

Build, Train, Save, Deploy and Test a Convolutional Neural Network Model using MNIST

This lab will use the [MNIST](#) computer vision data set to train a deep learning model to recognize handwritten digits. A single layer convolutional neural network will be built in the Watson Studio neural network designer, and then trained using the Watson Studio Experiment Builder. The trained model will be saved in the model repository, deployed, and then tested with sample image data. The lab consists of the following steps:

1. Set up the data files in IBM Cloud Storage.
2. Design the neural network
3. Train the model
4. Monitor the training progress and results
5. Save and Deploy the Trained Model
6. Test the Deployment

End-to-End Data Science

The general flow of the End to End Data Science PoT will be guided by the activities shown in Figure 1- End to End Flow. The Neural Network modeler capability spans the Build Model, and Save and Deploy activities.

Watson Studio supports the Data Science Lifecycle

Build, train, deploy, and monitor at scale ML/DL workflows to infuse AI into the enterprise to drive innovation.

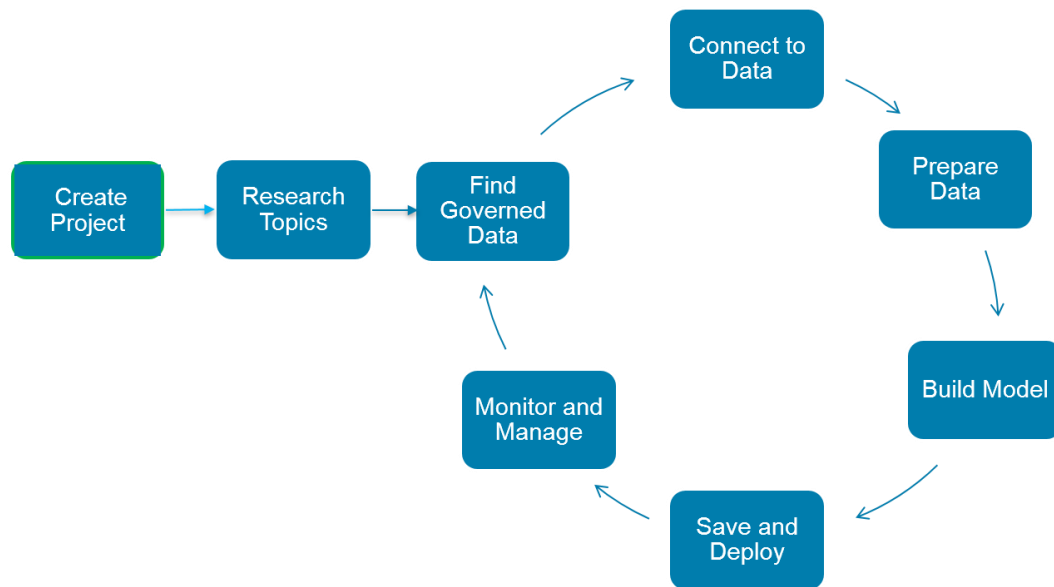


Figure 1- End to End Flow

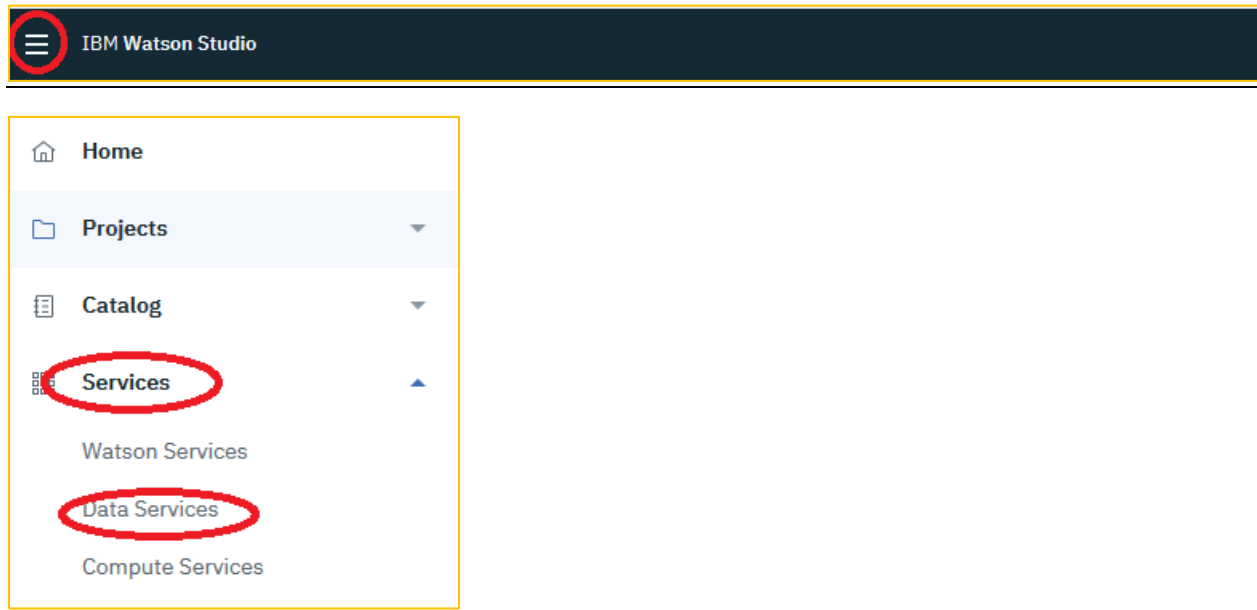
Step 1: Set up the Data Files in IBM Cloud Storage

Training a deep learning model using Watson Machine Learning relies on using Cloud Object Storage for reading input (such as training data) as well as for storing results (such as log files.)

1. Download the [mnist.zip](#) file. Extract the 3 files - a training file (mnist-tf-train.pkl), test file (mnist-tf-test.pkl), and a validation file (mnist-tf-valid.pkl) in pickle format.



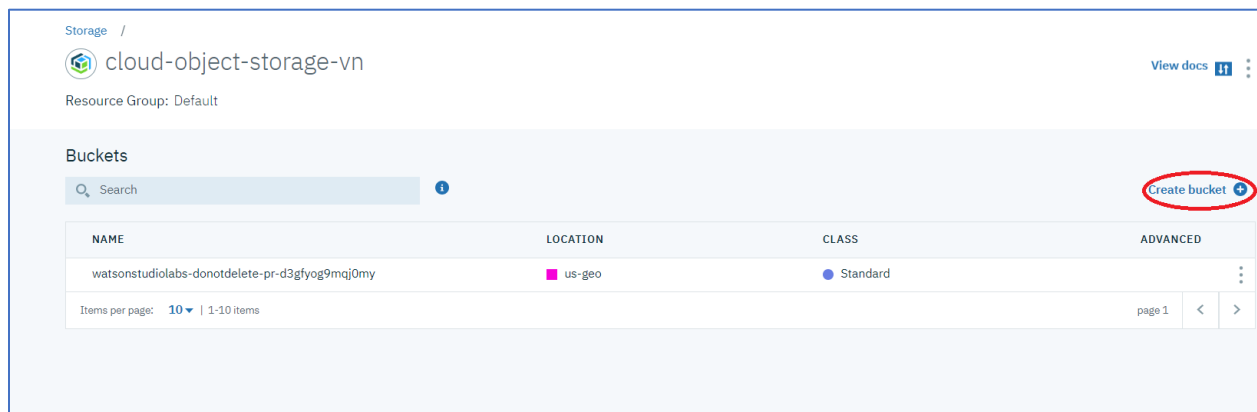
- Return to Watson Studio, click on the  icon, then click on **Services**, and then **Data Services**.



