

# Watson Studio: Heart Disease Modeling with Jupyter Notebooks

## Introduction

In this lab, you use a Jupyter Notebook to train a model using the XGBoost library to classify whether a person has heart disease or not. In addition to training a model, the notebook also explains how to persist a trained model to the IBM Watson Machine Learning repository, and deploy the model as a REST service.

To train and test the heart disease prediction model, you are using an open source data set published in the University of California, Irvine (UCI) Machine Learning Repository.

The notebook environment includes Python 3.6 runtime, XGBoost 0.82 and Scikit-Learn 0.20.

## Objectives

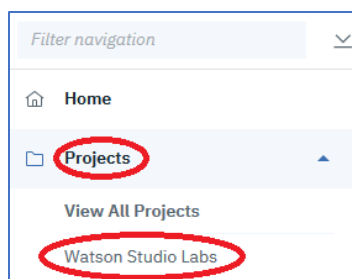
Upon completing the lab, you will know how to:

1. Load a CSV file into Pandas DataFrame.
2. Explore data using Pixiedust
3. Prepare data for training and evaluation.
4. Create, train, and evaluate a XGBoost model.
5. Visualize the importance of features that were used to train the model.
6. Use cross validation to select optimal model hyperparameters based on a parameter grid
7. Persist the best model in Watson Machine Learning repository using Python client library.
8. Deploy the model for online scoring using the Watson Machine Learning's REST APIs

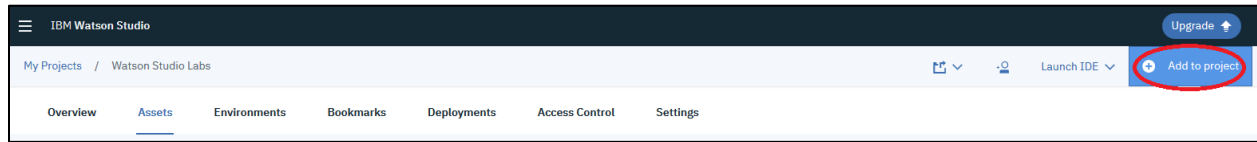
## Lab Steps

### Step 1 - Create a Jupyter Notebook

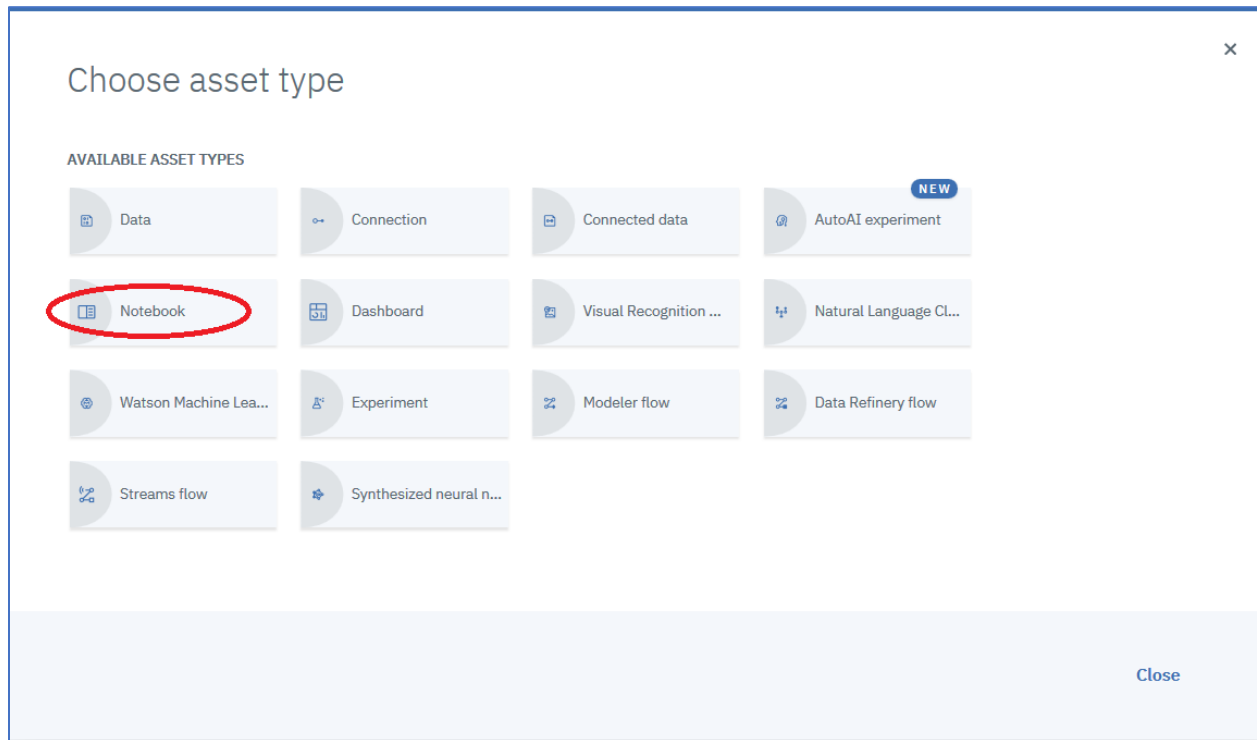
1. Click on the hamburger icon , then click on **Projects**, and then **Watson Studio Labs** (or whatever you named the project)



