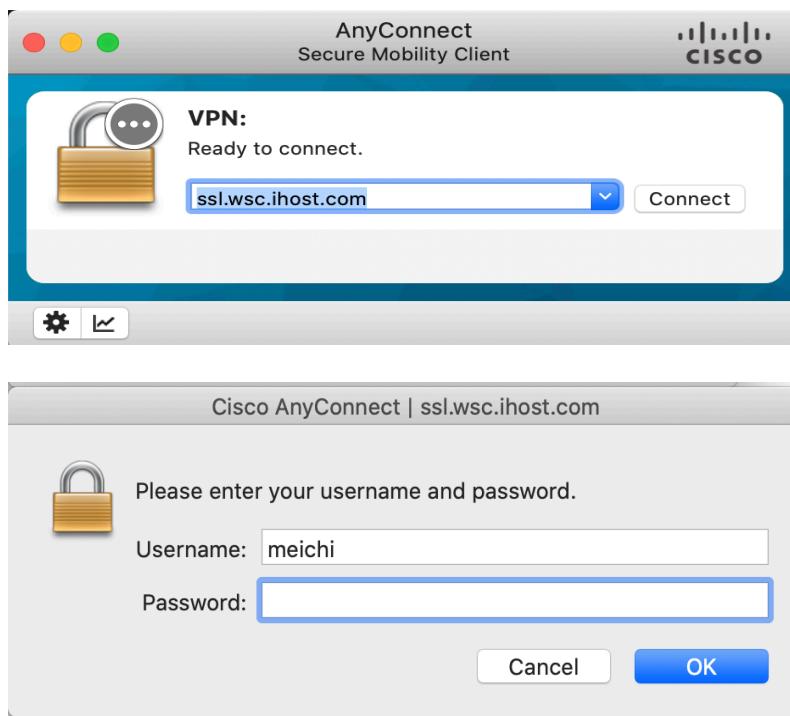
```
In [79]: from watson_machine_learning_client import WatsonMachineLearningAPIClient
In [80]: token = os.environ['USER_ACCESS_TOKEN']
wml_credentials = {
    "token": token,
    "instance_id": "wml_local",
    "url": os.environ['RUNTIME_ENV_APXS_URL'],
    "version": "2.5.0"
}
In [81]: client = WatsonMachineLearningAPIClient(wml_credentials)
In [82]: # Obtain the UID of your space
def guid_from_space_name(client, space_name):
    instance_details = client.service_instance.get_details()
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']['name'] == space_name)]['metadata']['guid']
In [83]: your_HOL_USERID = "HOLUserid2"
space_name="Space@CP4D_By_" + your_HOL_USERID
space_id = client.spaces.store(meta_props={client.spaces.ConfigurationMetaNames.NAME: space_name})["metadata"]
In [84]: client.set.default_space(space_id)
Out[84]: 'SUCCESS'
In [85]: metadata = {
    client.repository.ModelMetaNames.NAME:"Skmodel-trained@CP4D_By_"+your_HOL_USERID,
    client.repository.ModelMetaNames.TYPE: "scikit-learn_0.19",
    client.repository.ModelMetaNames.RUNTIME_UID: "scikit-learn_0.19-py3.6",
    client.repository.ModelMetaNames.SPACE_UID: space_id
}
In [86]: model_artifact = client.repository.store_model(rf, meta_props=metadata, training_data=X_train, training_target=y_train)
print ("Your CP4D Space name: " + space_name)
print ("Your CP4D Model name: " + "Skmodel-trained@CP4D_By_"+your_HOL_USERID)
print ("Your model is saved successfully")
Your CP4D Space name: Space@CP4D_By_HOLUserid2
Your CP4D Model name: Skmodel-trained@CP4D_By_HOLUserid2
Your model is saved successfully
```

You complete the first part of the lab to get the model trained and saved successfully in CP4D.

## [Log in to IBM Watson Machine Learning for z/OS](#)

The URL and authentication credentials for the WMLz Web UI are provided on a separate worksheet. Contact the lab instructor if you did not receive a worksheet.

- 10. Before you log in to IBM Watson Machine Learning for z/OS Web UI, you need to authenticate through **VPN** first. Open Cisco AnyConnect client and provide <wmlz\_vpn\_userid> & <wmlz\_vpn\_password> to get through VPN.