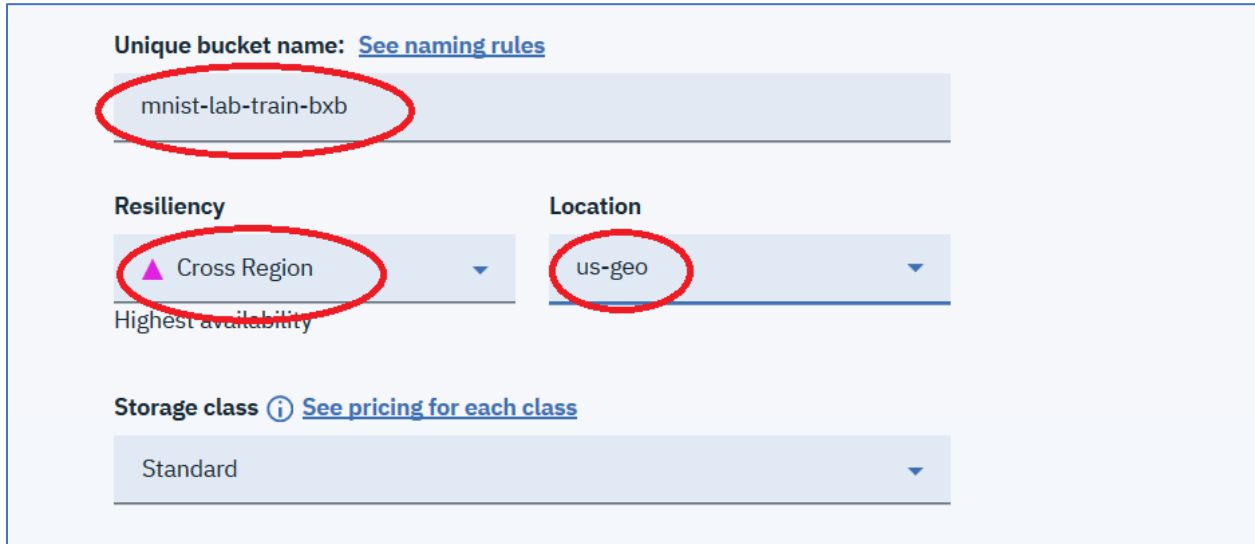
- Select the vertical **ellipse** on the right-hand side of the cloud object storage entry, and then click on **Manage in IBM Cloud**.



- Click on **Create bucket**



5. Enter a unique name for the bucket - mnist-lab-train-xxx (replace xxx with your initials), click on **Cross-Region** for the **Resiliency**, and click on **us-geo** for the **Location**. **MAKE SURE YOU CHANGE THE LOCATION TO US-GEO BECAUSE IT DEFAULTS TO AP-GEO**. Scroll down and click on **Create bucket**.



Unique bucket name: [See naming rules](#)

mnist-lab-train-bxb

Resiliency

▲ Cross Region

Highest availability

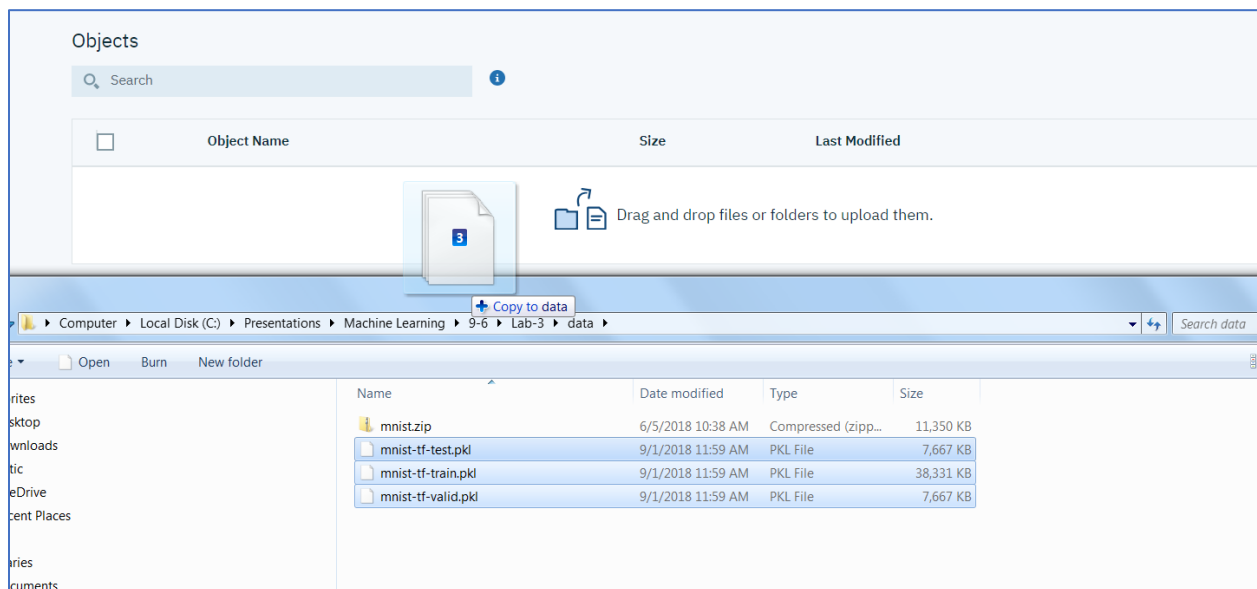
Location

us-geo

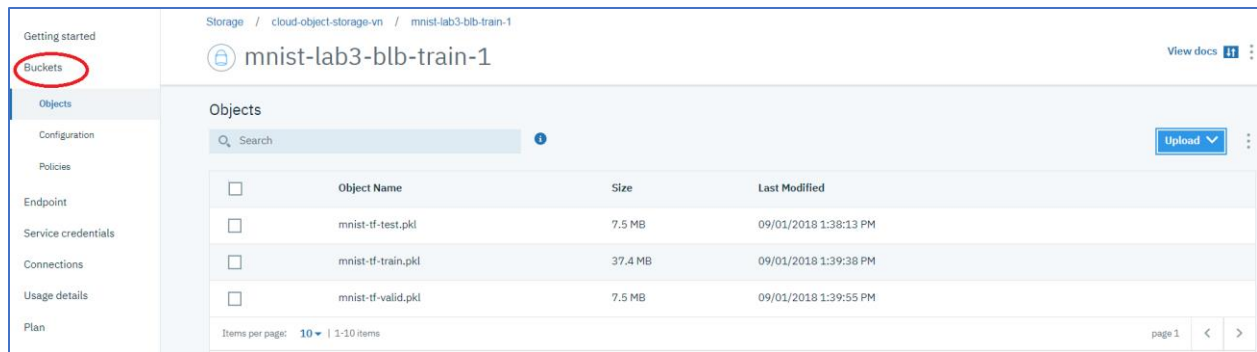
Storage class ⓘ [See pricing for each class](#)

