Neural Network Design Overview

This lab will introduce the IBM Flow Modeler for deep learning flows. The lab will consist of the following steps.

1. Creating the data flow (from example)
2. Creating the input data
3. Configuring the input data in the model
4. Run an Experiment on the Training Definition
5. Save and deploy the Model
6. Test the Model

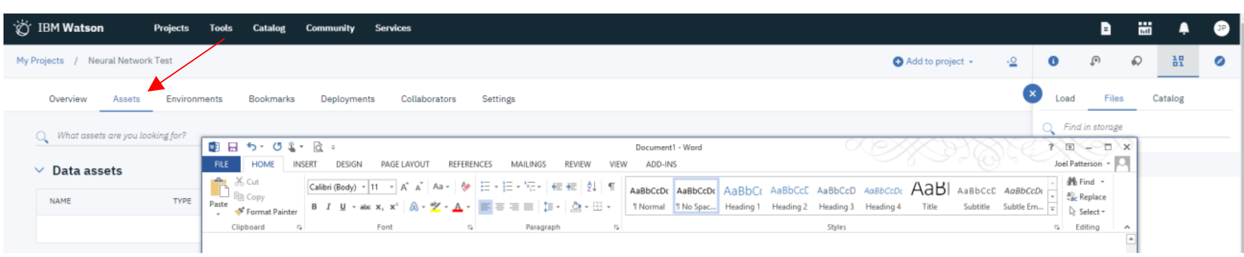
The IBM flow modeler offers a graphical interface for creating deep learning flows. Design deep models for the following types of data: image (CNN architecture), as well as text and audio data (RNN architecture). The neural network designer supports 31 types of layers. Any architecture that can be designed using the combination of these 31 layers, can be designed by using the flow modeler and then publish it as a training definition file.

From WikiPedia:
The **CIFAR-10 dataset** (Canadian Institute For Advanced Research) is a collection of images that are commonly used to train machine learning and computer vision algorithms. It is one of the most widely used datasets for machine learning research.[1][2] The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes.[3] The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.[4] Computer algorithms for recognizing objects in photos often learn by example. CIFAR-10 is a set of images that can be used to teach a computer how to recognize objects. Since the images in CIFAR-10 are low-resolution (32x32), this dataset can allow researchers to quickly try different algorithms to see what works. Various kinds of convolutional neural networks tend to be the best at recognizing the images in CIFAR-10.

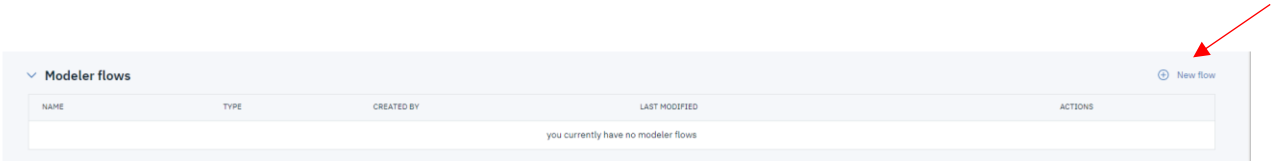
CIFAR-10 is a labeled subset of the 80 million tiny images dataset. When the dataset was created, students were paid to label all of the images.[5]

Step 1: Creating the data flow

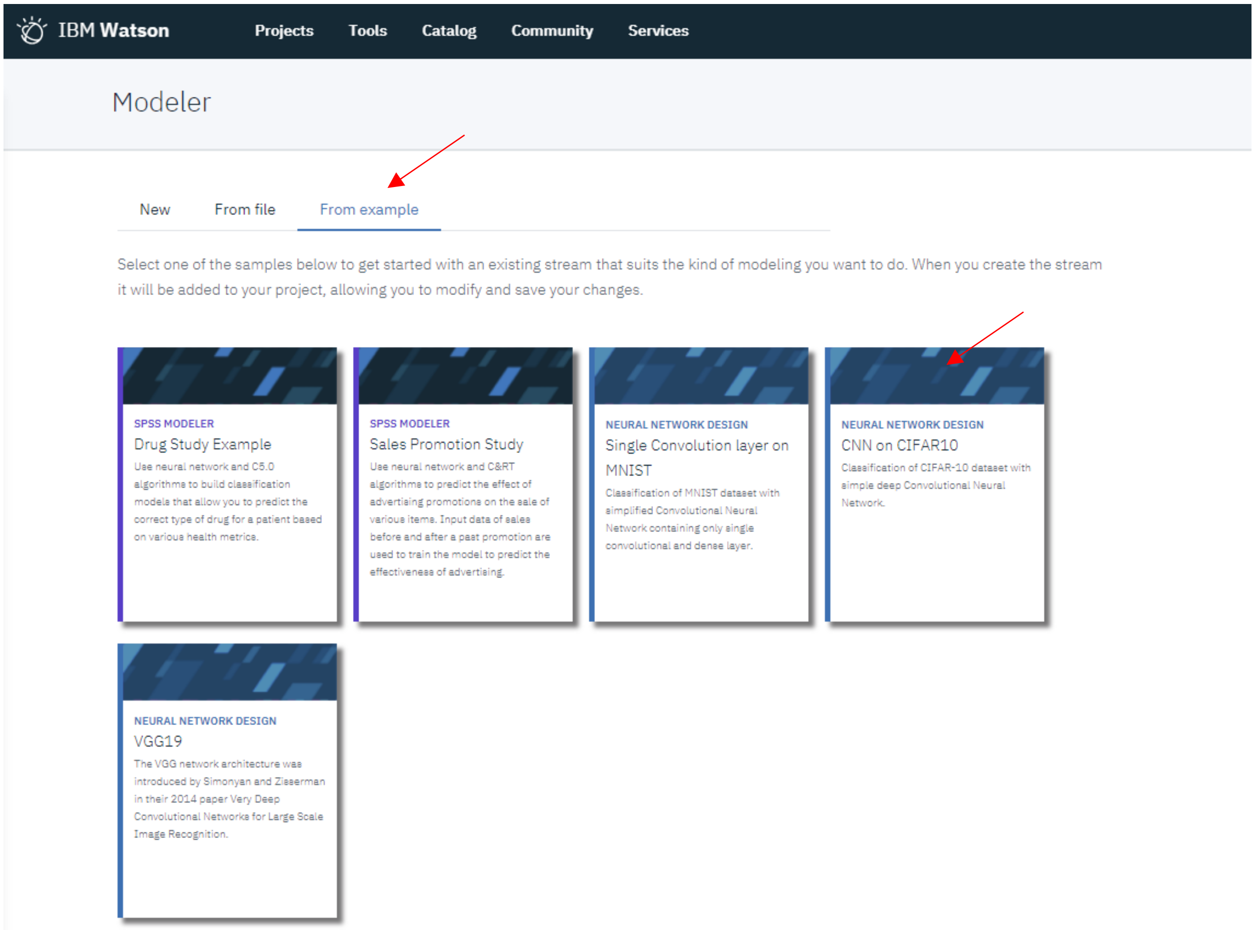
1. Select the **Assets** page of your project



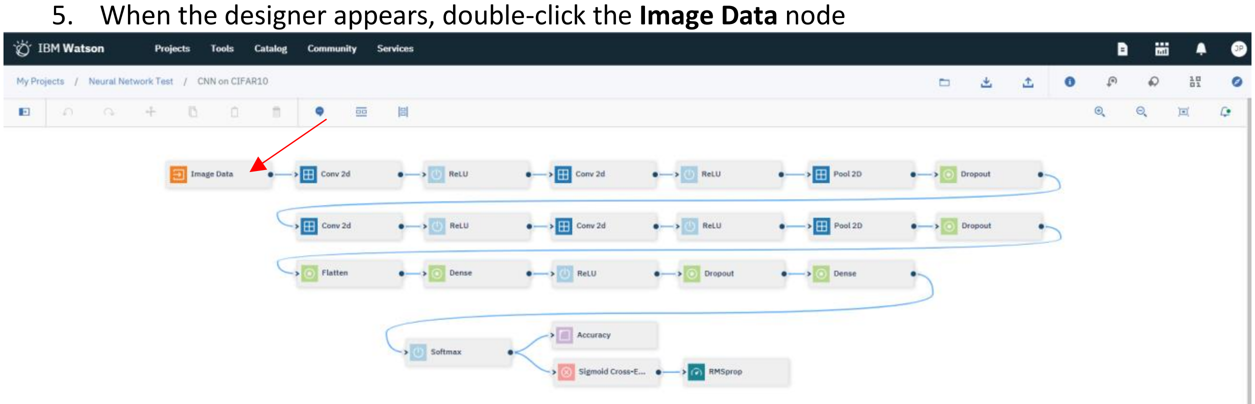
2. Scroll down to **Modeler Flows** and select **New Flow**



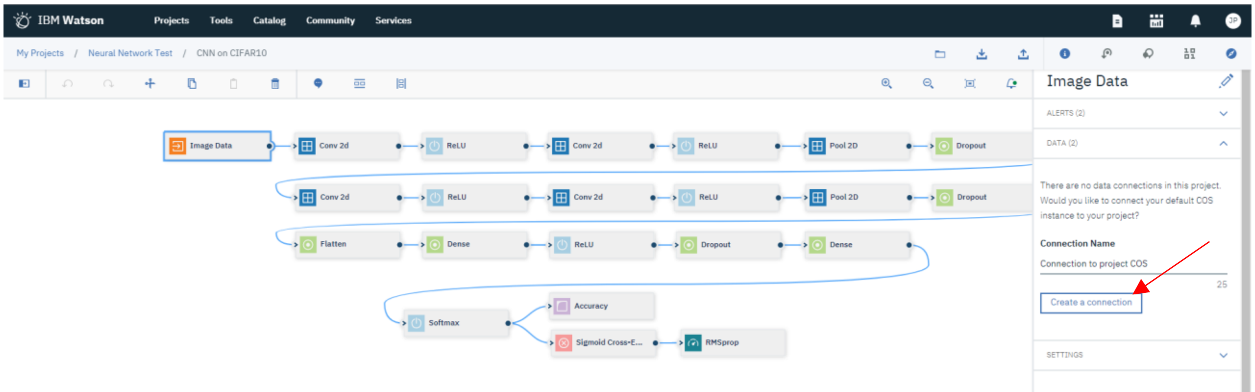
3. Select **From Example** and select **CNN on CIFAR10**



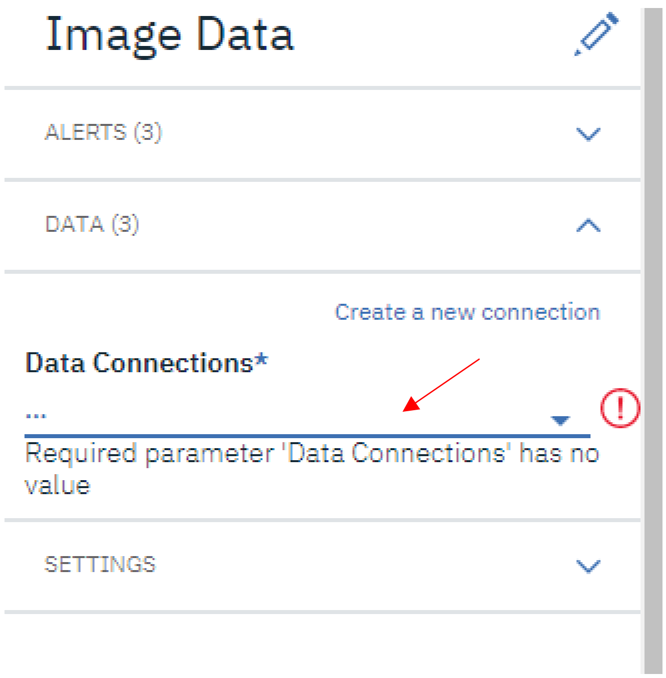
4. Select **Create**
5. When the designer appears, double-click the **Image Data** node



6. Select **Create a connection** in the **Image Data** configuration panel



7. A data connection will now be created and you should now have the option to set it. Select the **Data Connections** dropdown and set it to **Connection to project COS.**



8. We have not yet created the buckets for the input data, so **Save** this configuration

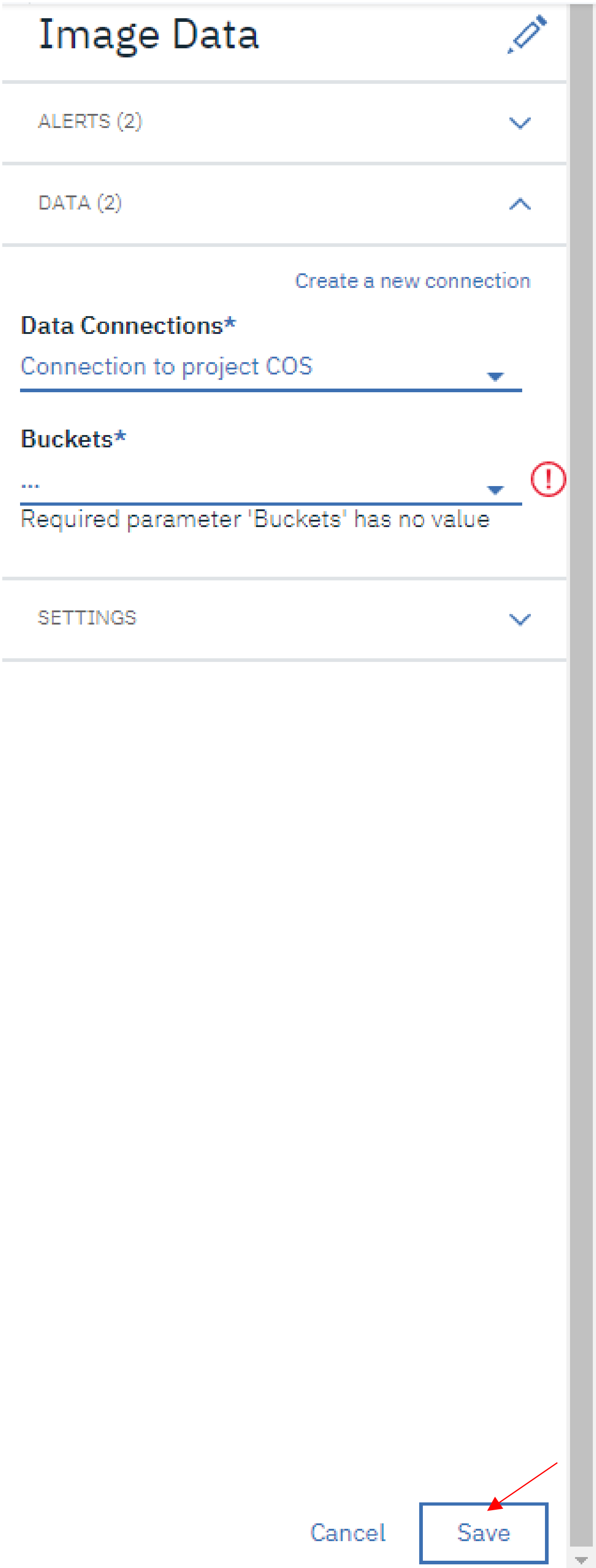
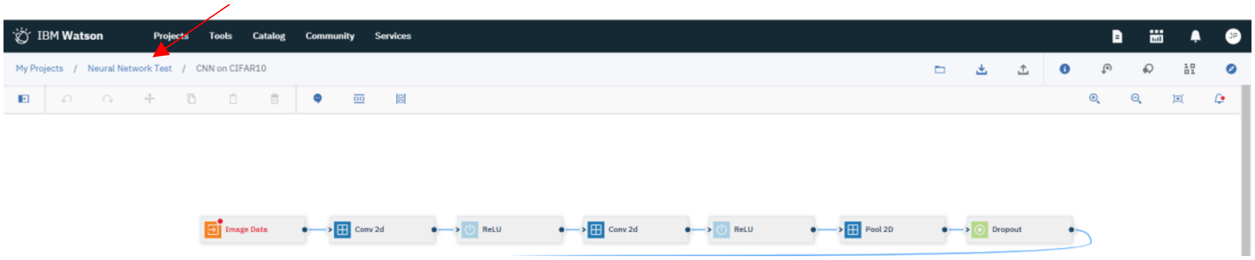