2. We are now going to create a notebook in our project. This notebook will be created from a url that points to the HeartDisease notebook in the github repository. Click the **Add to project** link.



3. Click on **Notebook**



4. Click on **From URL** under **New Notebook**, enter **Heart Disease** for the **Name**, and optionally enter a **Description**. Leave the default for the **Runtime**.

Cut and paste the following url into the **Notebook URL** field.

[https://github.com/bleonardb3/AA\\_POT\\_08-27/blob/master/Lab-2/Heart%20Disease.ipynb](https://github.com/bleonardb3/AA_POT_08-27/blob/master/Lab-2/Heart%20Disease.ipynb)

Click **Create Notebook**.

My Projects / Watson Studio Labs / Add Notebook

## New notebook

Blank From file **From URL**

Name **Heart Disease** 27 characters remaining

Description (optional)  
Type your Description here 500 characters remaining

Select runtime  
Default Python 3.6 XS (2 vCPU and 8 GB RAM)

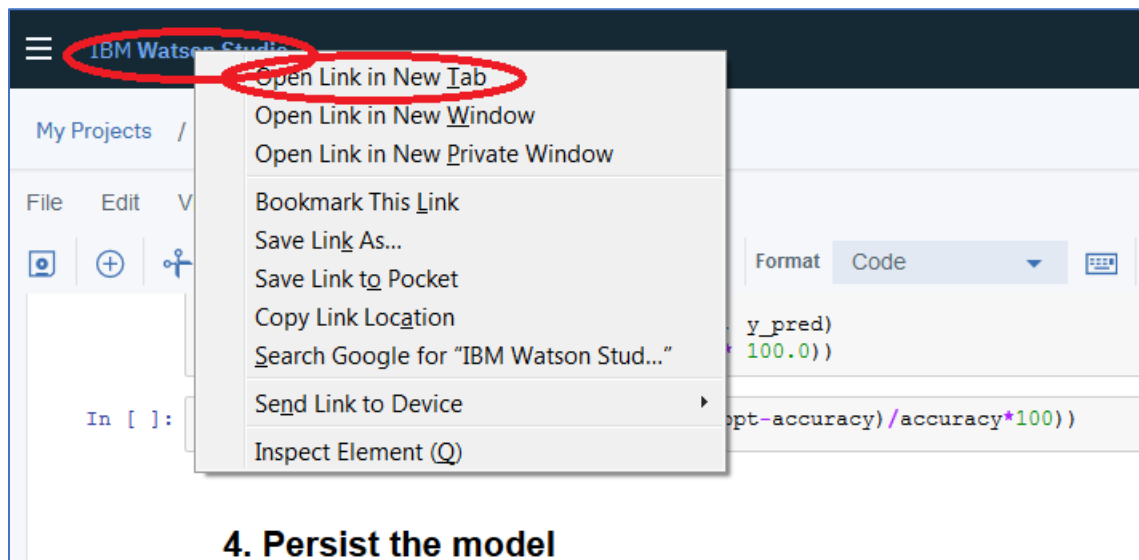
The selected runtime has 2 vCPU and 8 GB RAM and consumes 1 capacity unit per hour.  
[Learn more](#) about capacity unit hours and Watson Studio pricing plans.