- 11. Start the Chrome or Firefox browser, and enter the URL assigned to you, e.g. <wmlz\_login\_url>. For the first-time logon, you are likely to be prompted with a message regarding the security of the page you are visiting. Click "Advance" and accept the exception and proceed with the intended URL.



## Your connection is not private

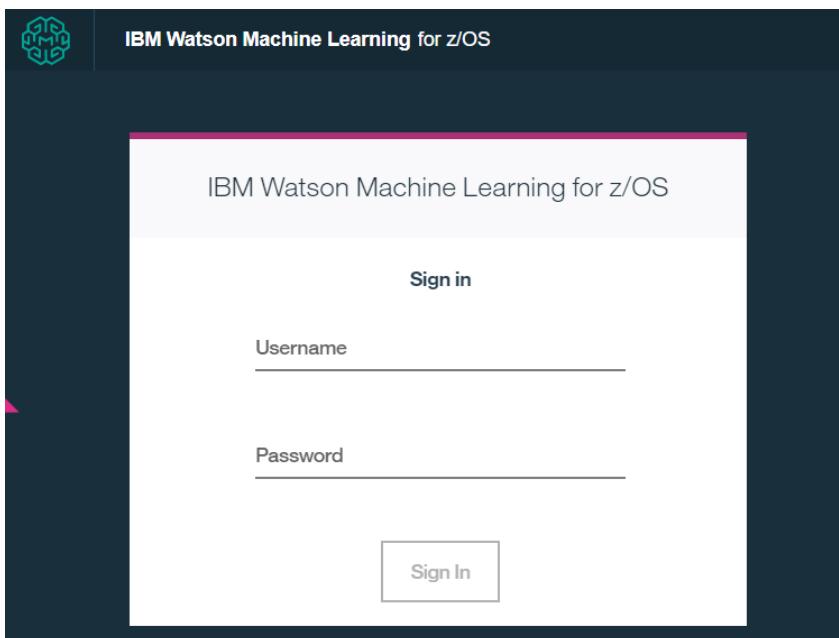
Attackers might be trying to steal your information from **mlzdl03** (for example, passwords, messages, or credit cards). [Learn more](#)

NET::ERR\_CERT\_AUTHORITY\_INVALID

Help improve Safe Browsing by sending some [system information and page content](#) to Google.  
[Privacy policy](#)



- 12. Enter the <wmlz\_login\_userid> and <wmlz\_login\_password> credentials from your lab worksheet.



You should be presented with the WMLz Model Management dashboard page

IBM Watson Machine Learning for z/OS

Model Management - Dashboard

Dashboard Models Deployments Data Sources Runtimes

**Current Model Metrics**



**0%**  
Performance

- ✓ 0 Models accurate
- ⚠ 0 Models warnings
- ❗ 0 Models errors
- ℹ 52 Models unevaluated

**Model with Warnings** [See All Models \(52\)](#)

MODEL ↑ ↓ EVALUATED VERSION ↑ ↓

**Top Deployments by API Calls** [See All Deployments](#)



## 2. Model Management and Deployment in WMLz

In this part of the Lab, you will learn as a z/OS System Administrator, how you can use WMLz to view the details of the model that your Data Scientist has created, manage the deployment of the model on your production environment, schedule the model evaluation and test the model using online RESTful API and work with your Application Developer to use the model for scoring and prediction.

### Section 1. Dashboard

- 1. The “**Model Management Dashboard**” shows the statistical information of IBM Machine Learning platform, including the model performance and overall status.

The screenshot displays the IBM Watson Machine Learning for z/OS Model Management - Dashboard. It features three main sections: "Current Model Metrics" showing a large blue circle with "0% Performance" and a summary of model status (1 Models accurate, 4 Models warnings, 12 Models errors, 6656 Models unevaluated); "Model with Warnings" listing five models with their evaluated versions (v1 or v2), publisher (wmlz11), last evaluation date, and warning count; and "Top Deployments by API Calls" listing two deployments with their date deployed, model, deployer, and API calls count.