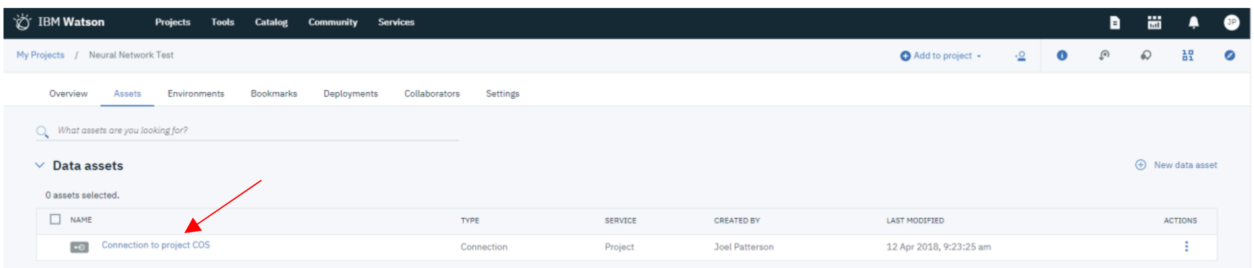
Image Data
ALERTS (2)
DATA (2)
Create a new connection
Data Connections*
Connection to project COS
Buckets*
...
Required parameter 'Buckets' has no value
SETTINGS
Cancel
Save

Step 2: Creating the input data

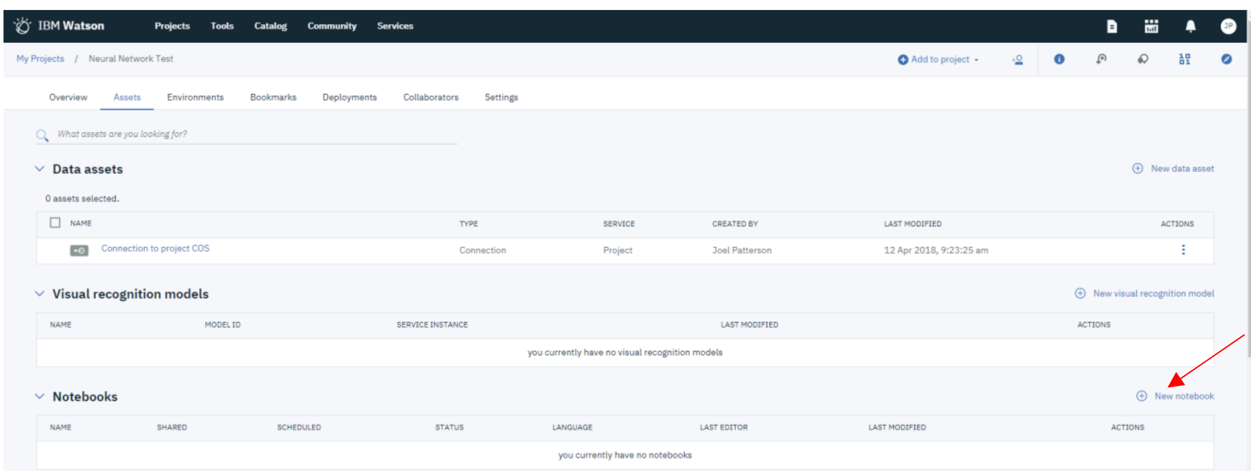
1. Return to the project by selecting the project name in the breadcrumb trail.



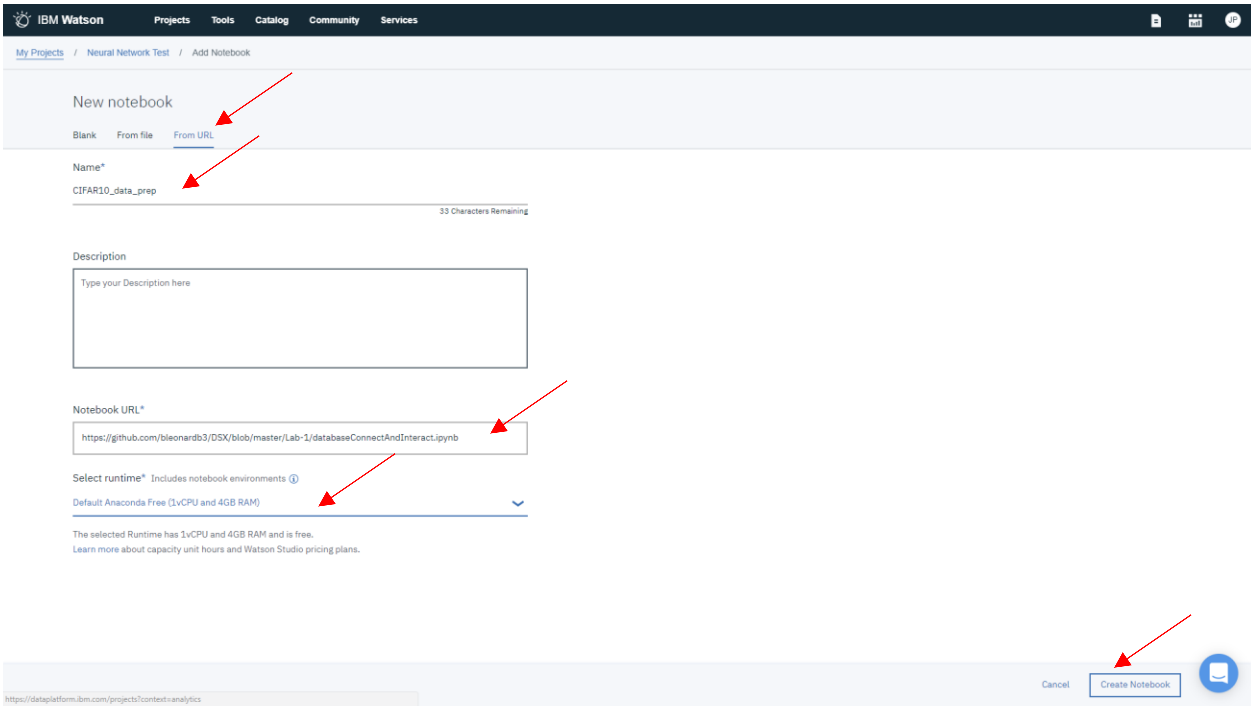
2. This returns you to the Assets page. You should see the connection to Cloud Object Storage in the Data Assets section.



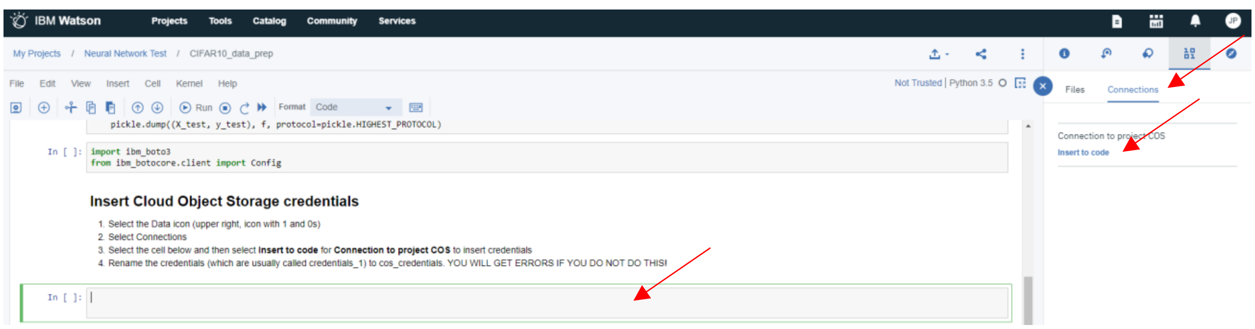
3. Add a Notebook asset



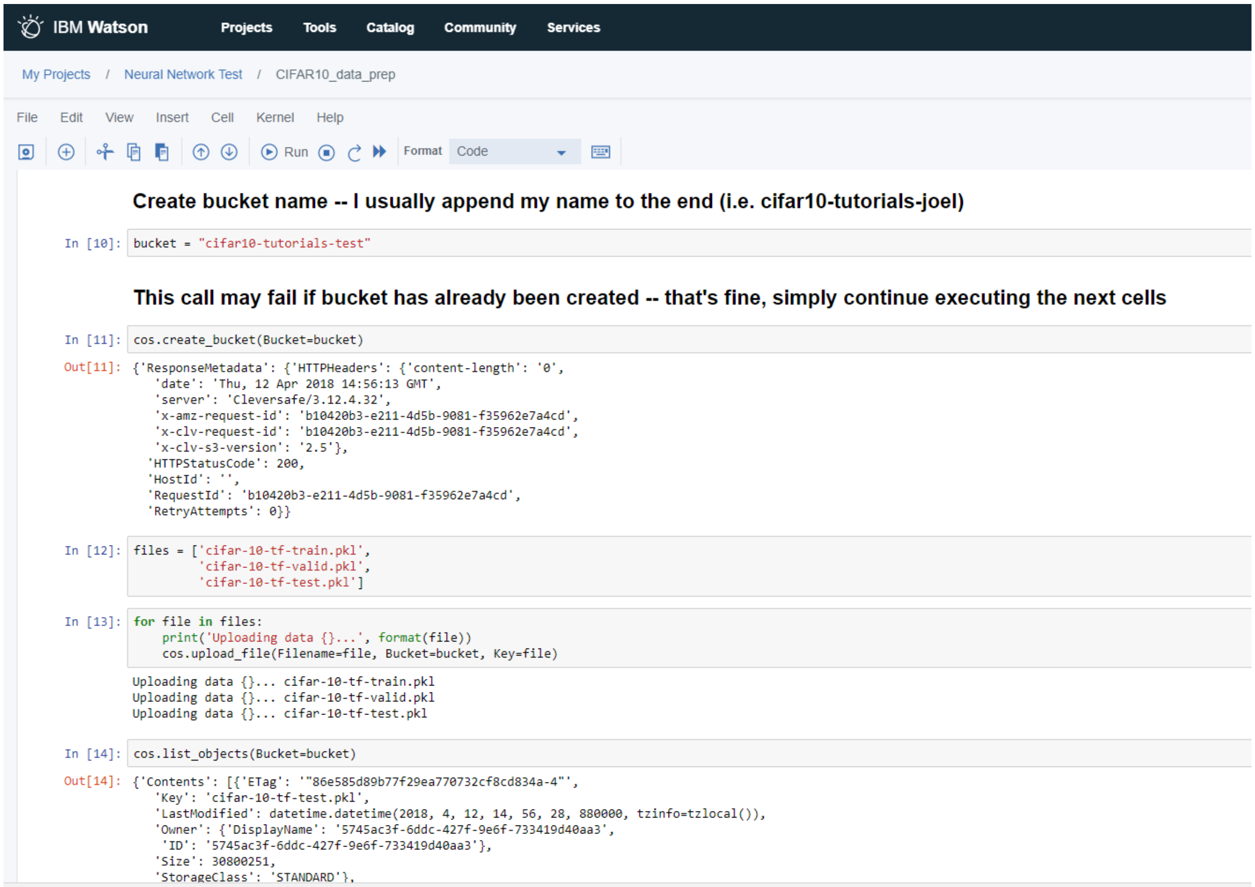
4. Select **From URL** to create the notebook from URL. Give the notebook a name (e.g Create Input Data) and optionally a description. Use https://github.com/bleonardb3/ML-POT/blob/master/Lab-3/CIFAR10_data_prep.ipynb for the Notebook url. The runtime can remain the **Default Anaconda Free** Environment because we are not using Spark here [note: only one free environment can be running at a time]. Select **Create Notebook**.



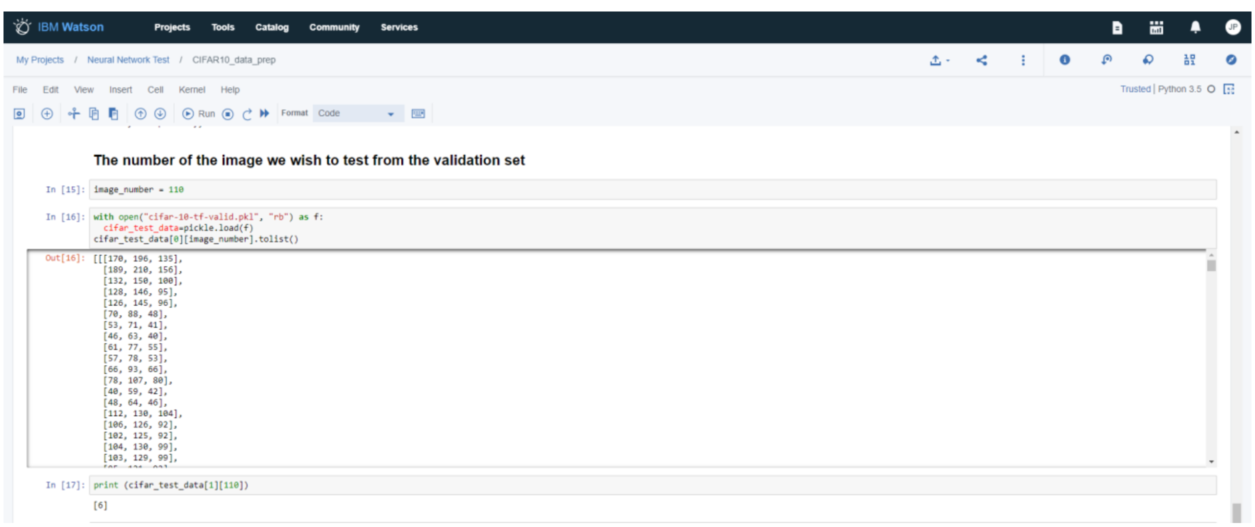
5. Run each cell in the notebook. You will need to insert the credentials to your COS store.