Notebook URL  
**https://github.com/bleonardb3/AA\_POT\_08-27/blob/master/Lab-2/Heart%20Disease.t**

Cancel **Create Notebook**


5. Before executing the code in the notebook, we need to do the following:
  - a. Obtain the credentials of the Watson Machine Learning service and copy those credentials into a designated notebook cell. We will need these credentials to work with the Watson Machine Learning API. The procedure to obtain the credentials will be described below.

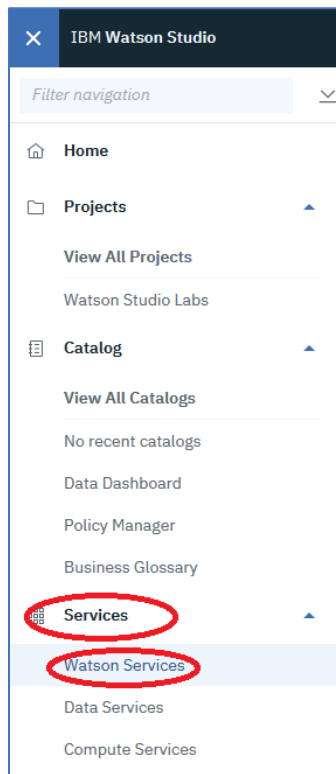
6. Right-click on **IBM Watson Studio**, and then click on **Open Link in New Tab**.




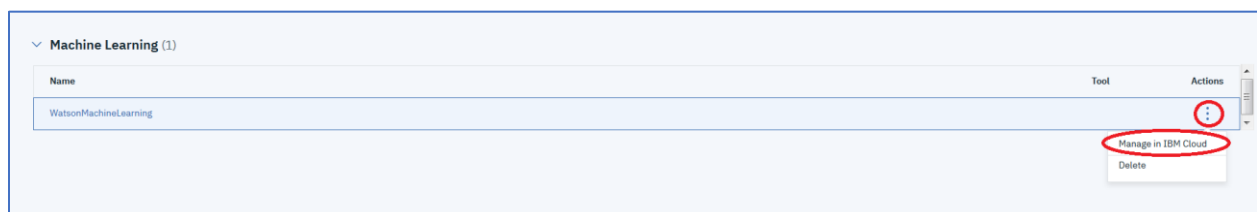
7. Click on the new **Watson Studio** browser tab.



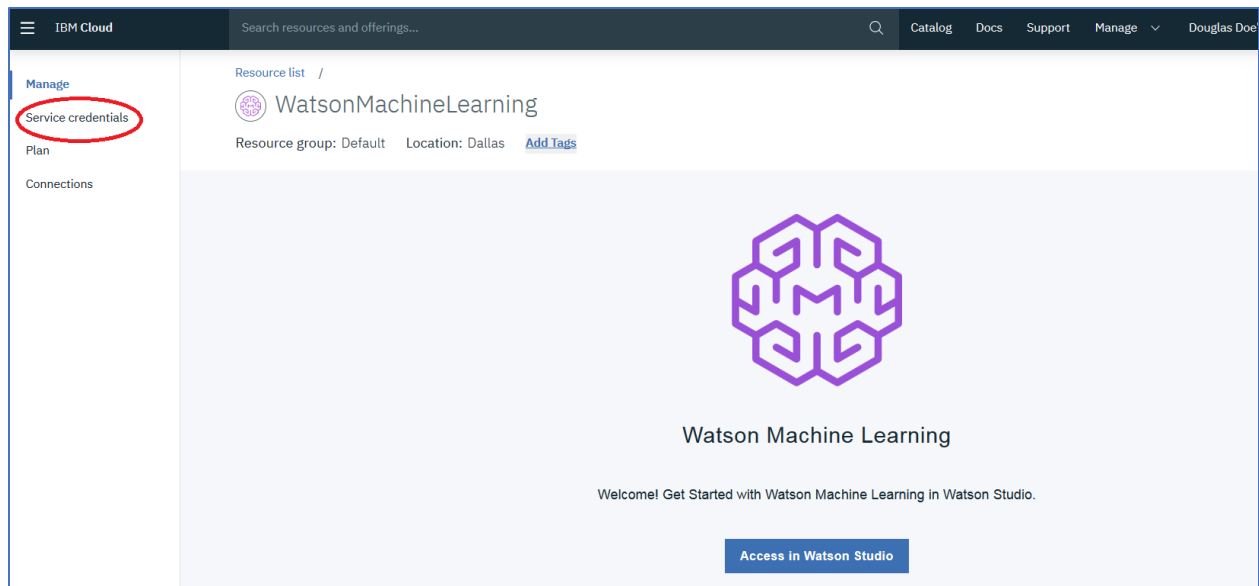
8. Click on the hamburger icon , and then **Services**, and **Watson Services**.