Standard

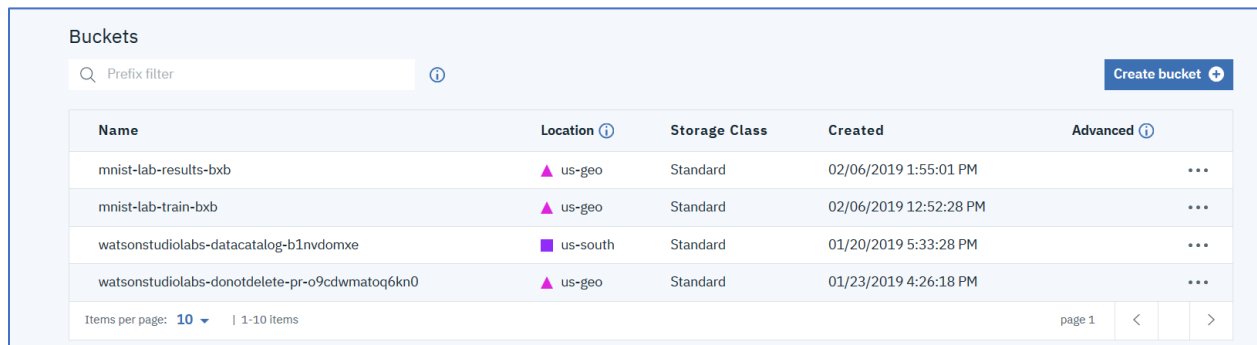
6. Navigate to the directory where the 3 mnist files are stored. Select these 3 files and drag and drop where indicated.



7. Click on **Buckets** to add a second bucket.

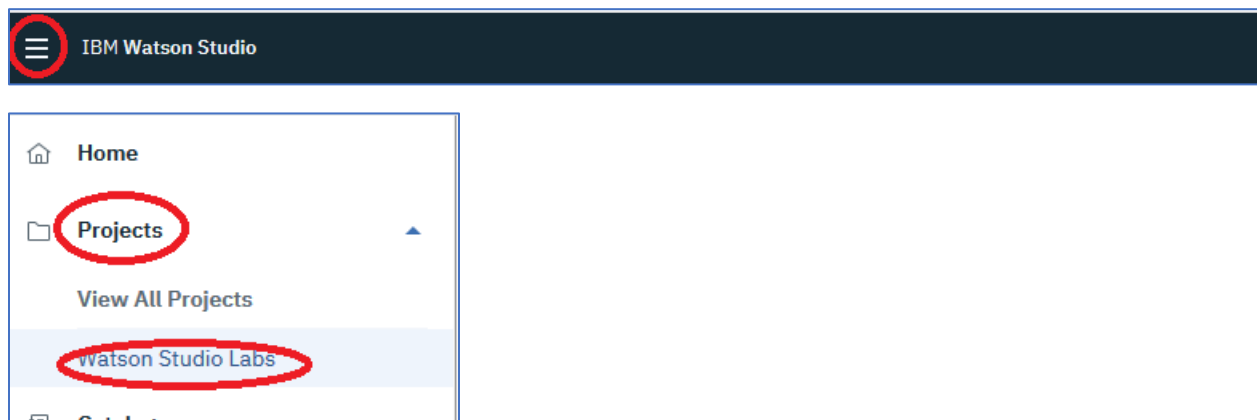


8. Name it mnist-lab-results-xxx, where xxx are your initials. Follow the procedure above to create the second bucket. No files need to be added. **MAKE SURE YOU CHANGE THE LOCATION TO US-GEO.**
9. The Cloud Object Storage panel should appear as below.