6. You should see something similar to the following after running the cells.

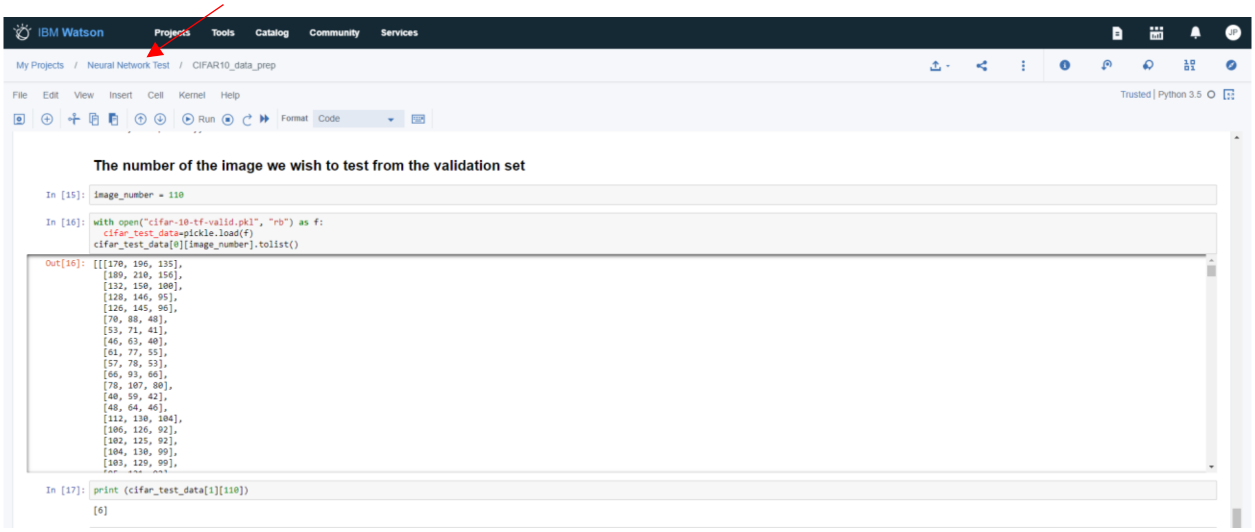


7. The ending cells are for getting data to test the deployed model. We'll come back to this.

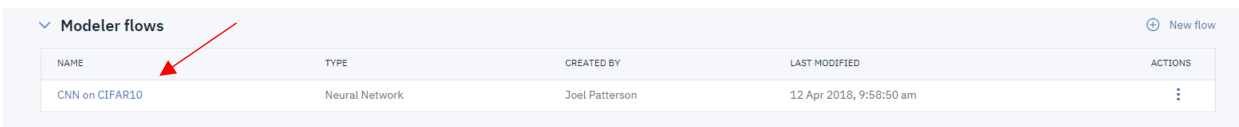


Step 3: Configuring the input data in the model

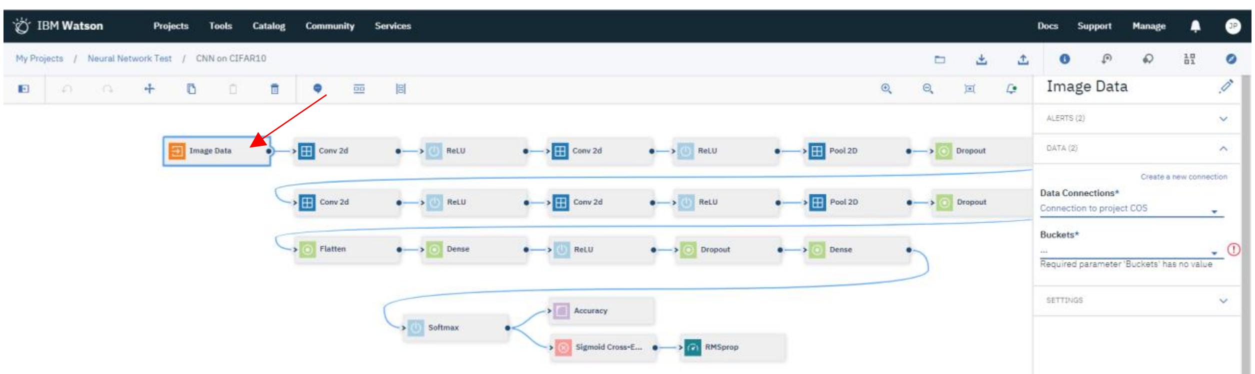
1. Return to the project by selecting the project name in the breadcrumb trail



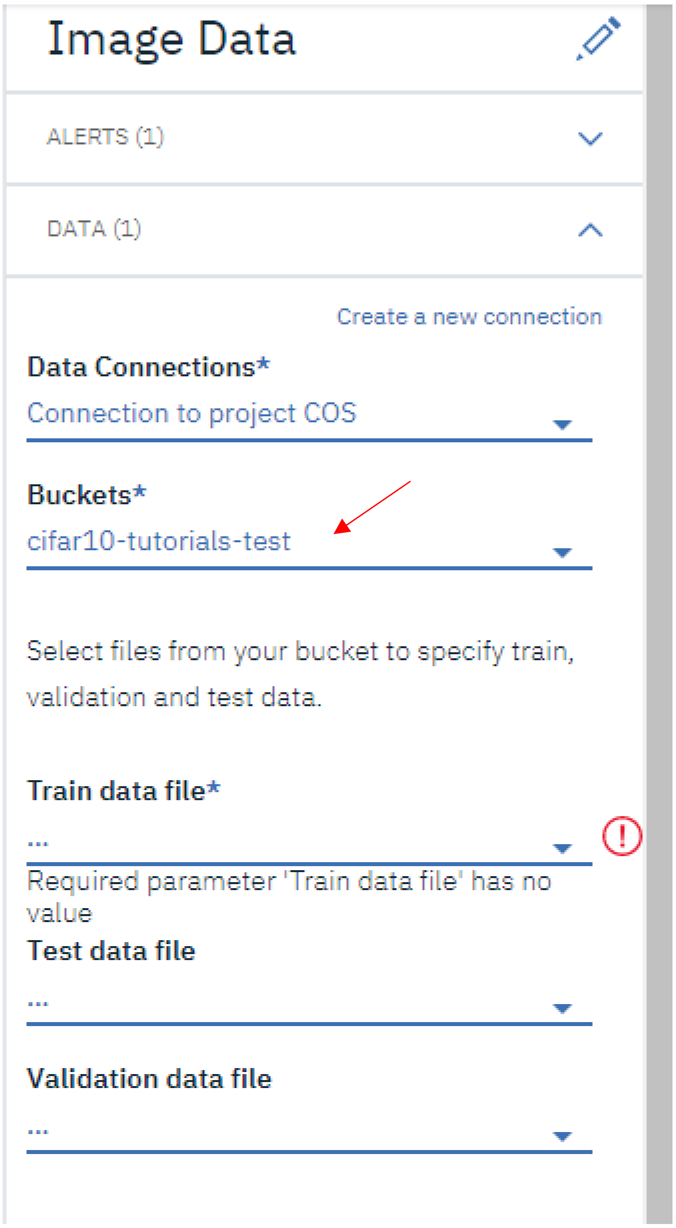
2. You should be in the **Assets** tab. Scroll down to **Modeler Flows** and open the **CNN** flow



3. Double-click the **Image Data** icon to open the configuration window.



4. Set **Buckets** to the value of the bucket you just created in the notebook.



5. Set the **Train, Test, Validate** files to the values in the bucket (you wrote these out in the notebook). Then **Save.**

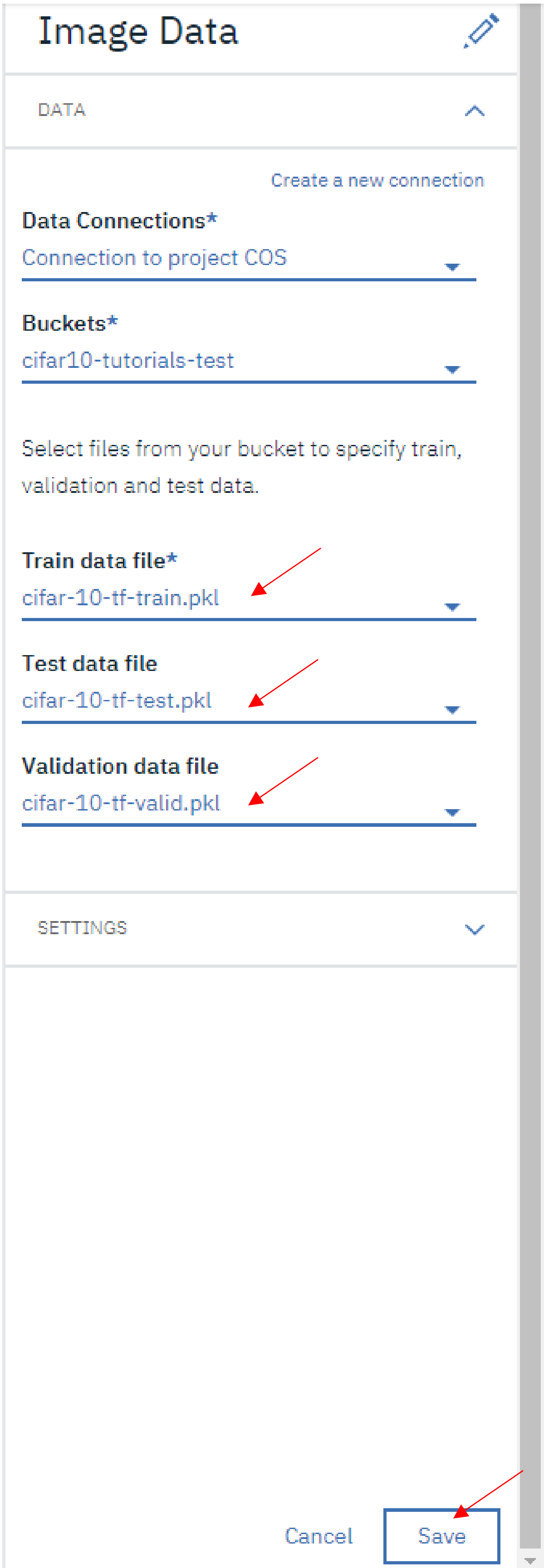
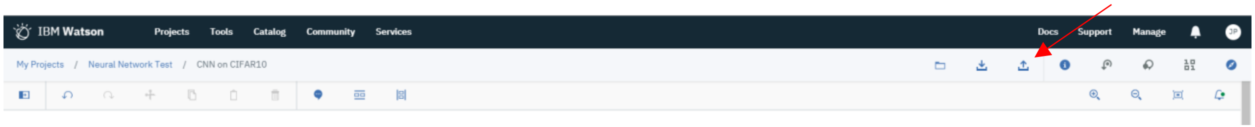
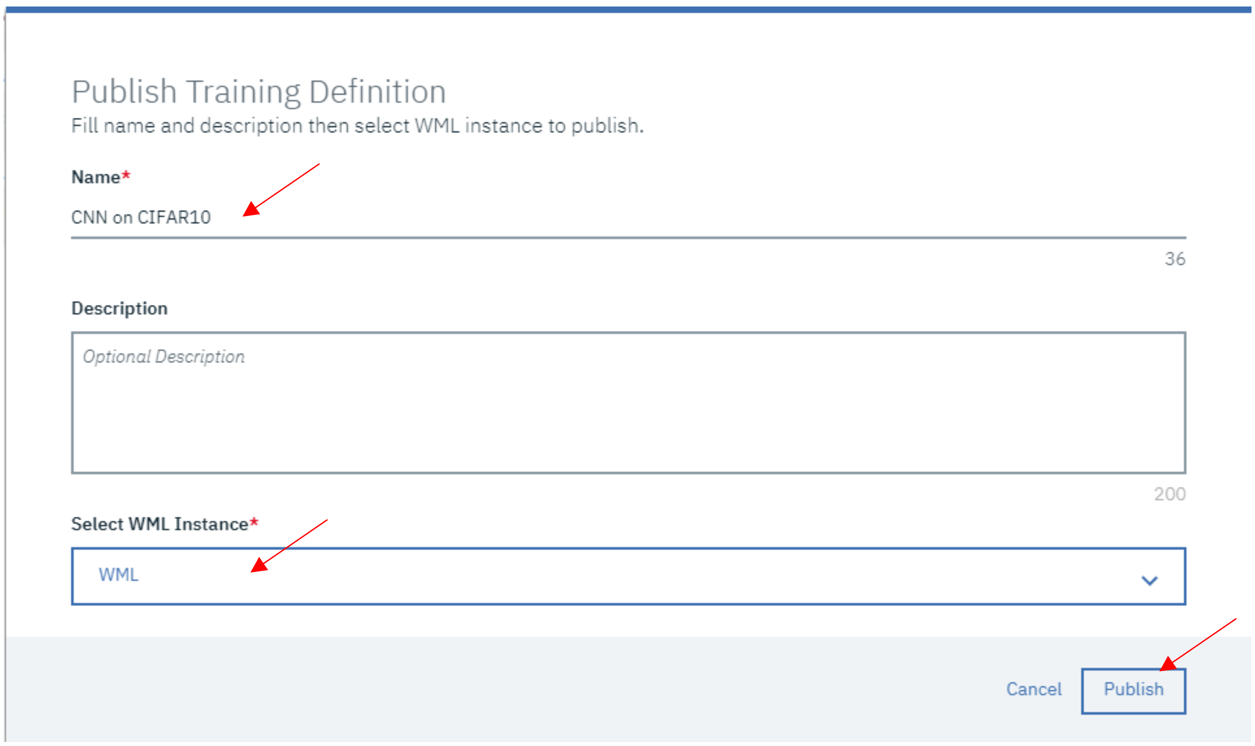


Image Data
DATA
Create a new connection
Data Connections*
Connection to project COS
Buckets*
cifar10-tutorials-test
Select files from your bucket to specify train, validation and test data.
Train data file*
cifar-10-tf-train.pkl
Test data file
cifar-10-tf-test.pkl
Validation data file
cifar-10-tf-valid.pkl
SETTINGS
Cancel
Save

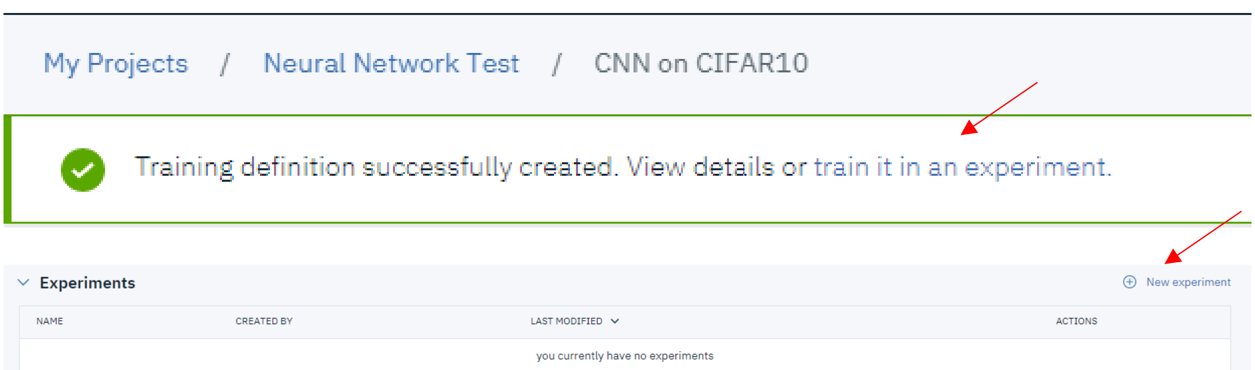
6. **Publish Training Definition** (container with up-pointing arrow)



7. Enter a **Name** for the definition and pick which WML Instance it should be published to (this is in case you have more than one defined). Select **Publish**.