MODEL	EVALUATED VERSION	PUBLISHER	LAST EVALUATION	WARNINGS
IML_NB_Sanity_201 9-01-31...	v1	wmlz11	Jan 31, 2019 11:59 AM	1
IML_VMB_XJ_Regression 19-01-20...	v1	wmlz11	Feb 4, 2019 5:37 PM	2
IML_VMB_Sanity_20 19-01-20...	v1	wmlz11	Feb 4, 2019 3:20 PM	1
IML_VMB_Sanity_20 19-01-23...	v1	wmlz11	Feb 5, 2019 8:22 AM	2
IML_NB_Sanity_201 9-01-20...	v2	wmlz11	Feb 4, 2019 3:25 PM	2

DEPLOYMENT	DATE DEPLOYED	MODEL	DEPLOYER	API CALLS
jason_test_test	Jan 31, 2019 6:07 AM	jason_test	wmlz11	3
NB-XGBClassifierPipelineG...	Jan 30, 2019 8:41 AM	IML_NB_Sanity_2019-01-30...	wmlz11	2

### Section 2. Model Management

- 2. Click the “**Models**” tab. The “**Models**” page is where you list and manage the Machine Learning models that have been created and exist in your environment. You can view the model details and model versions, retrain, delete and create deployment for a certain model. It also provides a filter to help you locate a specific model.

MODEL NAME	PUBLISHER	DATE PUBLISHED	MODEL TYPE	LATEST VERSION	EVALUATOR	ACTIONS
jerry-smartloan	jerry	Feb 5, 2019 11:45 AM	PMML	v1	—	⋮

- 3. Click on “**Import model**” to import models that are trained outside of WMLz. You see import From File or **From Cloud Pak 4 Data** option.  
Choose **From Cloud Pak 4 Data** option, fill in the CP4D host, username, password.  
Note, if the CP4D environment is listening on port other than 443, provide the complete CP4D host and port string in the **CP4D Host** field, such as <services-uscentral.skytap.com:13441>

IBM Watson Machine Learning for z/OS

Published Models > Import Model

## Import model

From File      **From Cloud Pak for Data**

**CP4D Host \***  
mlpattern.184.170.232.151.nip.io 48

**User Name \***  
ctp 47

**Password \***  
\*\*\*\*\* 44

**Deployment space \***  
Select deployment space ▼

**Cancel** **Import**

- 4. Once it is connected to the CP4D instance, the deployment space drop-down has a list of spaces that is available for you to choose. Choose **Space@CP4D\_By\_<your\_cp4d\_login\_userid>** from the list.

IBM Watson Machine Learning for z/OS

Published Models > Import Model

## Import model

From File      **From Cloud Pak for Data**

**CP4D Host \***  
mlpattern.184.170.232.151.nip.io  
48

**User Name \***  
ctp  
47

**Password \***  
\*\*\*\*\*  
44

**Deployment space \***  
Select deployment space ^

PruebaSalesECDeplmtSpace  
ChurnScikit\_CP4D\_2\_WMLz  
checkpyv2  
NA Test

Cancel

From File      **From Cloud Pak for Data**

**CP4D Host \***  
mlpattern.184.170.232.151.nip.io  
48

**User Name \***  
ctp  
47

**Password \***  
\*\*\*\*\*  
44

**Deployment space \***  
Select deployment space ^

PruebaSalesECDeplmtSpace  
ChurnScikit\_CP4D\_2\_WMLz  
checkpyv2  
NA Test

Cancel

- 5. Once **Space@CP4D\_By\_<your\_cp4d\_login\_userid>** deployment space is selected, go to the **Model name** drop-down list. Choose the mode **Skmodel-trained@CP4D\_By\_<your\_cp4d\_login\_userid>** model to be imported.

IBM Watson Machine Learning for z/OS

Published Models > Import Model

## Import model

From File      **From Cloud Pak for Data**

**CP4D Host \***  
mlpattern.184.170.232.151.nip.io  
48

**User Name \***  
ctp  
47

**Password \***  
\*\*\*\*\*  
44

**Deployment space \***  
ChurnScikit\_CP4D\_2\_WMLz  
▼

**Model name \***  
Select model name  
churn-scikit-trained@CP4D  
Select model name  
Cancel

- 6. Once you import the model, it takes you back to the Mode list page where you can see your imported model appear in the list.