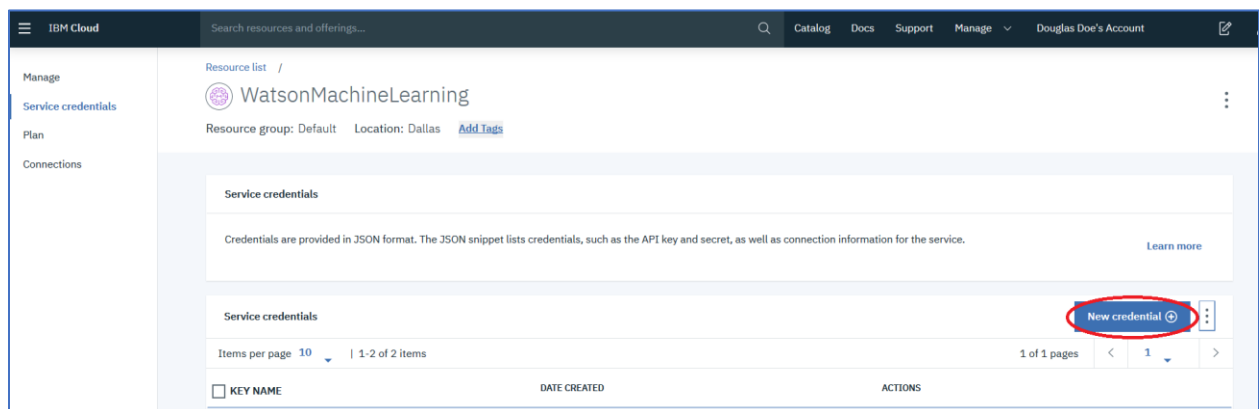
9. Hover over WatsonMachineLearning, click on the vertical ellipse on the right. , and click on **Manage in IBM Cloud**.



10. A new browser tab will be created titled **Service Details – IBM Cloud**. This browser tab will be interfacing with the IBM Cloud user interface. Click on **Service credentials**.



11. Click on **New Credentials+**.



12. Click on **Add**.

×

## Add new credential

**Name:**

Service credentials-1

**Role:**

Writer

**Select Service ID (Optional)**

Select Service ID...

**Add Inline Configuration Parameters (Optional):**

Cancel

Add

13. Click on ▼ next to View Credentials in the Service Credentials-1 row.

Service credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service. [Learn more](#)

Service credentials

New credential

Items per page 10 | 1-2 of 2 items

1 of 1 pages

KEY NAME	DATE CREATED	ACTIONS
<input type="checkbox"/> wdp-writer	JUN 25, 2019 - 02:42:46 PM	View credentials <div></div> <div></div>
<input type="checkbox"/> Service credentials-1	JUL 5, 2019 - 06:00:17 PM	View credentials <div></div> <div></div>