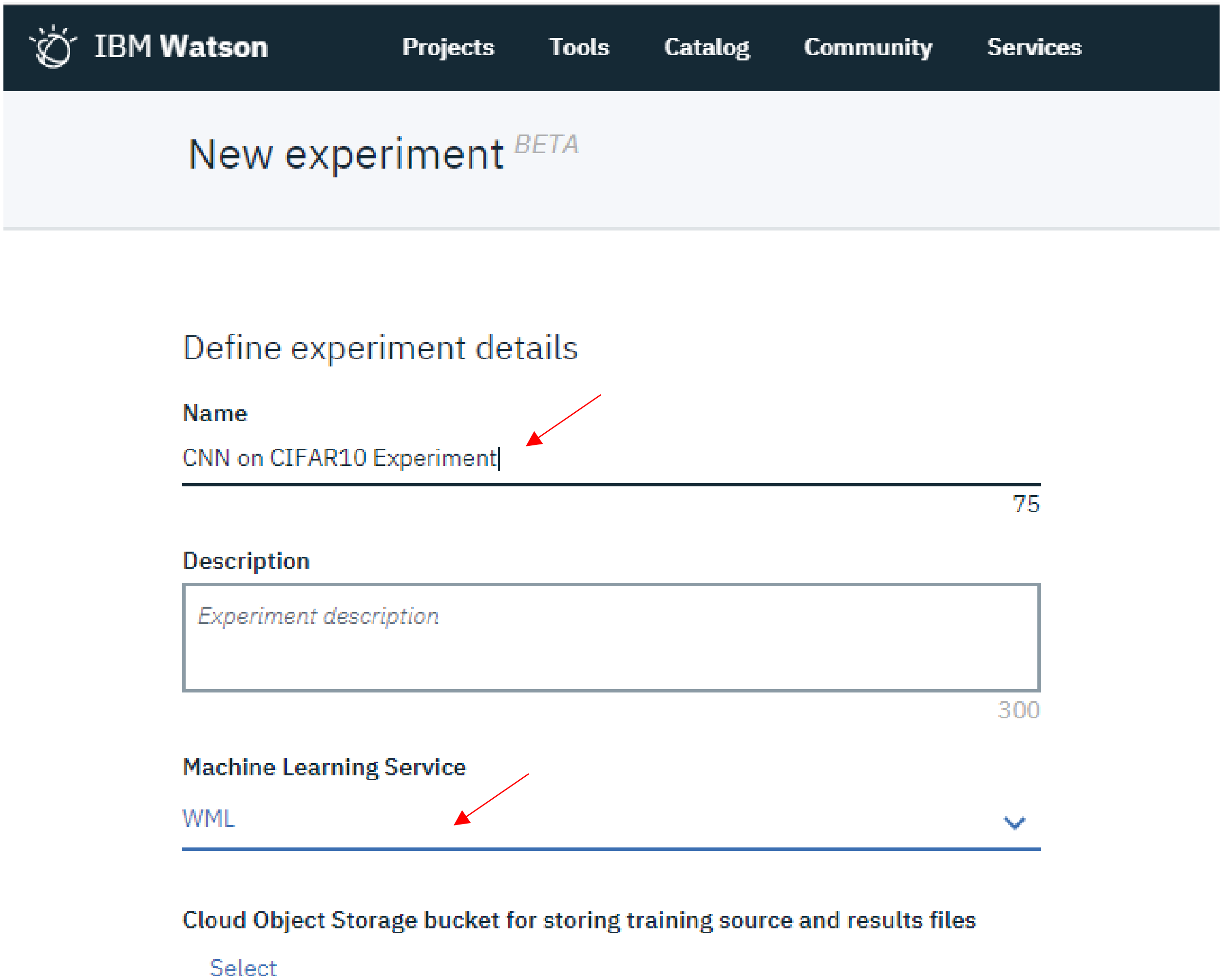


8. Upon successfully publication you have the option to train the definition in an experiment. You can select the quick link here or via the **Experiments** section in **Assets.**

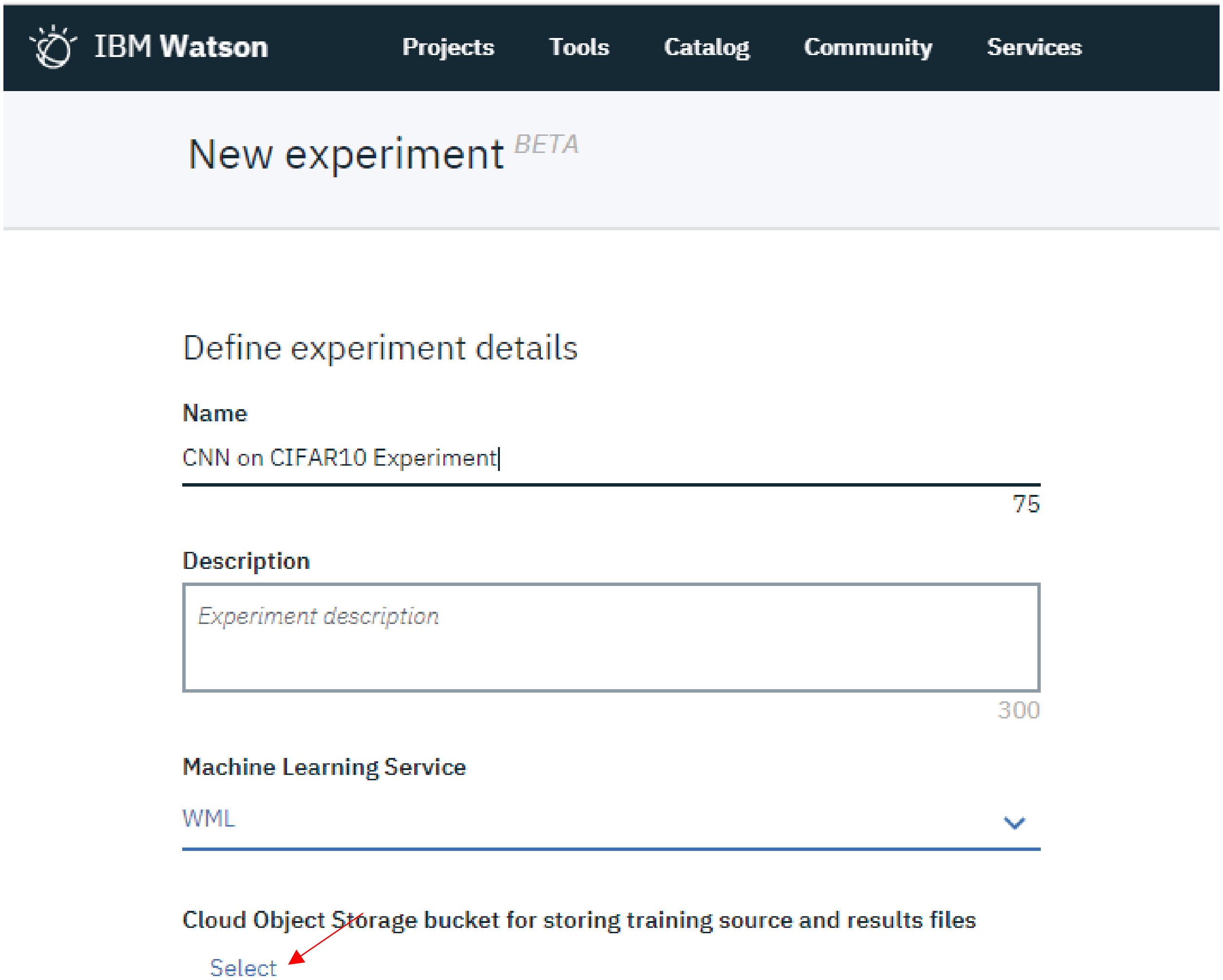


Step 4: Run an Experiment on the Training Definition

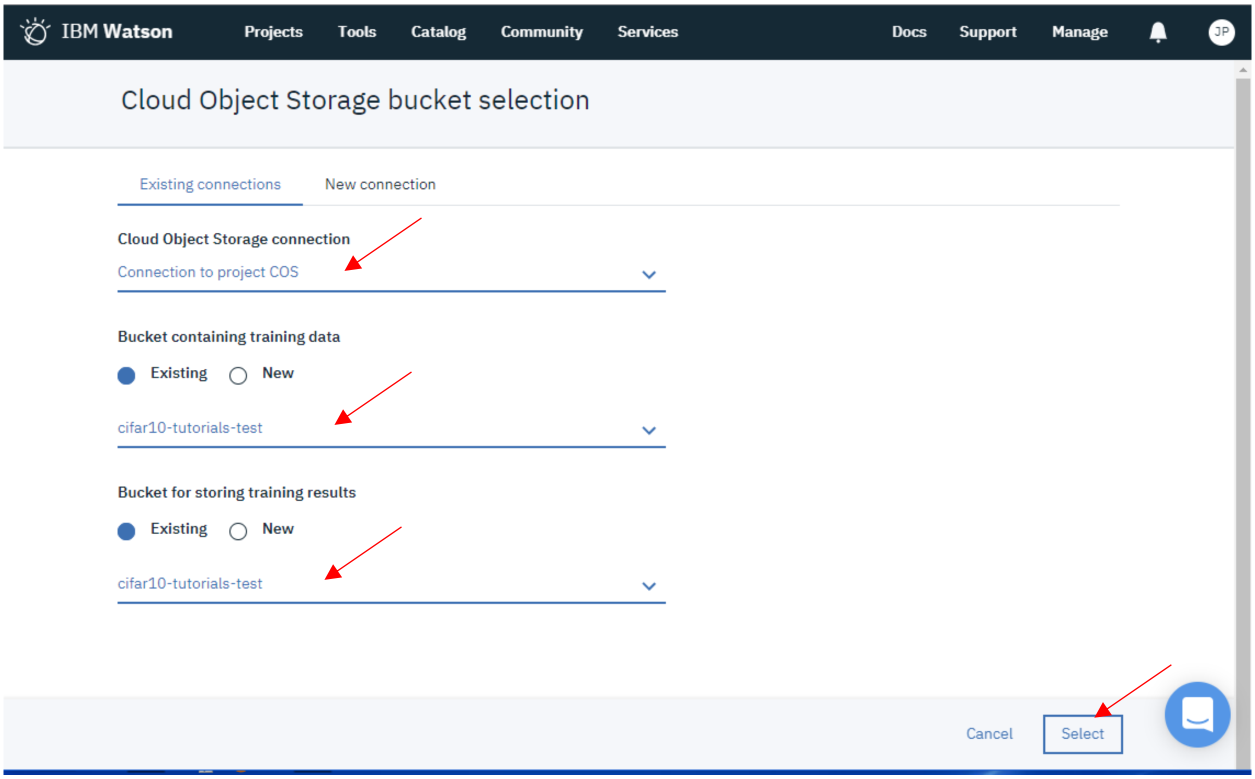
1. Give a **Name** to the Experiment and choose a **Machine Learning Service**



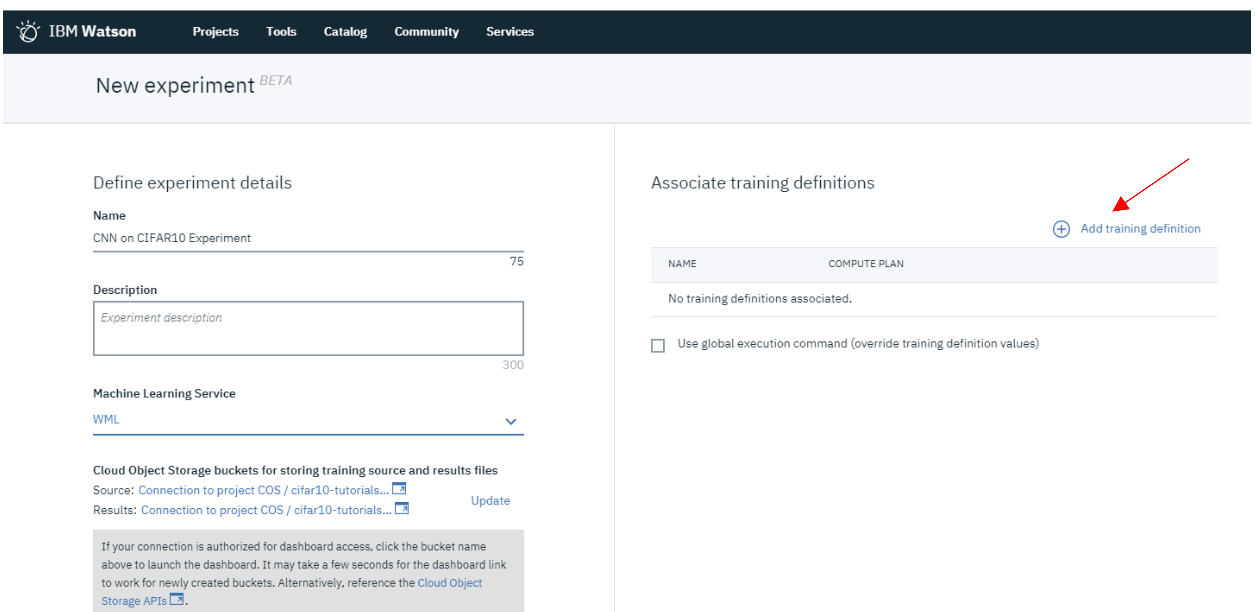
2. Click the **Select** link for **Cloud Object Storage bucket**



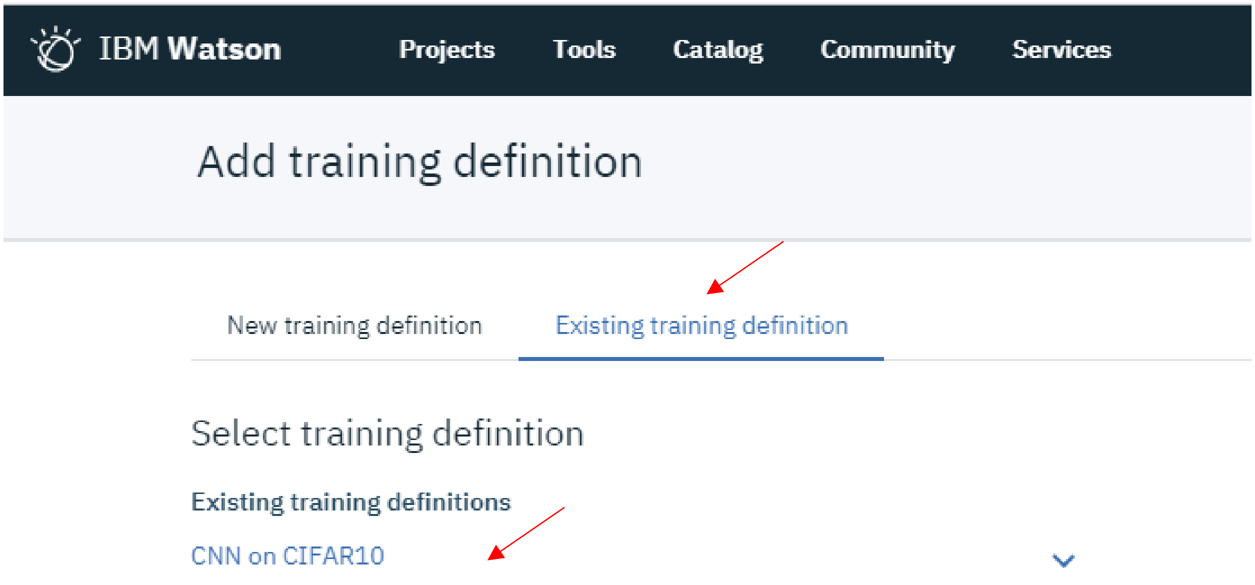
3. Using **Existing connections**, select your **Cloud Object Storage connection** and set both buckets to the bucket you created in the notebook. Click **Select.**