MODEL NAME	PUBLISHER	DATE PUBLISHED	MODEL TYPE	LATEST VERSION	EVALUATE
June22	jamespак	Jan 14, 2020 11:47 AM	scikit-learn	v1	—
IMPORTED_June21_jamespак_1579031205891	jamespак	Jan 14, 2020 11:46 AM	scikit-learn	v1	—
IMPORTED_June21_jamespак_1579031051134	jamespак	Jan 14, 2020 11:44 AM	scikit-learn	v1	—
churn-scikit-trained@CP4D	mlin	Jan 10, 2020 10:26 PM	scikit-learn	v3	—
CarsModelScala-ByMaggie	mlin	Jan 7, 2020 11:49 AM	SparkML	v3	—
v210_scikit_test	jamespак	Jan 2, 2020 2:41 PM	scikit-learn	v1	—
June21	mlztest	Dec 30, 2019 7:38 PM	scikit-learn	v1	—
churnQin-z-4	mlztest	Dec 20, 2019 11:02 AM	IBM SparkML	v1	—
June21_v210	mlztest	Dec 7, 2019 11:04 AM	scikit-learn	v1	—
importFromJames1202	mlin	Nov 26, 2019 2:40 PM	SparkML	v1	—

### Section 3. Deployment Model to Scoring Server

As a System Administrator, you will need to deploy the model to your z/OS environment and test online. It may sound like a daunting task, but it can be accomplished easily in the WMLz framework:

#### 7. Deploy the model

- Click on the models tab to show the models page. You should already be on this model list page.
- On the row where the model **Skmodel-trained@CP4D\_By\_<your\_cp4d\_login\_userid>** is, click the actions menu and select “Deploy” button under “ACTIONS.”

You also have the option to deploy a model created by others.

IBM Watson Machine Learning for z/OS

Published Models

Dashboard Models Deployments Data Sources

jerry

Import model

MODEL NAME	PUBLISHER	DATE PUBLISHED	MODEL TYPE	LATEST VERSION	EVALUATOR	ACTIONS
jerry-smartloan	jerry	Feb 5, 2019 11:45 AM	PMML	v1	—	⋮

1 models in total

1 of 1 pages First View details Deploy

Setup Deployment  
Retrain  
Delete  
Publish to WML  
Export latest version

- 8. On the “**Create deployment**” page, enter the name of the deployment and select “**Online**” for the **Type**. Choose the scoring service whose name ends in “scoring-cp4d”, and then click the “**Create**” button to save.

IBM Watson Machine Learning for z/OS

Published Models > Create deployment

## Create deployment

**Model Name**  
jerry-smartloan

**Deployment name**  
jerry-smartloan-dep

**Deployment type**  
Online

**Engine**  
PMML

**Model Version**  
1

**Scoring Service**  
mlz16 (9.30.128.39:30163)

Cancel Create

Wait until the deployment completes successfully. Then locate your deployment under the “**Deployments**” page. The “**Deployments**” page is where you view the list of models that have been deployed to your z/OS environment for scoring and prediction. You can test the model here, manage the deployment and schedule evaluations to check the performance of the model on a regular basis.

The screenshot shows the 'Deployments' page of the IBM Watson Machine Learning for z/OS interface. The search bar at the top contains the text 'jerry'. The table below lists one deployment:

DEPLOYMENT NAME	DEPLOYER	MODEL NAME	SCORING SERVICE	TYPE	ENGINE	DATE DEPLOYED	NEXT EVALUATION	ACTIONS
jerry-smartloan-dep	jerry	jerry-smartloan v1	mlz16	online	PMML	Feb 5, 2019 11:52 AM	—	<span style="color: red;">⋮</span>

1 deployment in total

Click the “**View details**” button under “**ACTIONS**” to check the details of the deployment.

The screenshot shows the same 'Deployments' page as before, but with a red box highlighting the 'View details' button in the 'ACTIONS' column of the deployment row. A tooltip for this button indicates it leads to 'View details'.

Note the API details for the RESTful API provided by the scoring service of WMLz which allows your application to call the model for online scoring. In the next step, you will test this scoring API and the model online in the Web UI. If the test is successful, you can then have your application developer update the application to call this REST API. This allows you to implement online scoring on the same system as the transactions running under z/OS.