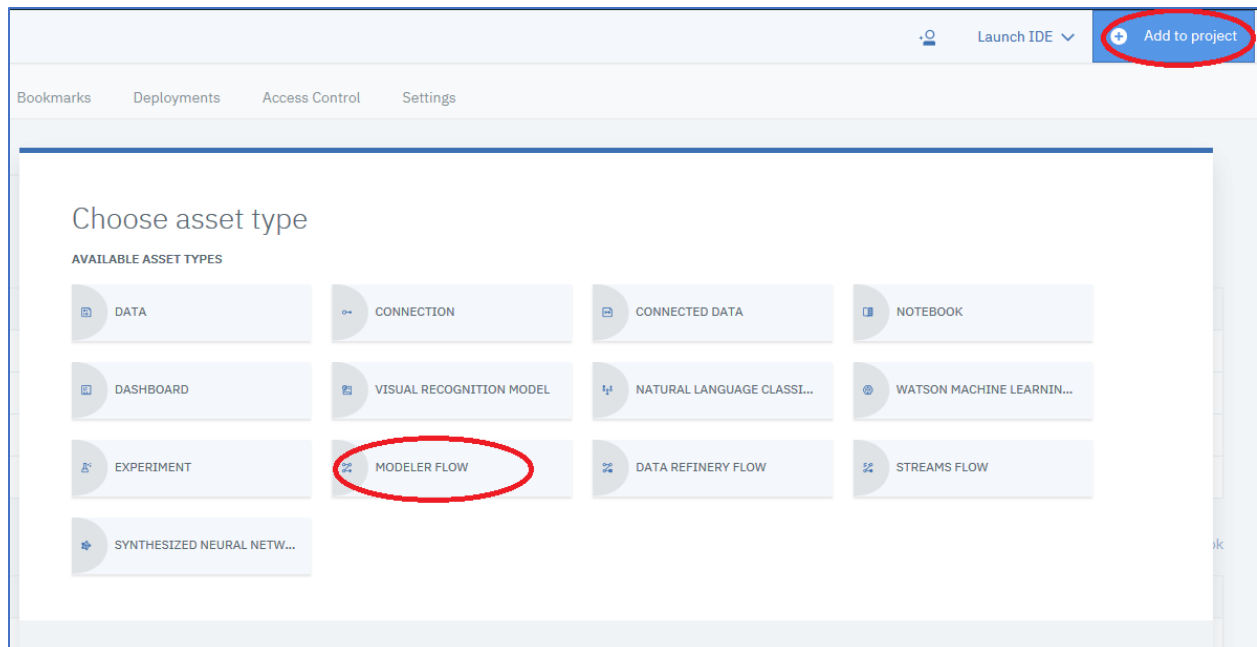


Step 2: Design the Neural Network and Publish Training Definition

1. Return to Watson Studio, and click on the  icon, then click on **Projects**, then **Watson Studio Labs**.



2. Click on the **Add to project** and then click on **MODELER FLOW**.

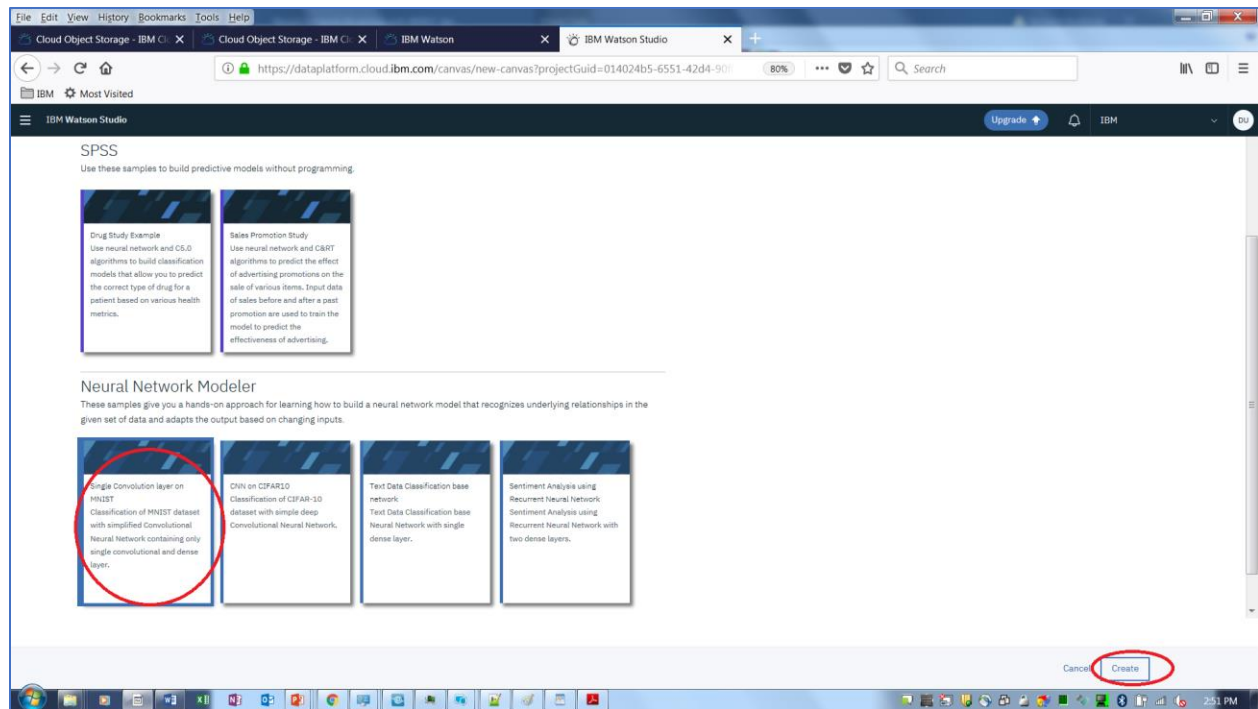


3. Click on **From example**.

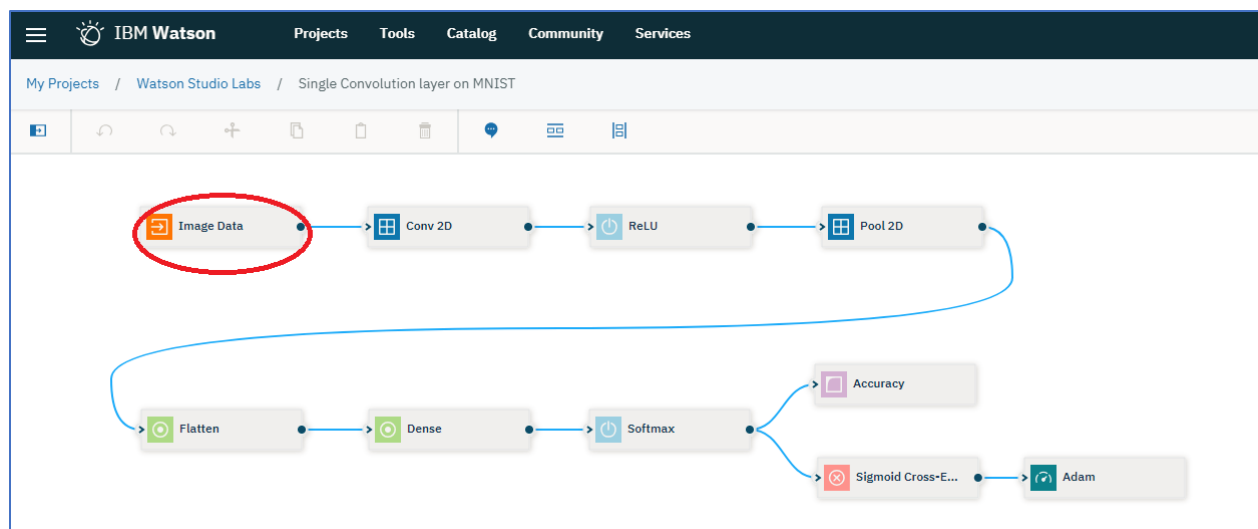
The screenshot shows the 'Modeler' form in the IBM Watson interface. The 'From example' tab is selected and circled in red. The form includes the following fields and options:

- Name***: A text input field with a placeholder 'Type name here.' and a character count of 50.
- Description**: A text area with a placeholder 'Type description here.' and a character count of 500.
- Select flow type**: Two radio buttons: 'Modeler Flow' (selected) and 'Neural Network Modeler BETA'.
- Runtime**: Two radio buttons: 'IBM SPSS Modeler' (selected) and 'Scala Spark 2.1 BETA'.

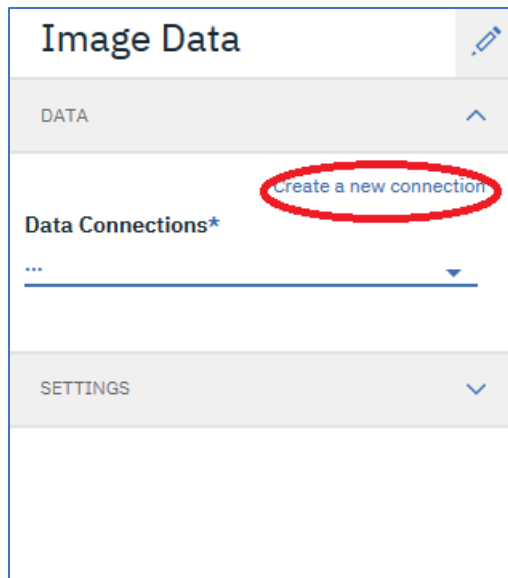
4. Click on the **Single Convolution Layer on MNIST** and then click on **Create**



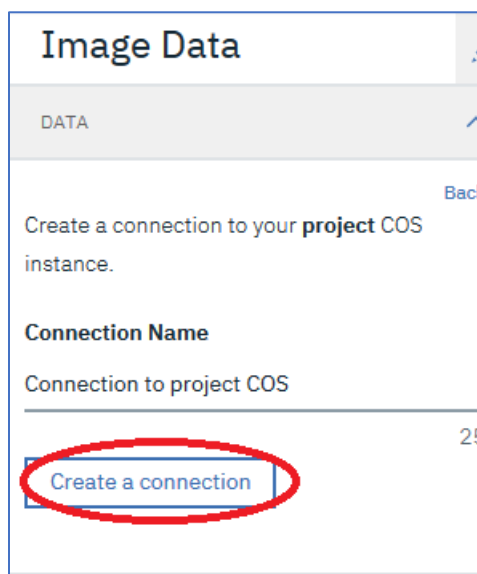
5. A standard convolution neural network (CNN) architecture is displayed. We need to configure the Image Data node. Double-click on **Image Data**.



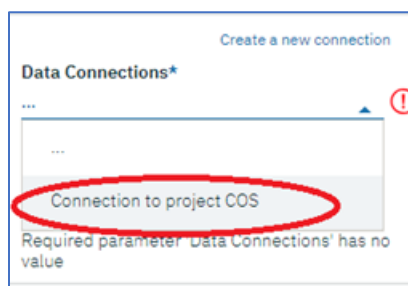
6. Click on **Create a new connection**.