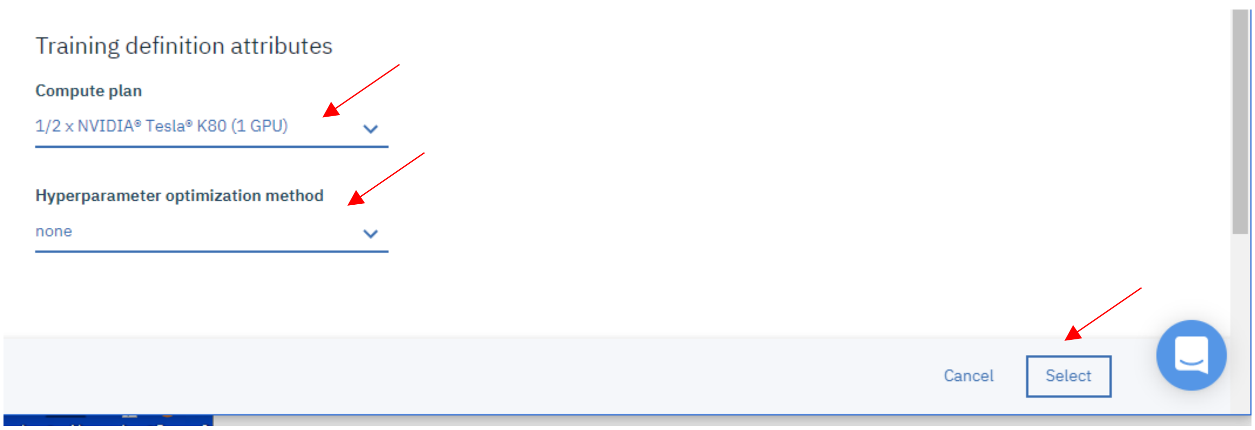
4. Select **Add training definition**



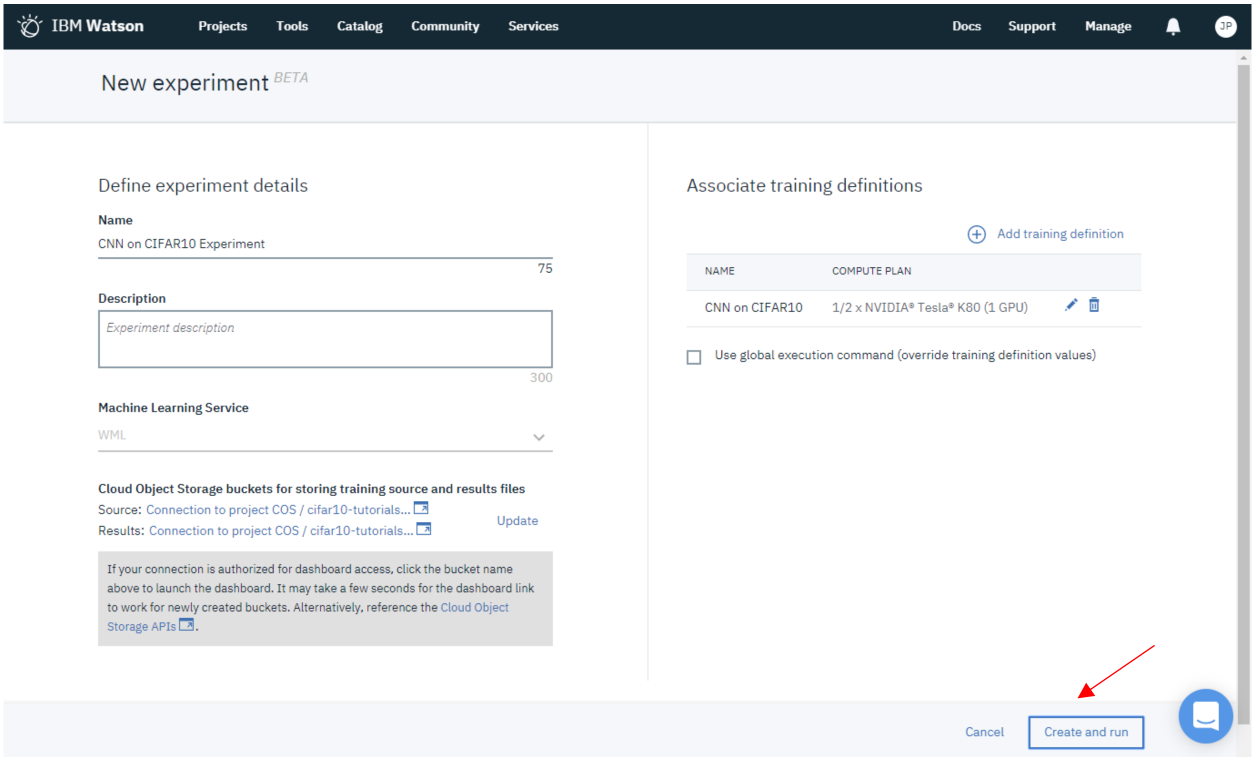
5. Select **Existing training definition** and choose the definition you saved.



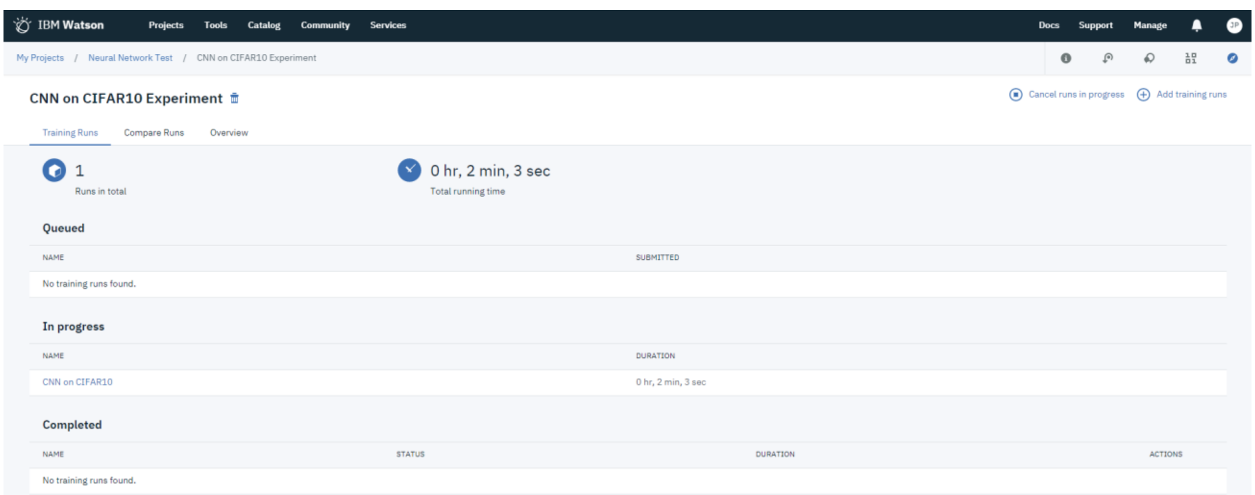
6. Select **Training definition attributes**. **Compute plan** should be ½ x NVIDIA Tesla K80 (1 GPU) – free accounts are only allowed this selection. **Hyperparameter optimization method** can be any value, but **none** will run the quickest. Click **Select.**

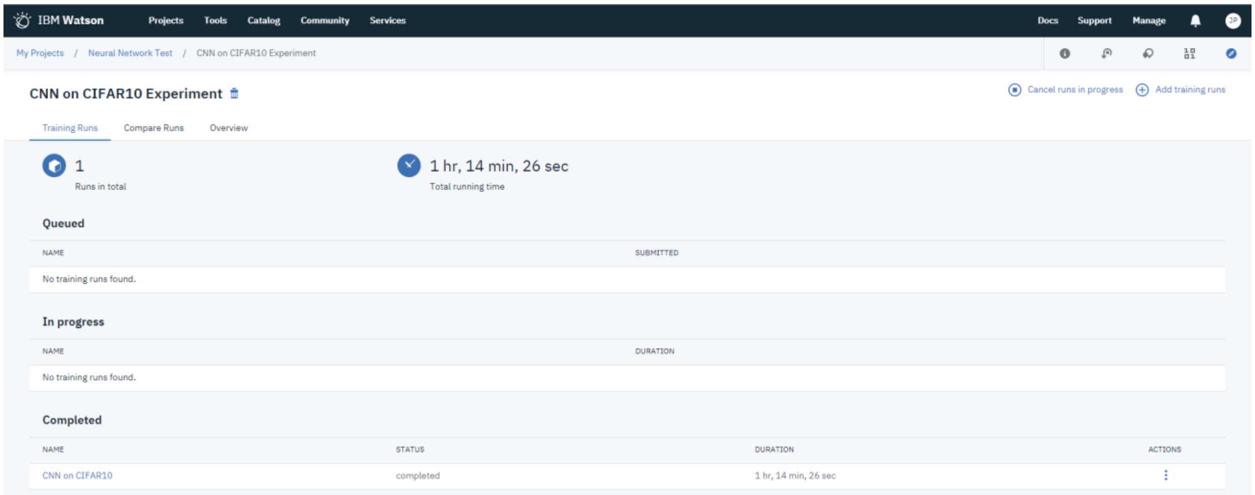


7. Select **Create and run**



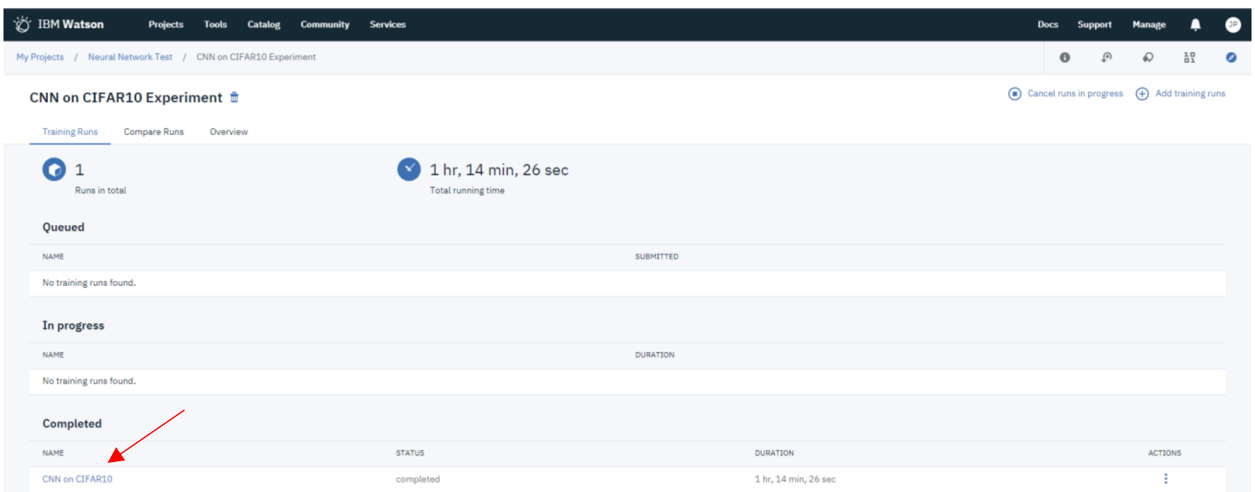
8. Training neural networks is resource intensive. Therefore, this experiment will take an hour or so to complete. Start working on the next lab and we will come back to this later.