14. Click on the copy icon



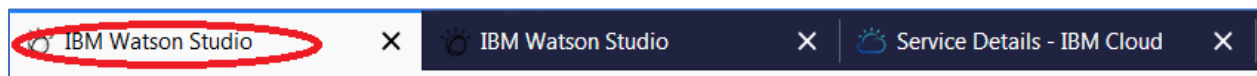
Service credentials

Items per page 10 | 1-2 of 2 items 1 of 1 pages < 1 >

KEY NAME	DATE CREATED	ACTIONS
<input type="checkbox"/> wdp-writer	JUN 25, 2019 - 02:42:46 PM	View credentials
<input type="checkbox"/> Service credentials-1	JUL 5, 2019 - 06:00:17 PM	View credentials

```
{
  "apikey": "pk3_Y0ZmsMQQFNK1ua109gn9rABBLenhAhk9-TjX7GL0",
  "iam_apikey_description": "Auto-generated for key f4f88830-8943-4f2c-a987-a9597d3455c7",
  "iam_apikey_name": "Service credentials-1",
  "iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Writer",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/ae951831c9a94c28bb9e44efe8e66ac8::serviceid:ServiceId-84c2cec8-d92c-40a2-86b9-b81cc61ba897",
  "instance_id": "2ad707c8-a75e-460c-8cb1-97dd8b302d81",
  "password": "47ee3c0f-ad6a-47ae-a771-1406c5cf76cd",
  "url": "https://us-south.ml.cloud.ibm.com",
  "username": "f4f88830-8943-4f2c-a987-a9597d3455c7"
}
```

15. Click on the Heart Disease browser tab to go back to the notebook. Should be two browser tabs to the left of the **Service Details – IBM Cloud**.



16. Type Ctrl-F and type in Paste to find the notebook cell to paste in the credentials.

In this section store the XGBoost model in the Watson Machine Learning repository by using Watson Machine Learning repository service Python client libraries.

```
In [ ]: # Install the WML client API
!pip install watson-machine-learning-client-V4

In [ ]: from watson_machine_learning_client import WatsonMachineLearningAPIClient
```

**Authenticate to Watson Machine Learning service on the IBM Cloud.**

**Action: PASTE CREDENTIALS FROM YOUR INSTANCE OF WATSON MACHINE LEARNING INTO THE FOLLOWING CELL.**

```
In [ ]: # Instantiate a client using credentials
wml_credentials = {
  "apikey": "",
  "instance_id": "",
  "url": "https://us-south.ml.cloud.ibm.com"
}

client = WatsonMachineLearningAPIClient(wml_credentials)
```

44: Save a XGBoost model in the Machine Learning Repository

Paste Highlight All Match Case Whole Words 1 of 1 match

17. Highlight the text from the starting curly brace { to the ending curly brace }. Right-click on the highlighted text and click **Paste**.

In this section store the XGBoost model in the Watson Machine Learning repository by using Watson Machine Learning repository service Python client libraries.

```
In [ ]: # Install the WML client API
!pip install watson-machine-learning
```

Authenticate to Watson Machine Learning service on the IBM Cloud.

Action: PASTE CREDENTIALS FROM YOUR INSTANCE OF WATSON MACHINE LEARNING INTO THE FOLLOWING CELL.

```
In [ ]: # Instantiate a client using credentials
wml_credentials = {
    "apikey": "",
    "instance_id": "",
    "url": "https://us-south.ml.cloud.ibm.com"
}

client = WatsonMachineLearningAPIClient(wml_credentials)
```

18. The credentials should appear similar to below with different values.

Authenticate to Watson Machine Learning service on the IBM Cloud.

Action: PASTE CREDENTIALS FROM YOUR INSTANCE OF WATSON MACHINE LEARNING INTO THE FOLLOWING CELL.

```
In [ ]: # Instantiate a client using credentials
wml_credentials = {
    "apikey": "PENUmtj4qaw5QrBilzSofqK5U1zhY4dkKXzpUjvldy3V",
    "iam_apikey_description": "Auto-generated for key 0bacef4f-b5e7-43da-a913-cd78af9a01bd",
    "iam_apikey_name": "Service credentials-1",
    "iam_role_crn": "crn:vl:bluemix:public:iam:::serviceRole:Writer",
    "iam_serviceid_crn": "crn:vl:bluemix:public:iam-identity::a/a16072b2aefd43c58409437f325c618a::serviceid:ServiceId-c96f9654-cc14-4afa-bdf7-ae7f96b89564",
    "instance_id": "e0504ccf-c440-4070-af55-2b5bdb9b7404",
    "url": "https://us-south.ml.cloud.ibm.com"
}

client = WatsonMachineLearningAPIClient(wml_credentials)
```

19. Return to the top of the notebook, read through the documentation in the beginning and then select the first code cell.