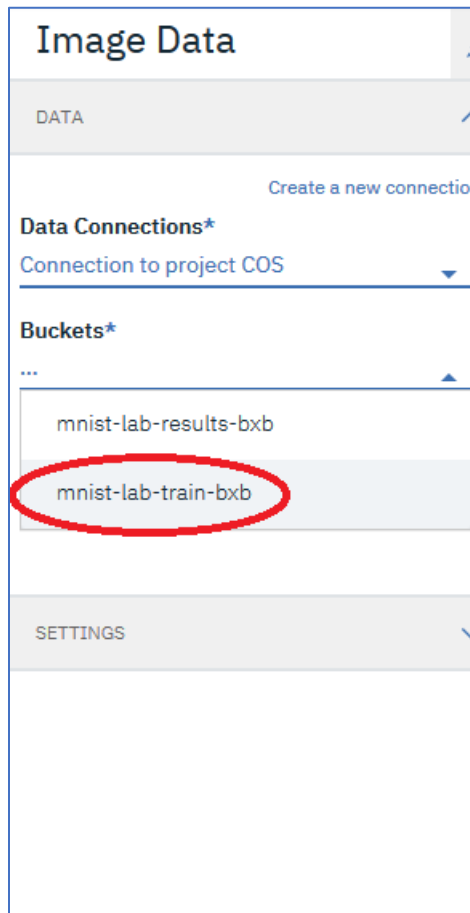
7. Leave the default **Connection Name** and click **Create Connection**.



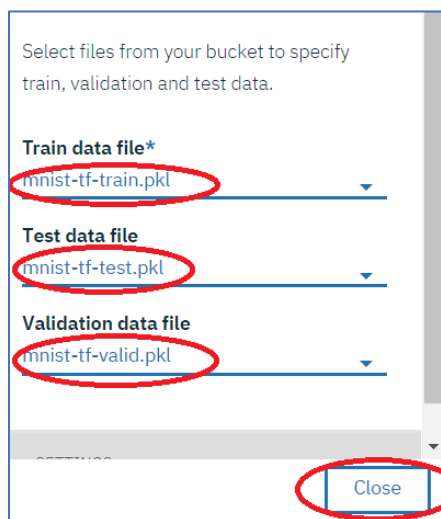
8. Click on the downward triangle icon ▼ underneath **Data Connections***. Click on the connection that was just created.





9. Click on the downward triangle icon ▼ underneath **Buckets***, and then click on the **mnist-lab-train-xxx** where “xxx” are your initials.



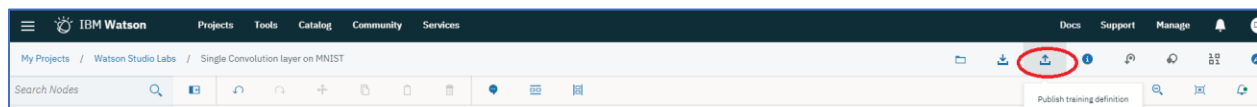
10. Click on the ▼ icon underneath **Train data file*** and select the **mnist-tf-train.pkl**. Assign the Test data file (mnist-tf-test.pkl), and Validation data files (mnist-tf-valid.pkl) in the same way and then click on **Close**.



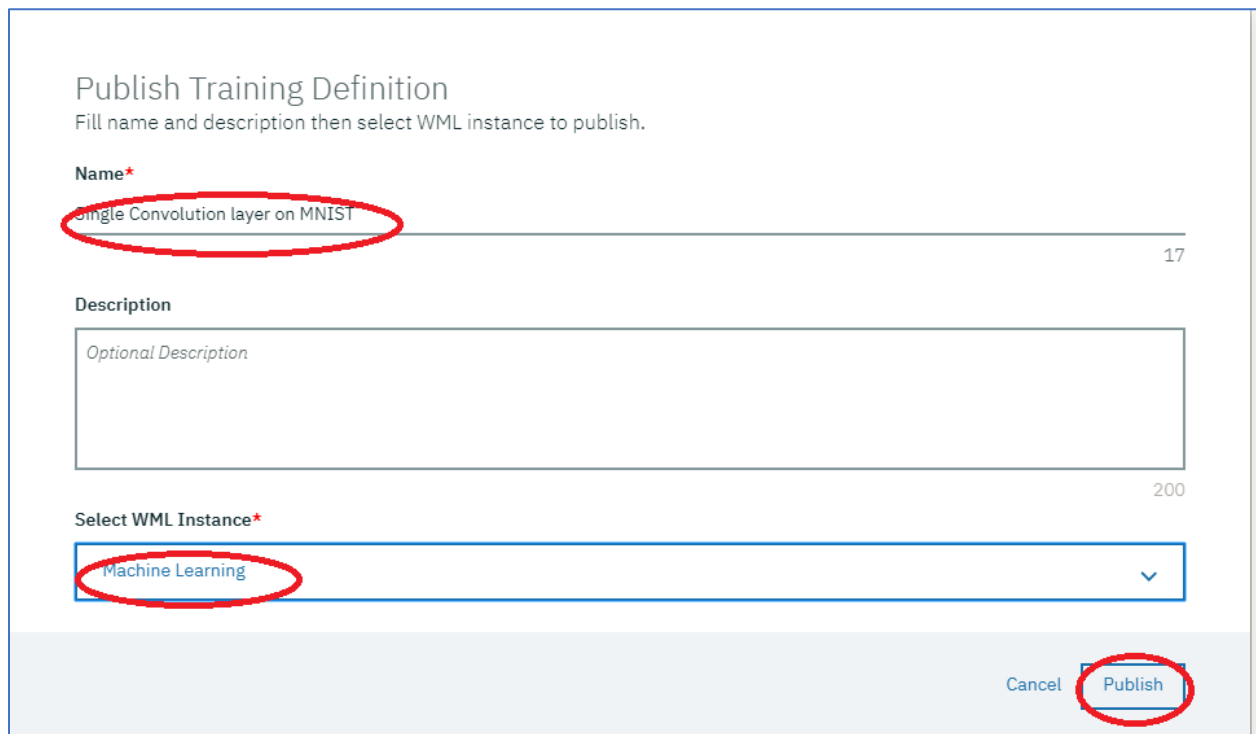
11. Explore the neural network flow modeler options

1. Click on the  icon to see the list of neural network component categories that are available
2. Explore the contents in each category. Hover over the components to get a pop-up description.
3. Drag some nodes on the canvas and double-click to see the parameters. **Note remove these nodes before doing step 12.**
4. Click on the download icon  to see the multiple options for code generation.

12. Click on the **Publish** icon to create a training definition.




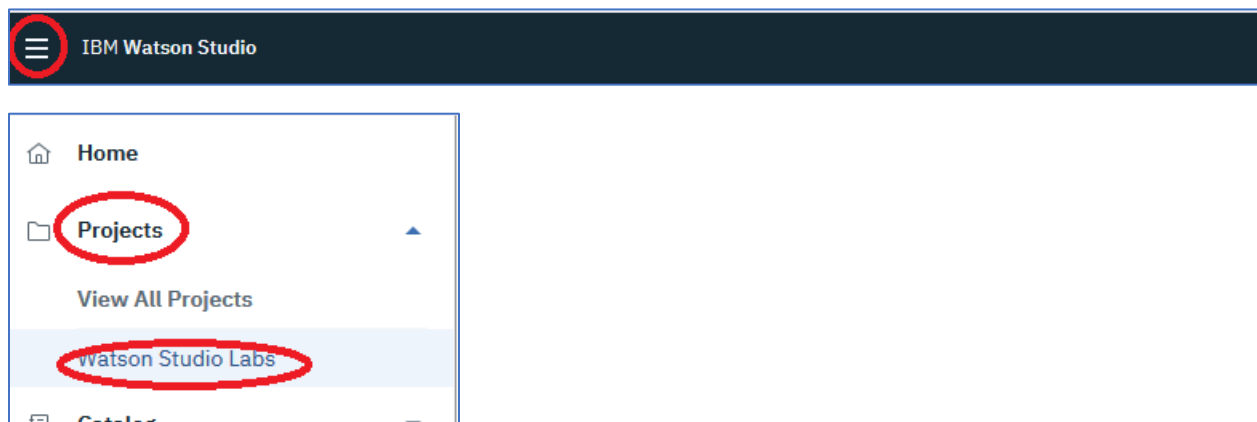
13. Enter a name for the training definition (or leave the default), and select the Machine Learning service that you created. Note, it will not be named Machine Learning unless that is the name that you used. Click on **Publish**.