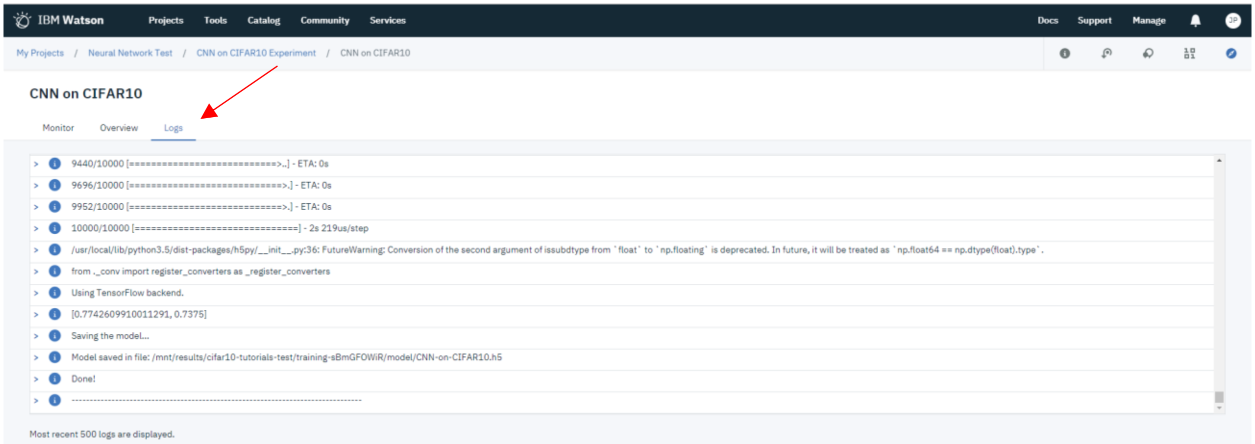


Step 5: Save and deploy the Model

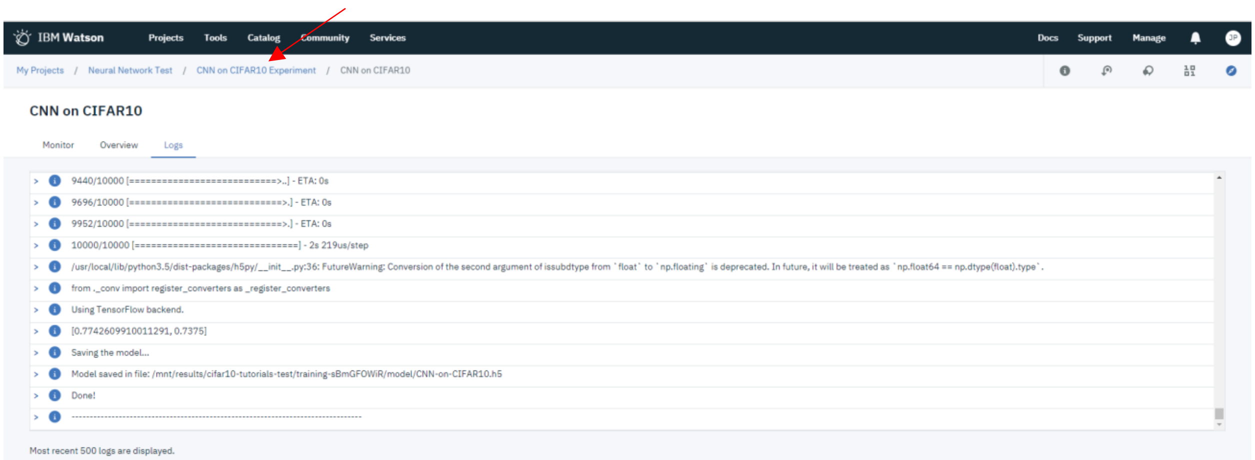
1. Select the completed training run



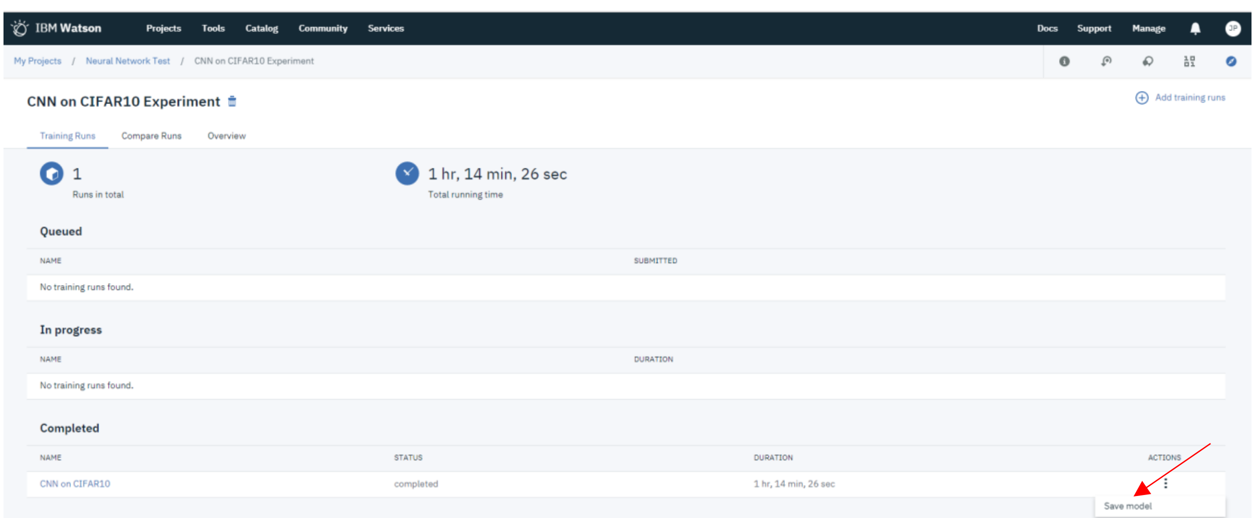
2. View Overview and Logs



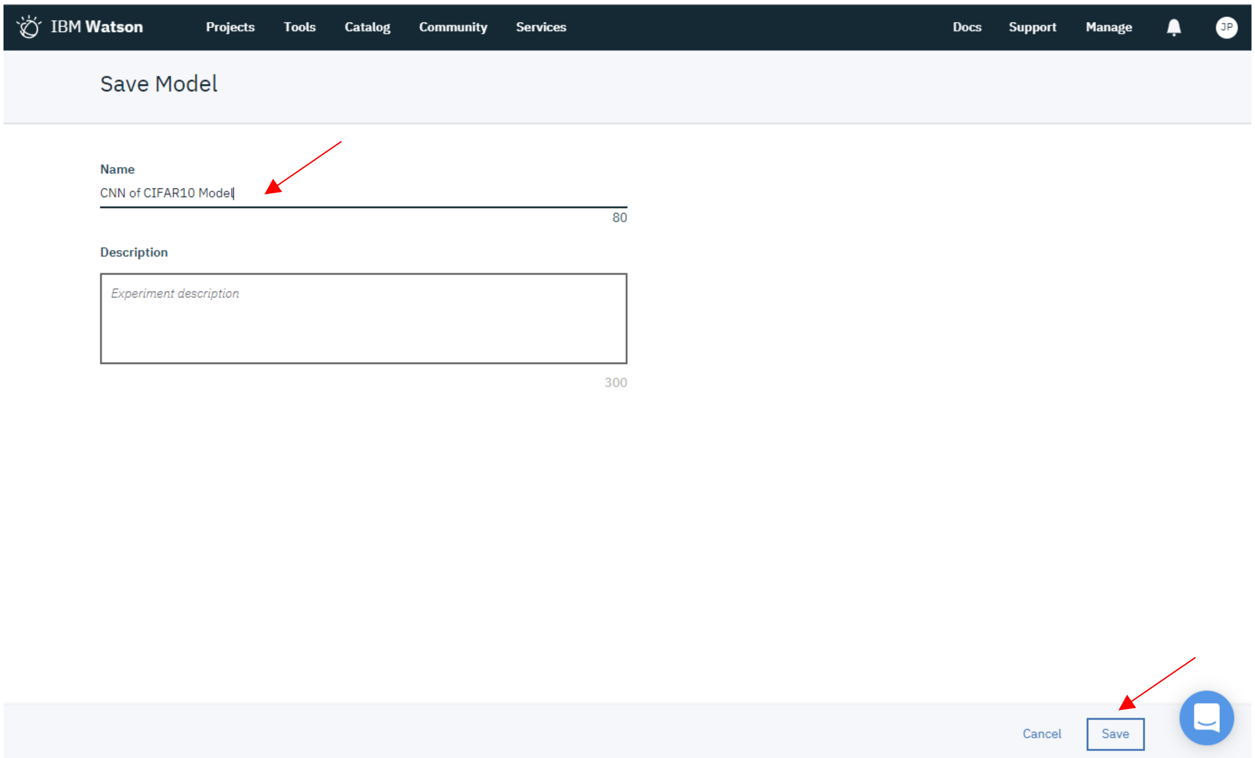
3. Return to the Experiment by selecting it in the breadcrumb trail



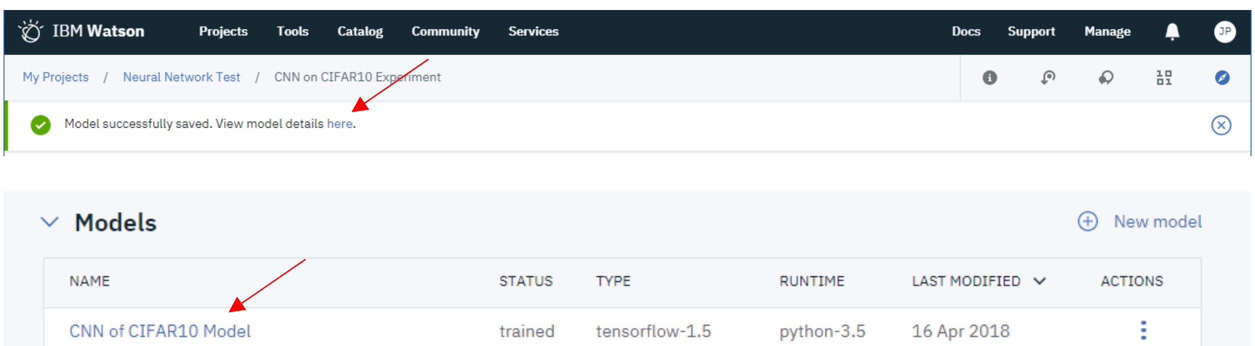
4. Select the **Actions** for the completed Experiment and **Save model**



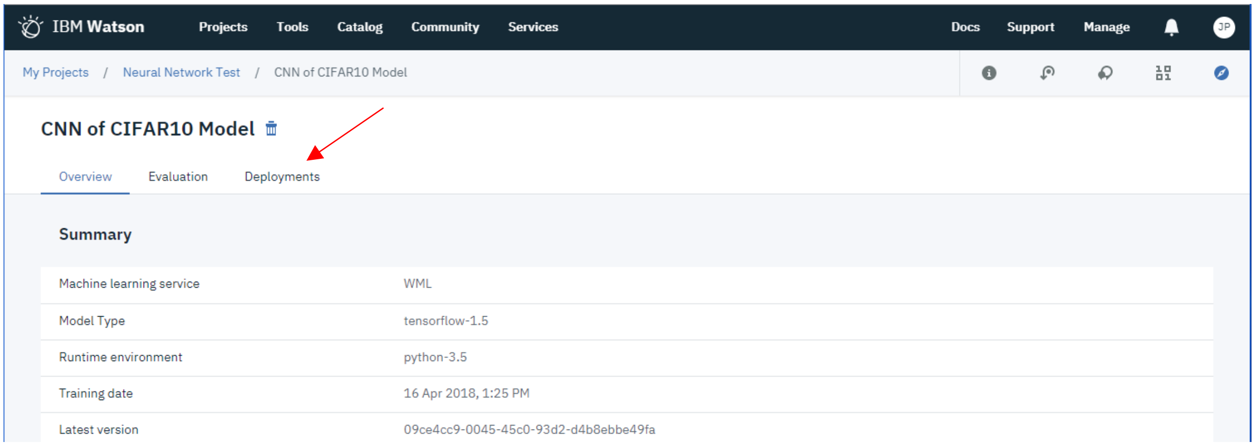
5. Give a **Name** to the model and select **Save**



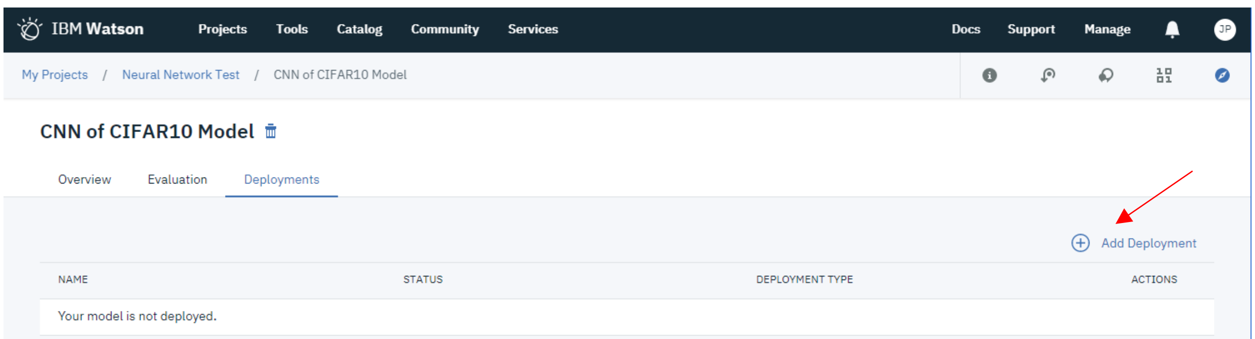
6. Upon successful save you can view the model by selecting the link or by viewing the model in the **Assets** tab.