IBM Watson Machine Learning for z/OS

Deployments > jerry-smartloan-dep - Overview

DEPLOYMENT NAME: jerry-smartloan-dep

SCORING ENDPOINT: https://9.30.128.39:30163/ml/v2/scoring/online/cfc1aed4-ba8a-4185-bd85-0c961fabe380

ONLINE FEEDBACK ENDPOINT: —

API SPECIFICATION: API specification for scoring endpoint is available. [Download](#)

PUBLISHER: jerry

DATE DEPLOYED: Feb 5, 2019 11:52 AM

ASSOCIATED MODEL NAME: jerry-smartloan [\[i\]](#)

TYPE: online

ENGINE: PMML

NEXT EVALUATION TIME: —

SCORING SERVICE: miz16 (9.30.128.39:30163)

NUMBER OF INNOVATIONS: 0

AVERAGE ELAPSED TIME: 0 ms

REQUEST HEADER: See the API documentation for more details

**Model Schema**

Input Schema			Output Schema		
NAME	TYPE	NULLABLE	NAME	TYPE	NULLABLE
ds	real	false	R-loan_status	string	false
home_ownership	string	false	RC-loan_status	real	false
term	real	false	Rloan_status	string	false
total_current_balance	real	false	RP-charged_off	real	false
			RP-fully_paid	real	false

[Table](#) [JSON](#) [JSON Schema](#)

**Evaluation Results**

START TIME	END TIME	STATUS	RECORDS COUNT
no results found			

[Schedule evaluation](#)

- 9. Go back to the “**Deployments**” page, click the “**Test API**” button under “**ACTIONS**”

The screenshot shows the IBM Watson Machine Learning for z/OS interface. The top navigation bar has a logo and the text "IBM Watson Machine Learning for z/OS". Below it, a dark header bar says "Deployments". Underneath, there's a search bar with the text "jerry". The main content area has tabs for "Dashboard", "Models", "Deployments" (which is selected and highlighted in pink), and "Data Sources". A table lists one deployment: "jerry-smartloan-dep" by "jerry" using "jerry-smartloan v1" with "mlz16" as the scoring service, running online with PMML, deployed on Feb 5, 2019 at 11:52 AM. The table has columns for Deployment Name, Deployer, Model Name, Scoring Service, Type, Engine, Date Deployed, Next Evaluation, and Actions. The Actions column for this row contains several buttons: "View details", "Schedule evaluation", "Test API" (which is highlighted with a red box), "Update", and "Delete". Below the table, it says "1 deployments in total" and "1 of 1 pages".

- 10. On the “**Test API**” page, enter a record for values according to the schema of the input record of the model as below. Input values are case sensitive!

The screenshot shows the "Test API" page. At the top, it says "Deployments > Deployment: dddd > Test API". The left side has an "Input" form with fields for RATEPLAN\*, GENDER\*, ESTINCOME\*, STATUS\*, USAGE\*, and CHILDREN\*. The "Table" tab is selected, showing the following sample input record:

	Sample Input Record:
rateplan:	3
gender:	F
estincome:	50000
status:	M
usage:	712
children:	3
age:	41
dropped:	5
localbilltype:	FreeCall
longdistance:	20
local:	55
international:	1
longdistancebilltype:	Standard
paymethod:	CC
carowner:	Y

- 11. Click the “Submit” button to test the prediction. The result will be shown in “Result” area.

The screenshot shows the IBM Watson Machine Learning for z/OS interface. At the top, there's a navigation bar with a menu icon, the text "IBM Watson Machine Learning for z/OS", and a dropdown arrow. Below the navigation bar, the path "Deployments > Deployment: dddd > Test API" is visible. The main area is divided into two sections: "Input" on the left and "Result" on the right. The "Input" section contains several form fields with values: RATEPLAN \* (3), GENDER \* (F), ESTINCOME \* (30000), STATUS \* (S), USAGE \* (60), and CHILDREN \*. Below these fields are two buttons: "Clear" and "Submit". The "Result" section displays a JSON response:

```
[  
  {  
    "prediction": "F",  
    "probability": [  
      0.6,  
      0.4  
    ]  
  }  
]
```

You now complete the second part of the lab to get the CP4D trained model imported to WMLz, and deployed to WMLz scoring server, and tested a scoring call against the deployed model using WMLz!

## We Value Your Feedback!

Don't forget to submit your session feedback! Your feedback is very important to us – we use it to continually improve the lab.