A screenshot of the 'Publish Training Definition' form in IBM Watson Studio. The form has a title 'Publish Training Definition' and a subtitle 'Fill name and description then select WML instance to publish.' It contains three main sections: 'Name*' with a text input field containing 'Single Convolution layer on MNIST' (circled in red); 'Description' with a large text area containing 'Optional Description'; and 'Select WML Instance*' with a dropdown menu showing 'Machine Learning' (circled in red). At the bottom right, there are 'Cancel' and 'Publish' buttons, with the 'Publish' button circled in red. Character counts '17' and '200' are visible next to the Name and Description fields respectively.

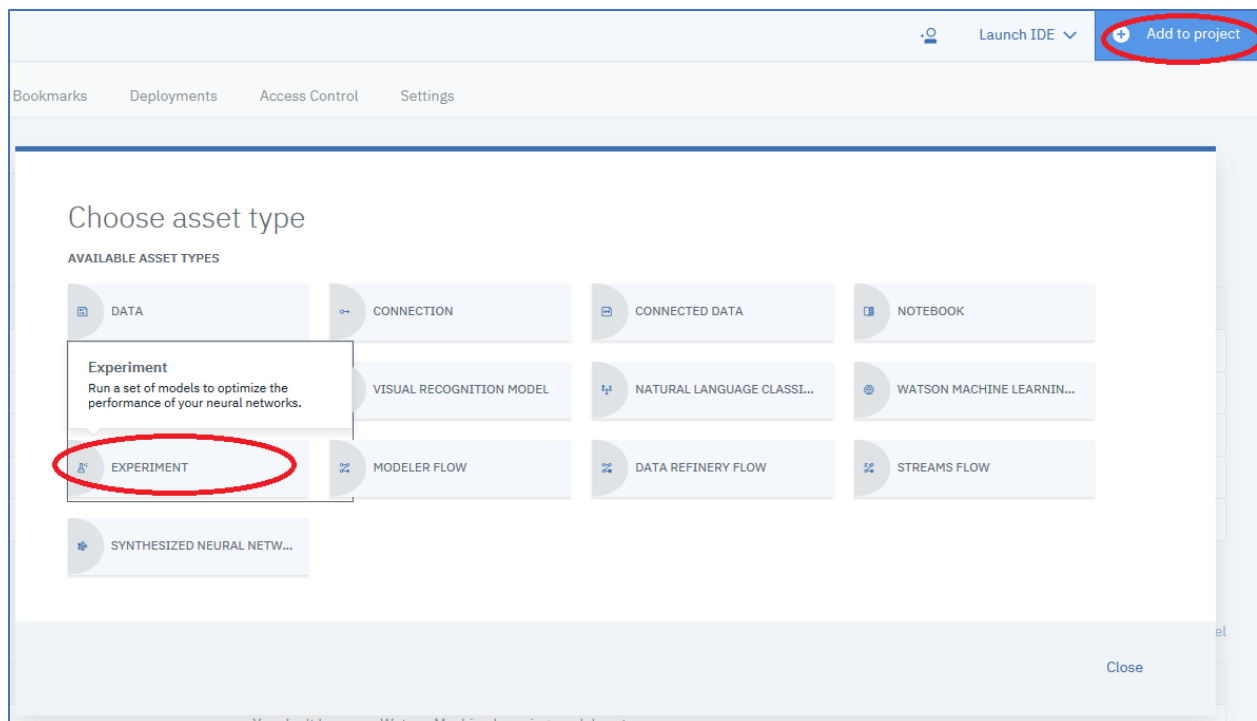
Step 3: Train the Model using Experiment Builder

As part of the model building process, we want to be able to compare different algorithms, and/or different algorithmic parameters to determine the best model. Experiment Builder is a facility in Watson Studio that supports this effort. Different training runs can be defined and run in parallel and their results can then be compared. In this lab, we will define only one training run to minimize the training time.

1. Return to the Watson Studio Labs **Assets** panel by clicking on the  icon, then click on **Projects**, and then **Watson Studio Labs**. Click on the **Assets** tab if the Assets panel is not displayed.



2. Click on **Add to project**, and then click **Experiment** to create a new Experiment.



3. Enter an Experiment **Name**, select the **Machine Learning** service, and then click on **Select** to assign a Cloud Storage bucket.

New experiment ^{BETA}

Define experiment details

Name
Single Convolution Layer on MNIST

Description
Experiment description

Machine Learning Service
Machine Learning

Cloud Object Storage bucket for storing training source and results files
Select

Associate training definitions

+ Add training definition

NAME	COMPUTE PLAN
No training definitions associated.	

☐ Use global execution command (override training definition values)

4. Select **Existing connections**, and then select the **Connection to project COS** connection.

IBM Watson Projects Tools Catalog Community Services

Cloud Object Storage bucket selection

Existing connections New connection

Cloud Object Storage connection

Select Cloud Object Storage connection

Connection to project COS

5. We now need to assign the Training and Results buckets. Select **Existing** underneath **Bucket containing training data** and click on mnist-lab-train-xxx, where “xxx” are your initials. Click on **Select**.

Existing connections New connection

Cloud Object Storage connection

Connection to project COS

Bucket containing training source data

☒ Existing ☐ New

mnist-lab-train-bxb

Cancel **Select**

6. Click on **Select** underneath **Cloud Object Storage** for storing results.

New experiment BETA

Define experiment details

Name

Single Convolution Layer on MNIST

67

Description


Experiment description

300

Machine Learning Service

Machine Learning

Cloud Object Storage bucket for storing training source files

Source: Connection to project COS / mnist-lab-train-bxb  [Update](#)

Cloud Object Storage bucket for storing results

Select

- Follow the same procedure used to assign the training bucket to assign the results bucket. Assign bucket mnist-lab-results-xxx, where “xxx” are your initials, and then click on **Select**.

Cloud Object Storage results bucket selection

Existing connections New connection

Cloud Object Storage connection
Connection to project COS

Bucket containing results data
☒ Existing ☐ New
mnist-lab-results-bno

Cancel Select

- We now need to associate a Training Definition. Click on **Add Training Definition**.

New experiment BETA

Define experiment details

Name
Single Convolution Layer on MNIST

Description
Experiment description

Machine Learning Service
Machine Learning

Cloud Object Storage buckets for storing training source and results files
Source: Connection to project COS / mnist-lab-3-train... Update
Results: Connection to project COS / mnist-lab-3-resul... Update

Associate training definitions

Add training definition

NAME	COMPUTE PLAN
No training definitions associated.	