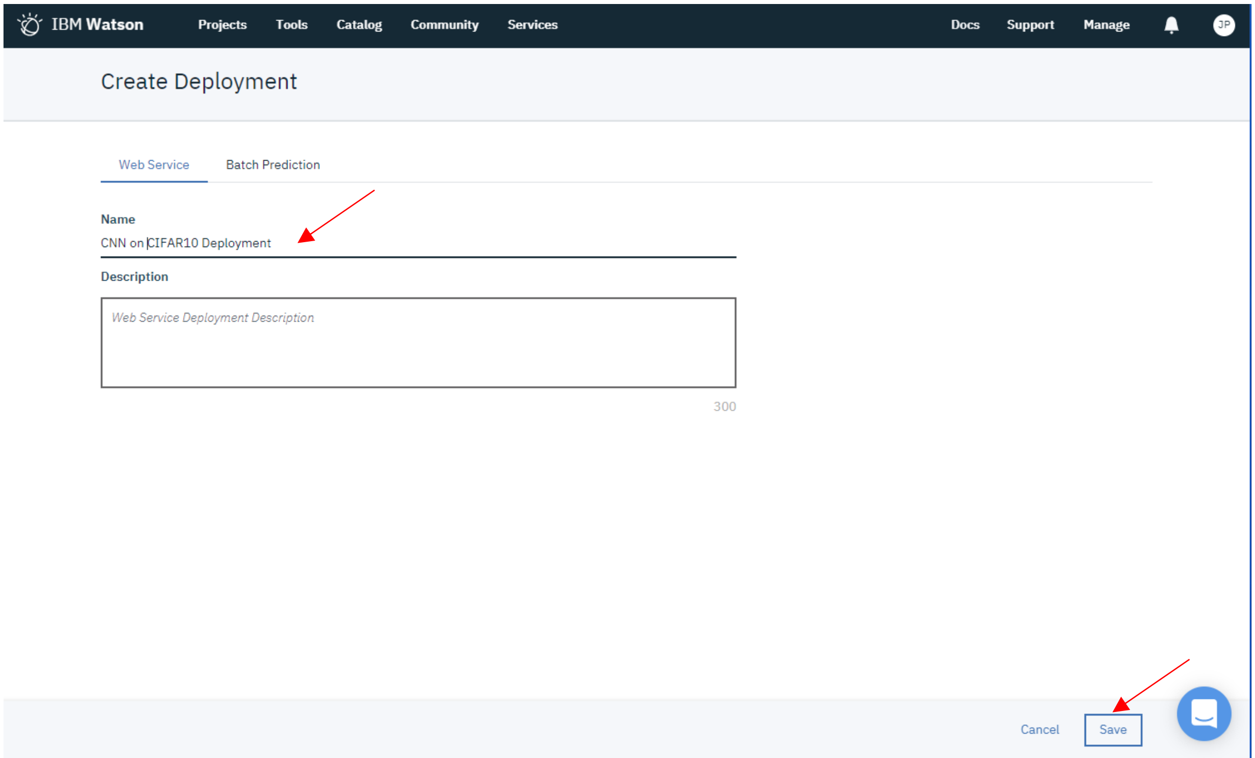
7. Select the **Deployments** tab



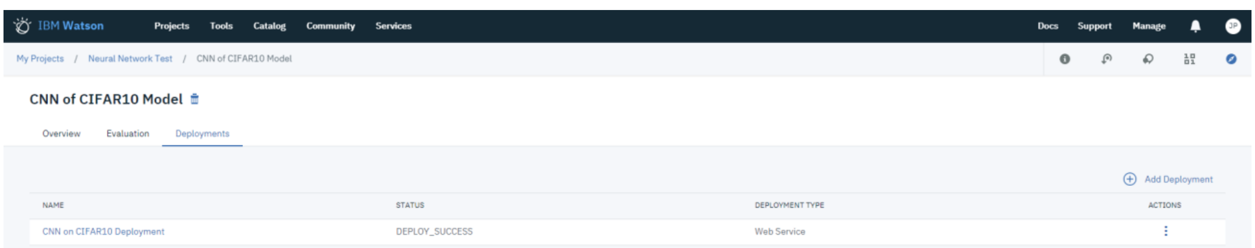
8. Select **Add deployment**



9. Give a **Name** to the deployment and select **Save**

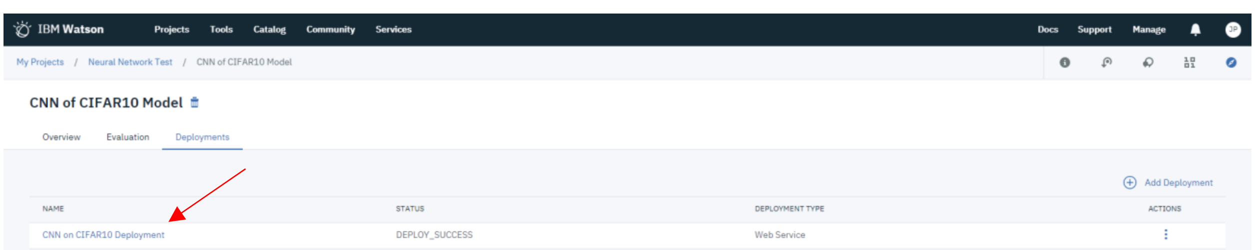


10. Wait until deployment is successful (refreshing the page may help here)

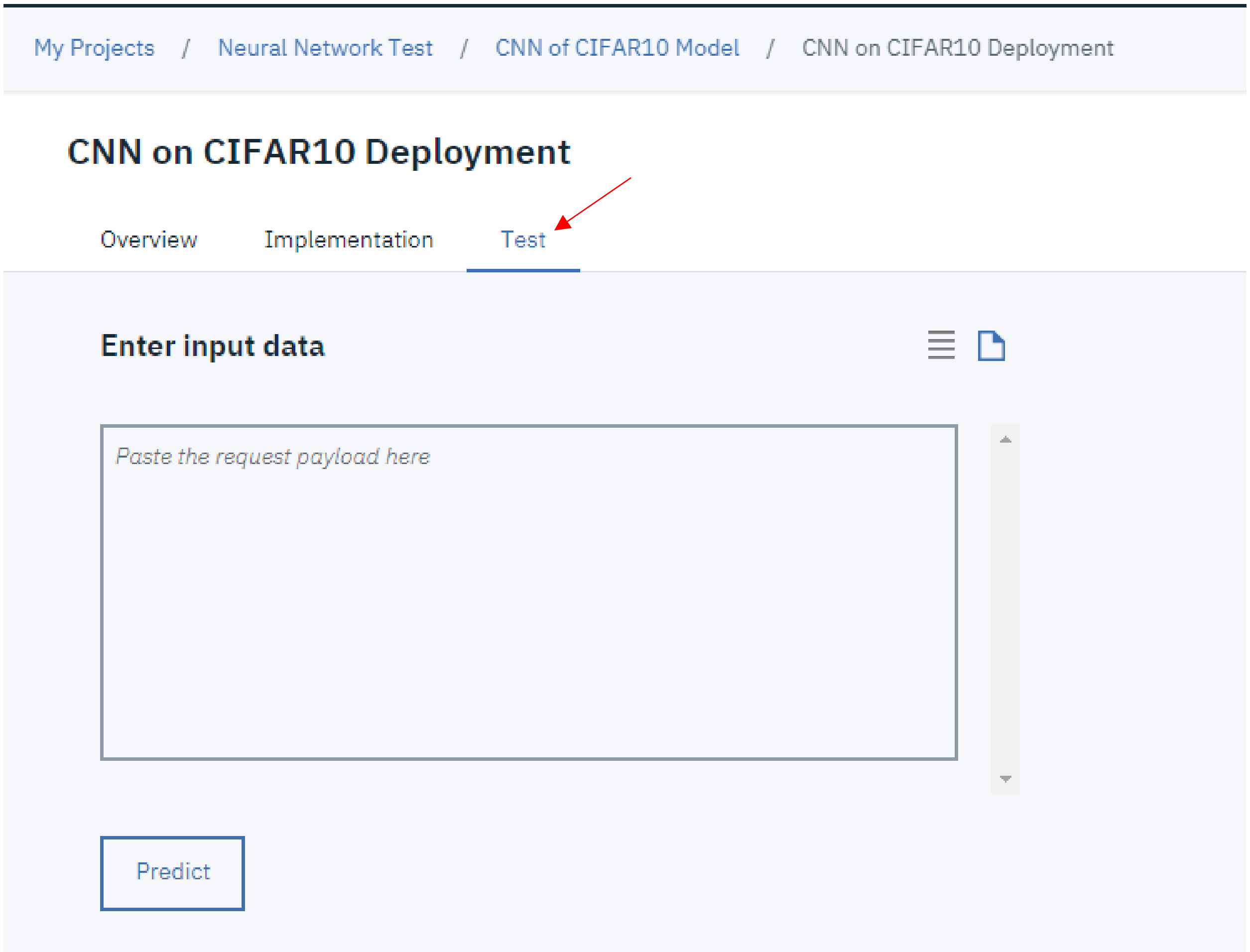


Step 6: Test the Model

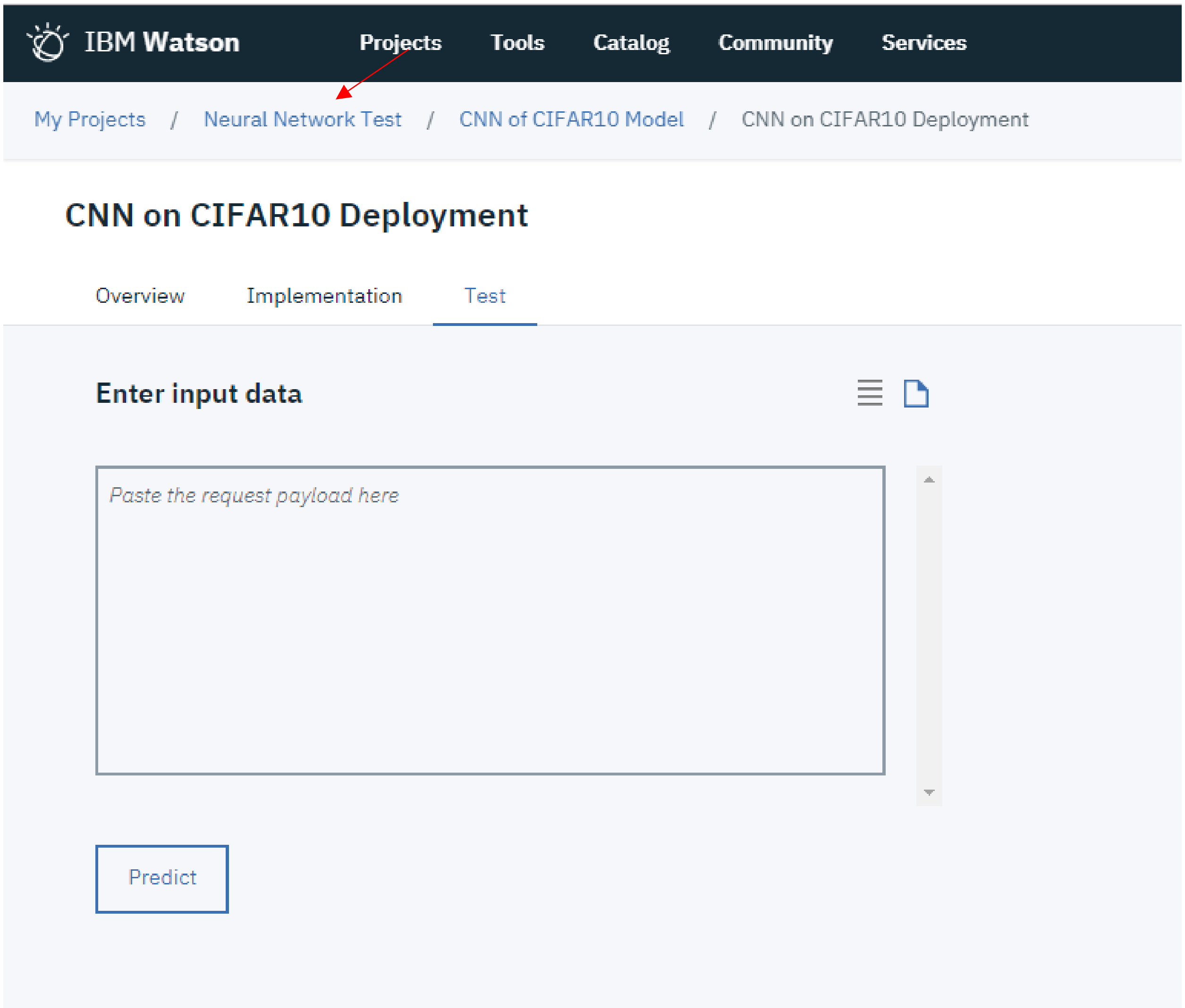
1. Select the deployed model



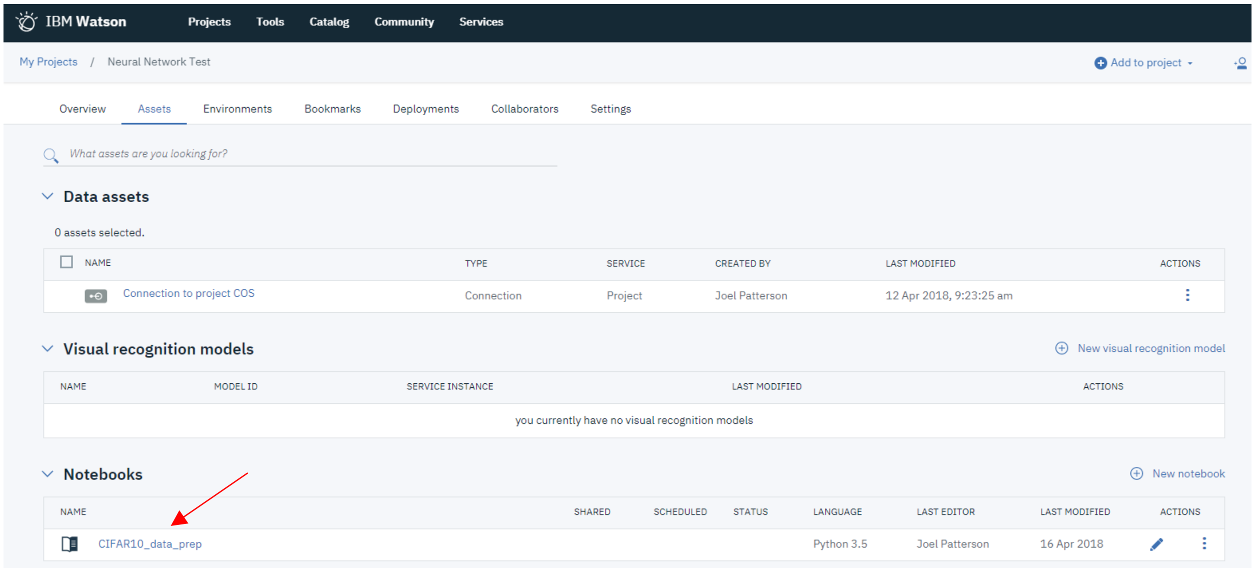
2. Select **Test** to test the model



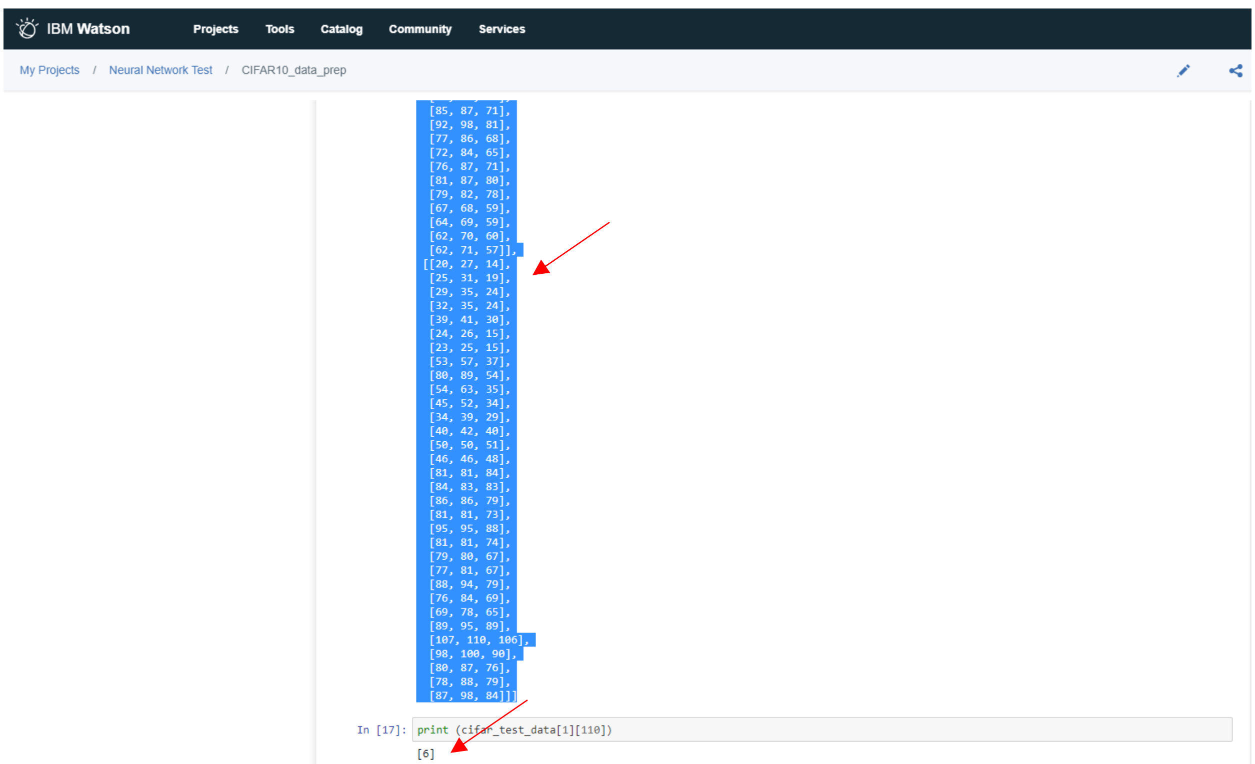
3. Right-click on the project breadcrumb and **Open in new tab**