### 1. Setup

Before you execute the sample code in this notebook, you must download the **Heart Disease Data Set** data in the Notebook's local filesystem

#### Download Heart Disease Data Set to Notebook's local filesystem

The Heart Disease Data Set is a freely available data set on the UCI Machine Learning Repository portal. The **Heart Disease Data Set** is hosted [here](#).

In order to download the data from UCI Machine Learning Repository, use the `wget` library. Use the following command to install the `wget` library: `!pip install wget`.

```
In [ ]: !pip install wget
```

Now, the code in the cell below downloads the data set and saves it in the local filesystem. The name of downloaded file containing the data will be displayed in the output of this cell.

```
In [ ]: !import wget
link_to_data = 'http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data'
# make sure no duplicates
!rm processed.cleveland*.data
```

For those not familiar with Jupyter notebooks, read below.

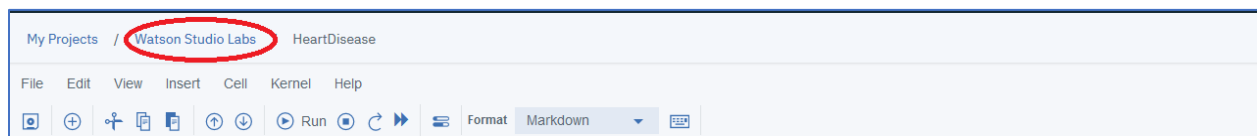
A Jupyter notebook consists of a series of cells. These cells are of 2 types (1) documentation cells containing markdown, and (2) code cells (denoted by a bracket on the left of the cell) where you write Python code, R, or Scala code depending on the type of notebook. Code cells can be run by putting the cursor in the code cell and pressing **<Shift><Enter>** on the keyboard. Alternatively, you can execute the cells by clicking on **Run icon** on the menu bar that will run the current cell (where the cursor is located) and



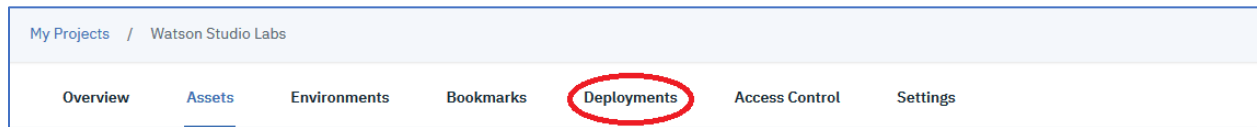
then select the cell below. In this way, repeatedly clicking on **Run** executes all the cells in the notebook. When a code cell is executed the brackets on the left change to an asterisk ‘\*’ to indicate the code cell is executing. When completed, a sequence number appears. The output, if any, is displayed below the code cell.

20. Execute each of the notebook cells in order (either by typing in <Shift><Enter> or using the **Run** menu option). Read the notebook documentation to gain an understanding of the code that is executing. **When all the cells in the notebook have been successfully executed, please return to this document, and continue with Step 21.**

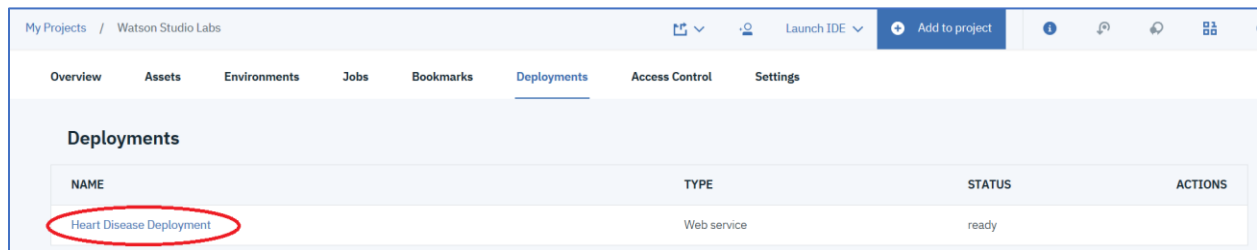
21. The notebook built, trained, saved, and deployed a machine learning model. Click Watson Studio Labs to exit out of the notebook.