☐ Use global execution command (override training definition values)

- Click on **Existing training definition**, and select **Single Convolution Layer on MNIST**, select **1/2 x NVIDIA Tesla K80 (1 GPU)** for the compute plan, and then click **Select**.

Add training definition

new training definition **existing training definition**

Select training definition

Existing training definitions

Single Convolution layer on MNIST

Training definition attributes

Compute plan

1/2 x NVIDIA® Tesla® K80 (1 GPU)

Hyperparameter optimization method

None

Cancel **Create**

10. Click **Create and run**.

New experiment BETA

Define experiment details

Name

Single Convolution Layer on MNIST

67

Description

Experiment description

300

Machine Learning Service

Machine Learning

Cloud Object Storage buckets for storing training source and results files

Source: Connection to project COS / mnist-lab-3-train... [View](#)

Results: Connection to project COS / mnist-lab-3-resul... [View](#) [Update](#)

If your connection is authorized for dashboard access, click the bucket name above to launch the dashboard. It may take a few seconds for the dashboard link to work for newly created buckets. Alternatively, reference the Cloud Object Storage APIs.

Associate training definitions

[+ Add training definition](#)

NAME	COMPUTE PLAN
Single Convolution layer on MNIST	1/2 x NVIDIA® Tesla® K80 (1 GPU)

☐ Use global execution command (override training definition values)

Cancel **Create and run**

Step 4: Monitor the Training Progress and Results

Training runs will be first queued, then in-process, and then completed. Use the **Training Runs** tab to keep track of progress. Usually, it will take between 3-5 minutes.

Queued Status

My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

Cancel runs in progress Add training runs

Training Runs Compare Runs Overview

1 Runs in total 0 hr, 0 min, 0 sec Total running time

Queued

NAME	SUBMITTED
Single Convolution layer on MNIST	0 hr, 0 min, 6 sec ago

In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACTIONS
No training runs found.			

In-Process Status

My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

Cancel runs in progress Add training runs

Training Runs Compare Runs Overview

1 Runs in total 0 hr, 0 min, 43 sec Total running time

Queued

NAME	SUBMITTED
No training runs found.	

In progress

NAME	DURATION
Single Convolution layer on MNIST	0 hr, 0 min, 43 sec

Completed

NAME	STATUS	DURATION	ACTIONS
No training runs found.			

Completed Status – Note the statistics on the accuracy of the training set and validation set.

Single Convolution on MNIST

Add training runs

Training Runs Compare Runs Overview

1 Runs in total 0 hr, 2 min, 13 sec Total running time

Queued

NAME	SUBMITTED
No training runs found.	

In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACC	LOSS	VAL_ACC	VAL_LOSS	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 13 sec	0.994	0.019	0.976	0.108	

1. When completed, click on the Single Convolution layer on MNIST link.

My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

1 Runs in total 0 hr, 2 min, 45 sec Total running time

Queued

NAME	SUBMITTED
No training runs found.	

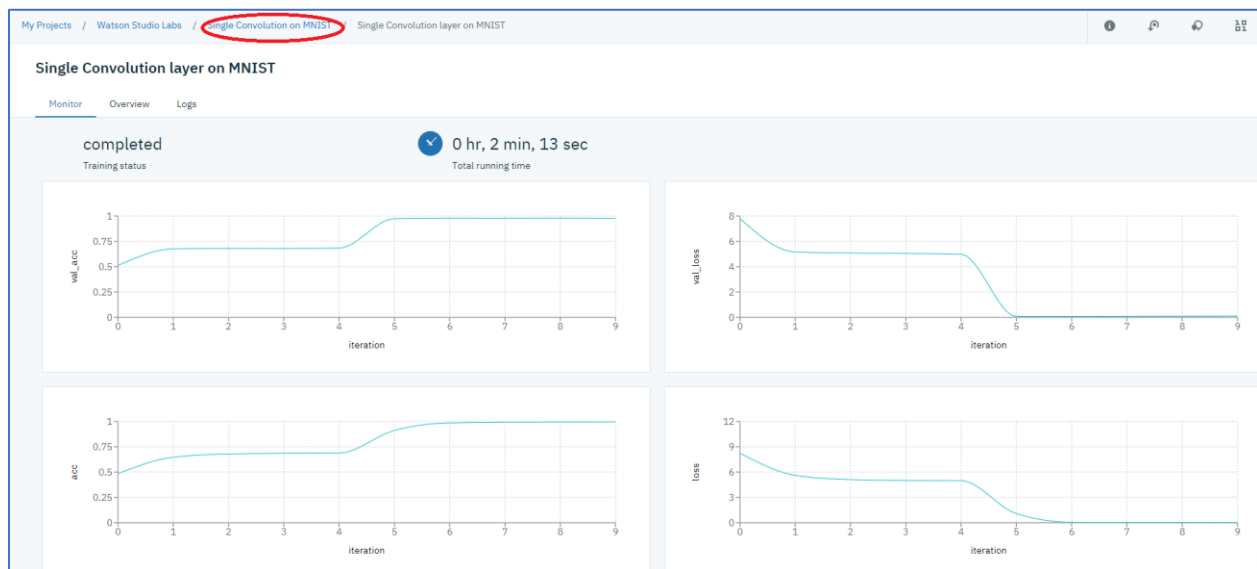
In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 45 sec	

2. The display of the statistics over the training iterations is displayed. Click on the Single Convolution on MNIST tab to return to the Training Runs screen.



Step 5: Save and Deploy the Trained Model

We will now save the trained model to the Watson Machine Learning repository.

1. Click on the vertical ellipse under ACTIONS and click **Save model**.

Completed

NAME	STATUS	DURATION	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 45 sec	

Save model

2. Enter a **Name** for the model (Single Convolution layer on MNIST) and click **Save**.

Save Model

Name
Single Convolutional layer on MNIST 65

Description
Model description 300

Cancel Save

- Return to the Watson Studio **Assets** panel, by clicking on **Watson Studio Labs** in the breadcrumb path. Click on the **Assets** tab if the Assets panel is not showing.

IBM Watson Projects Tools Catalog Community Services

My Projects / Watson Studio Labs Single Convolution Layer on MNIST

✓ Model successfully saved. View model details [here](#).

- Click on the newly saved model

Models New model

NAME	STATUS	TYPE	RUNTIME	LAST MODIFIED	ACTIONS
<u>Single Convolutional Layer on MNIST</u>	trained	tensorflow-1.5	python-3.5	5 Jun 2018	

- Click on **Deployments**.

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes 'My Projects', 'Watson Studio Labs', and 'Single Convolutional layer on MNIST...'. The main header shows 'Single Convolutional layer on MNIST' with a sub-header containing 'Overview', 'Evaluation', and 'Deployments' (circled in red). Below this is a 'Summary' section with a table of model details.

Property	Value
Machine learning service	Machine Learning
Model Type	tensorflow-1.5
Runtime environment	python-3.5
Training date	5 Jun 2018, 3:48 PM
Latest version	1c472926-e0ac-4146-9985-5b1c02bb8881

6. Click on **Add Deployment**.

This screenshot shows the 'Deployments' tab of the IBM Watson Studio interface. The main content area is empty, displaying the message 'Your model is not deployed.' Below the message is a table with columns: NAME, STATUS, DEPLOYMENT TYPE, and ACTIONS. In the top right corner of the main content area, there is a button labeled '+ Add Deployment' which is circled in red.