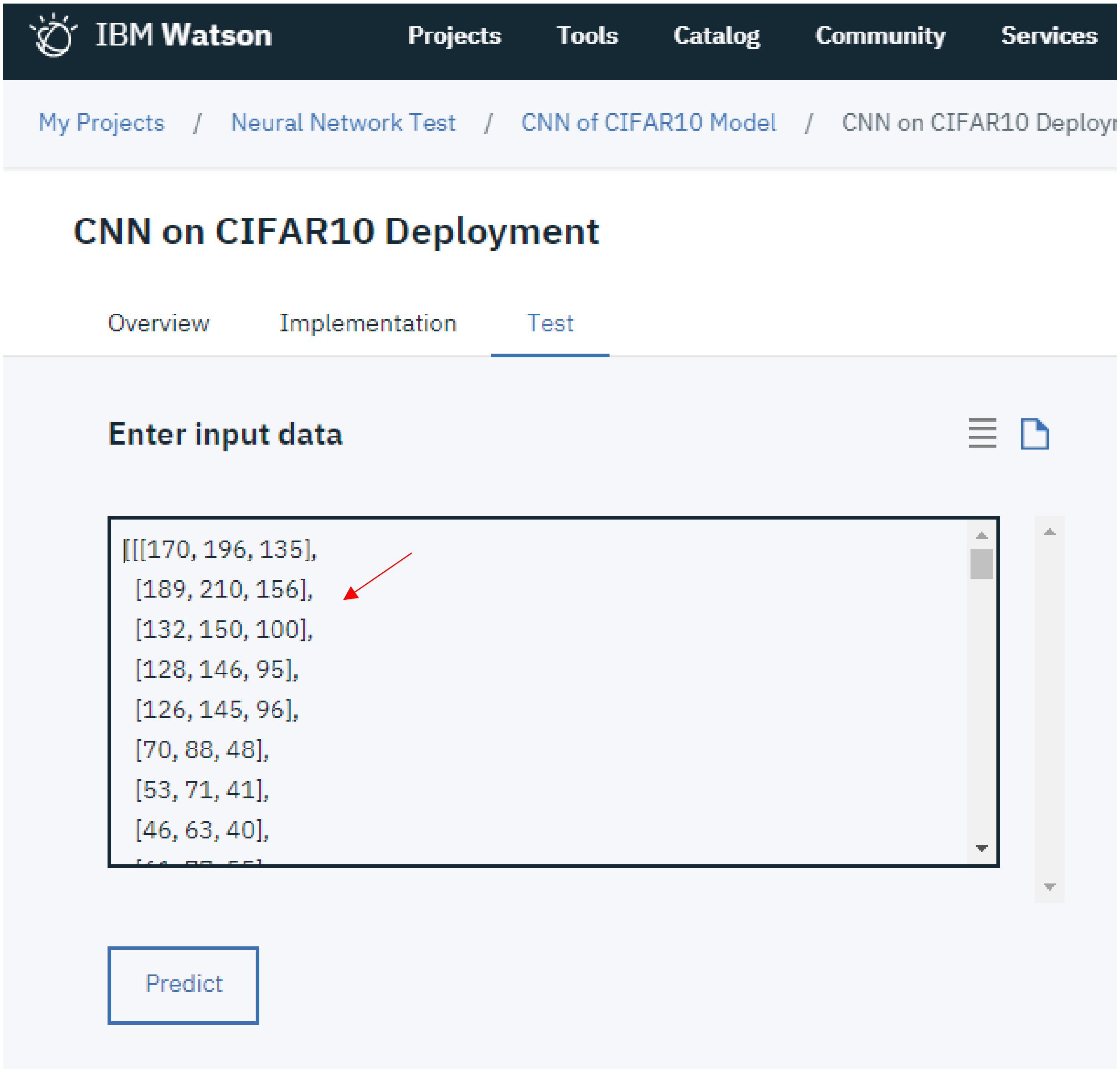
4. Open the CIFAR10 notebook you created earlier.



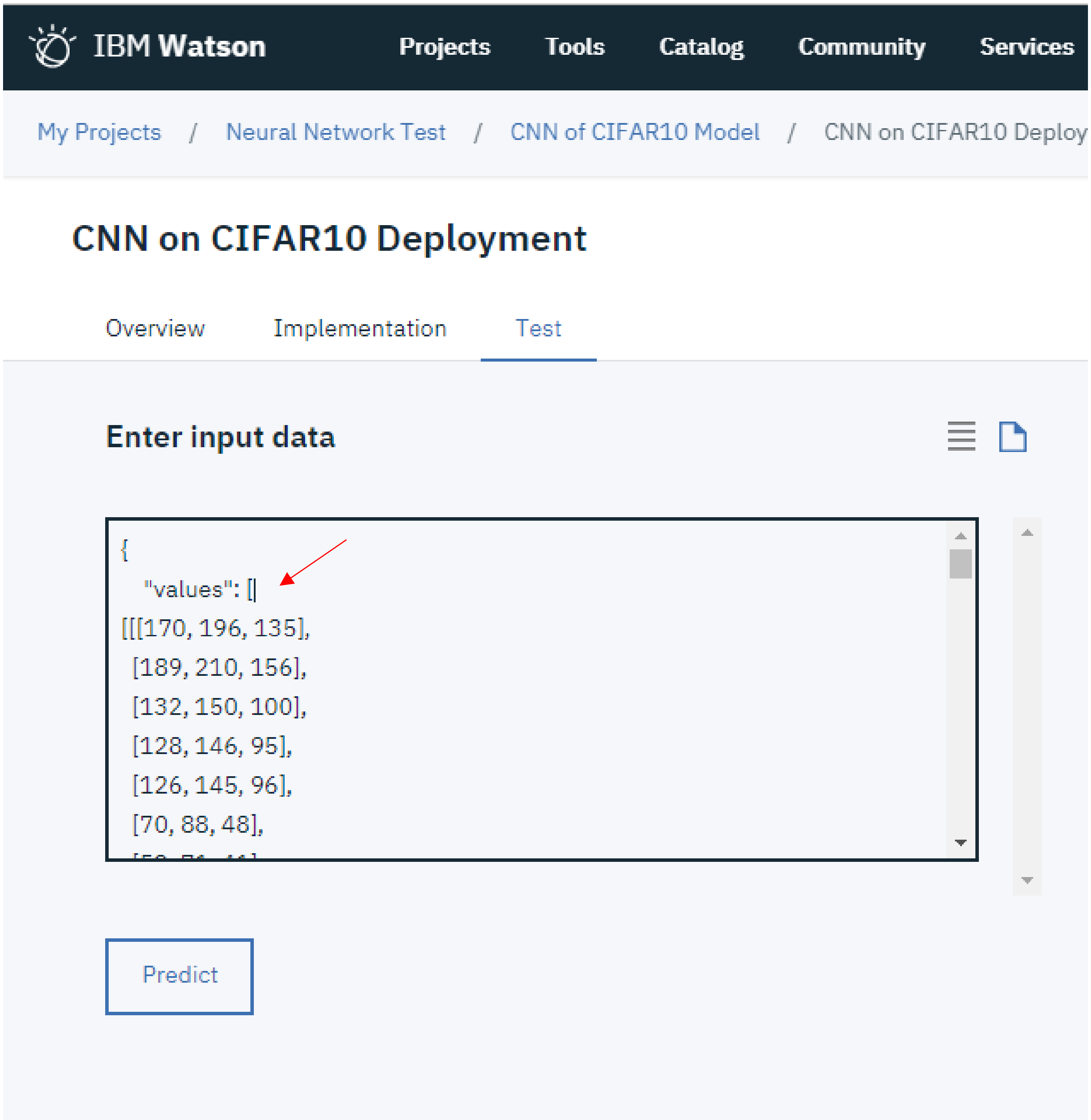
5. Copy the image data at the end of the notebook. Also take note of the value at the end.



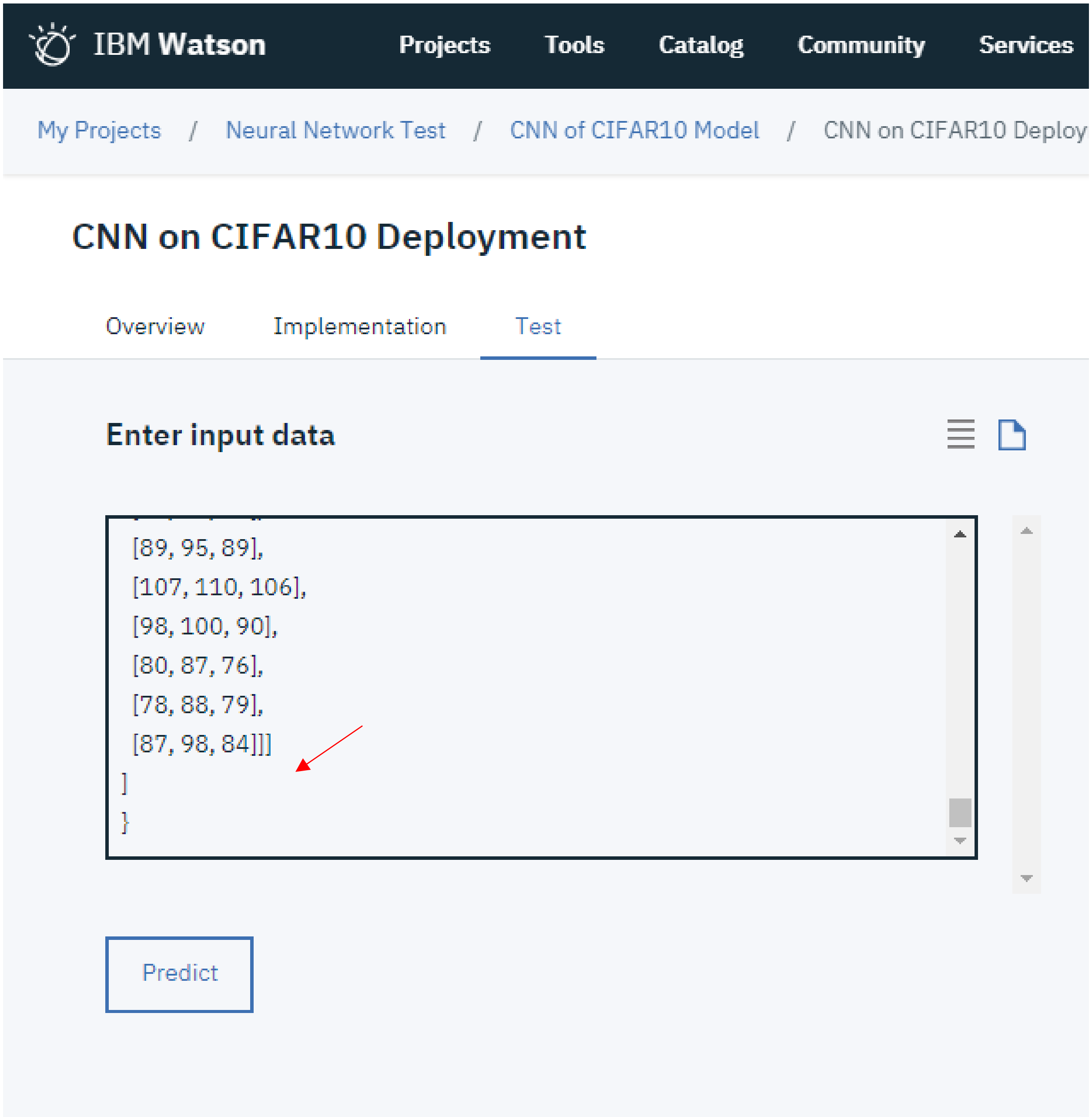
6. Paste this into the test window.



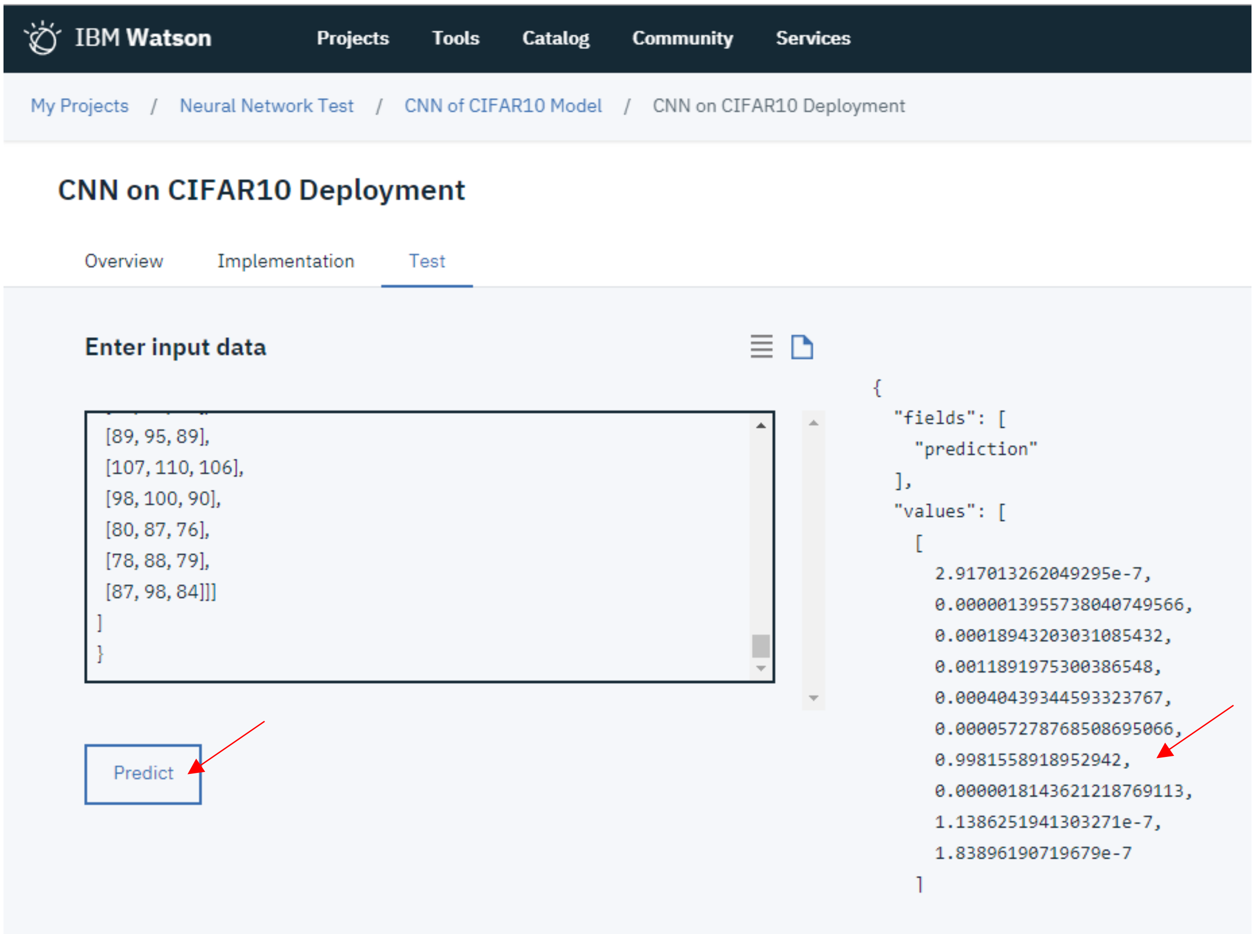
7. Add { “values”: [to the top



8. Add] } to the bottom



9. **Predict.** You should get an array of 10 values [0-9 scale]. The maximum value should map to the index value you saw in the notebook.


IBM Watson
Projects
Tools
Catalog
Community
Services
My Projects / Neural Network Test / CNN of CIFAR10 Model / CNN on CIFAR10 Deployment
CNN on CIFAR10 Deployment
Overview
Implementation
Test
Enter input data
[89, 95, 89],
[107, 110, 106],
[98, 100, 90],
[80, 87, 76],
[78, 88, 79],
[87, 98, 84]]]
]
}
Predict
{
"fields": [
"prediction"
],
"values": [
[
2.917013262049295e-7,
0.0000013955738040749566,
0.00018943203031085432,
0.0011891975300386548,
0.00040439344593323767,
0.00005727876850869506 6,
0.9981558918952942,
0.0000018143621218769113,
1.1386251941303271e-7,
1.83896190719679e-7
]