22. Click the **Deployments** tab.



23. The deployment listed was generated programmatically from the notebook using the Watson Machine Learning APIs. It is a Web service deployment. Click on **Heart Disease Deployment**.



24. Click on **Implementation**.

## Heart Disease Deployment

Overview **Implementation** Test

### Deployment

Name	Heart Disease Deployment
Type	Web Service
Deployment ID	e36a838e-b6ed-4983-9e87-a137b554b260
Status	ready
Asset type	model
Asset name	Heart Disease
Machine learning service	WatsonMachineLearning
Created	13 Sep 2019 07:31pm
Last modified	17 Sep 2019 03:35pm

25. The **Implementation** panel provides sample code in various programming languages and the scoring endpoint to be used when invoking the deployed model via a RESTful interface. Click on **Test**.

## Heart Disease Deployment

Overview **Implementation** **Test**

### Implementation

View API Specification

Scoring End-point	<a href="https://us-south.ml.cloud.ibm.com/v4/deployments/e36a838e-b6ed-4983-9e87-a137b554b260/predictions">https://us-south.ml.cloud.ibm.com/v4/deployments/e36a838e-b6ed-4983-9e87-a137b554b260/predictions</a>
Authorization: Bearer <token>	Review the <a href="#">WML authentication</a> documentation for details about generating IAM tokens.
ML-Instance-ID	The "ML-Instance-ID" HTTP header must be populated with the WML instance id, which can be obtained as <a href="#">described here</a> .
Content-type: application/json	Required if the request body is sent in JSON format.

### Code Snippets

cURL   Java   JavaScript   Python   Scala



```
# TODO: manually define and pass values to be scored below
curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' --header 'Authorization: Bearer $IAM_TOKEN' --header 'ML-Instance-ID: $ML'
```

26. The **Test** panel enables testing the deployed model. Copy and then paste the following json where it says **Paste the request payload here**.

```
{"input_data": [{"fields":  
["age", "sex", "cp", "restbp", "chol", "fbs", "restecg", "thalach", "exa  
ng", "oldpeak", "slope", "ca", "thal"], "values":  
[[52.0, 1.0, 1.0, 118.0, 186.0, 0.0, 2.0, 190.0, 0.0, 0.0, 2.0, 0.0, 6.0]]}]  
}
```

## Heart Disease Deployment

Overview   Implementation   **Test**

Enter input data  

Paste the request payload here

**Predict**

27. Click on **Predict**.

# Heart Disease Deployment

Overview

Implementation

Test

Enter input data



```
{"input_data": [{"fields": ["age", "sex", "cp", "restbp", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal"], "values": [[52.0, 1.0, 1.0, 118.0, 186.0, 0.0, 2.0, 190.0, 0.0, 0.0, 2.0, 0.0, 6.0]]}]}
```

Predict

28. The prediction is no heart disease with a 95% confidence.

## Heart Disease Deployment

Overview

Implementation

Test

Enter input data



```
{"input_data": [{"fields": ["age", "sex", "cp", "restbp", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal"], "values": [[52.0, 1.0, 1.0, 118.0, 186.0, 0.0, 2.0, 190.0, 0.0, 0.0, 2.0, 0.0, 6.0]]}]}
```

Predict

```
{
  "predictions": [
    {
      "fields": [
        "prediction",
        "probability"
      ],
      "values": [
        [
          0,
          0.9500793814659119,
          0.049920618534088135
        ]
      ]
    }
  ]
}
```

## **You have completed Lab-2!**

- ✓ Loaded a CSV file into Pandas DataFrame.
- ✓ Explored data using Pixiedust
- ✓ Prepared data for training and evaluation.
- ✓ Created, trained, and evaluated a XGBoost model.
- ✓ Visualized the importance of features that were used to train the model.
- ✓ Used cross validation to select optimal model hyperparameters based on a parameter grid
- ✓ Persisted the best model in Watson Machine Learning repository using the Python client library.
- ✓ Deployed the model for online scoring using the Python client library
- ✓ Visualized Deployment in the Watson Studio UI.
- ✓ Tested the deployment