7. Enter a **Name** (e.g. Single Convolution layer on MNIST Deployed), select **Web Service** (should be the default), and click on **Save**.

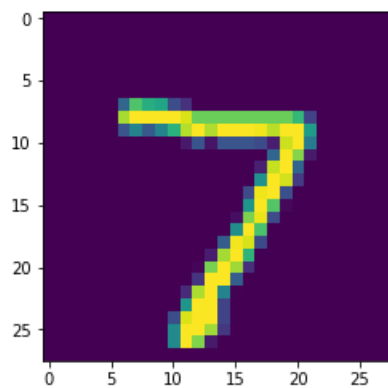
The screenshot shows the 'Create Deployment' form in IBM Watson Studio. The form has two main sections: 'Define deployment details' and 'Deployment type'. In the 'Define deployment details' section, the 'Name' field is filled with 'Single Convolutional layer on MNIST Deployed' and is circled in red. Below it is a 'Description' field. In the 'Deployment type' section, the 'Web service' radio button is selected and circled in red. At the bottom right of the form, there are 'Cancel' and 'Save' buttons, with the 'Save' button circled in red.

8. The model is successfully deployed.

Overview	Evaluation	Deployments	Lineage
<div> <div>+</div> Add Deployment </div>			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
Single Convolution Layer on MNIST Deployed	DEPLOY_SUCCESS	Web Service	⋮

Step 6: Test the Deployed Model

We will now test the deployed model using the sample image data contained in the file test.json that was extracted from the mnist.zip file previously. The test.json file is a representation of the following image.



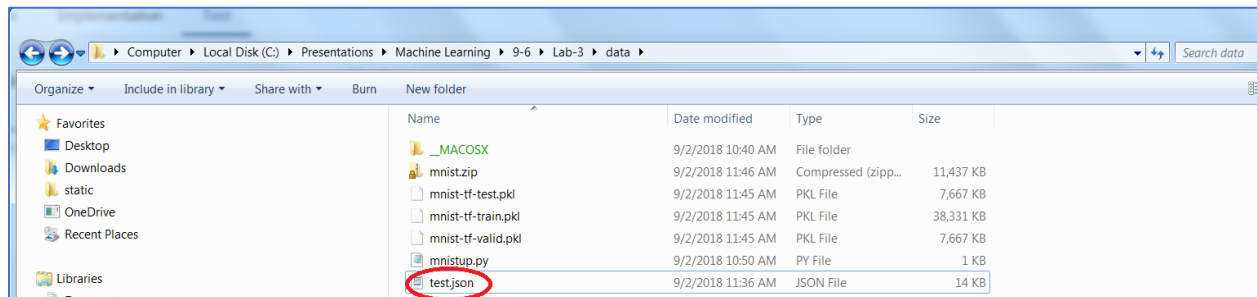
1. Click on the vertical ellipse, and then click on **View**.

Single Convolution on MNIST			
Overview	Evaluation	Deployments	Lineage
<div> <div>+</div> Add Deployment </div>			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
Single Convolution Layer on MNIST Deployed	DEPLOY_SUCCESS	Web Service	⋮
			<div>View</div> <div>Delete</div>

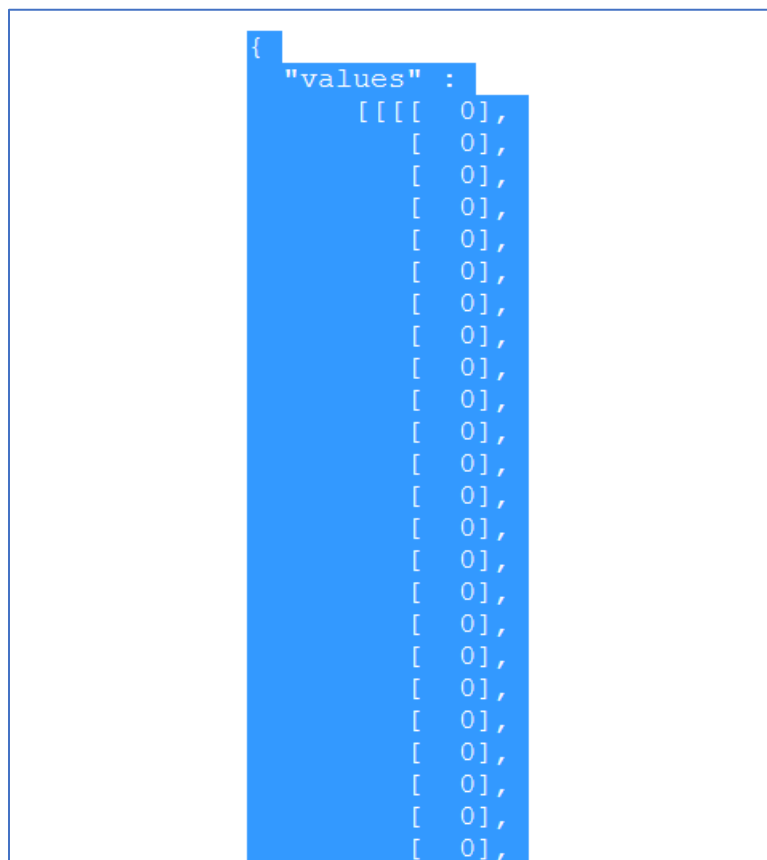
2. Click on **Test**.

Single Convolution Layer on MNIST Deployed	
Overview	Implementation
Test	
Deployment	
Name	Single Convolution Layer on MNIST Deployed
Type	Web Service
Deployment ID	89cf10e5-bd95-4b56-a728-e8b65fa98d83

3. Go to the file directory where you have the “test.json” file stored, and double-click on the file.





4. Select the contents of the file by placing the cursor to the left of the { and pressing and holding the <Shift><Ctrl><End> keys.



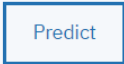
- Copy and paste the content into the **Paste the request payload here** input data section. Make sure you have both the top bracket { at the beginning of the input data section and the bottom bracket } at the end of the input section data section, and then click on **Predict**.

Single Convolution Layer on MNIST Deployed

Overview Implementation **Test**



Enter input data  

```
{  
  "values":  
    [[[[ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0]]]]  
}
```

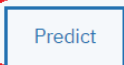


Single Convolution Layer on MNIST Deployed

Overview Implementation **Test**

Enter input data  

```
{  
  "values":  
    [[[[ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0]]]]  
}
```



6. Each of the values represent the confidence level for the digits 0,1,2,3,4,5,6,7,8, and 9. The digit that is recognized would have the largest of the confidence levels. Based on the values returned, we can see that the number 7 would be selected as the best fit for this sample image which matches the image shown above.

Single Convolution Layer on MNIST Deployed

Overview

Implementation

Test

Enter input data



```
{
  "values": [
    [[[[ 0],
      [ 0],
      [ 0],
      [ 0],
      [ 0],
      [ 0],
      [ 0]]]]
  ]
}
```

Predict

```
{
  "fields": [
    "prediction"
  ],
  "values": [
    [
      1.3815683352121771e-15,
      2.1411425399327925e-18,
      1.4536198037363307e-13,
      1.547869092014681e-13,
      3.484191859962678e-18,
      1.3316728982702908e-19,
      3.832952076167682e-23,
      1,
      1.7163898882906203e-13,
      4.053319832553193e-11
    ]
  ]
}
```