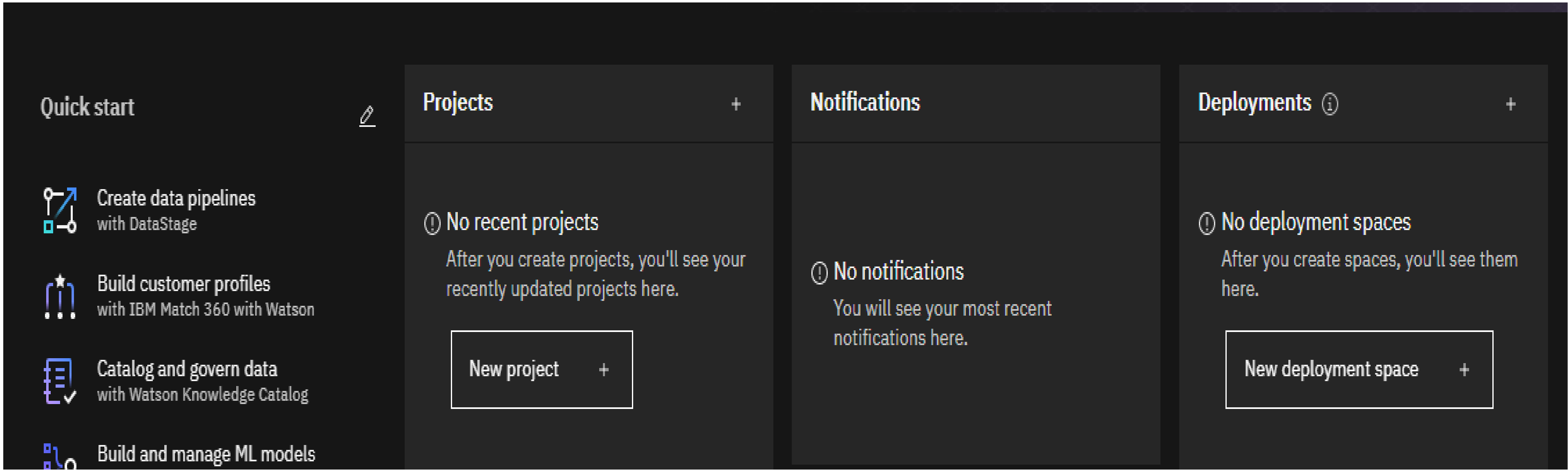
Train on IBM Cloud: Cloud deployment



Quick start

Create data pipelines
with DataStage

Build customer profiles
with IBM Match 360 with Watson

Catalog and govern data
with Watson Knowledge Catalog

Build and manage ML models

Projects +

ⓘ No recent projects

After you create projects, you'll see your recently updated projects here.

New project +

Notifications

ⓘ No notifications

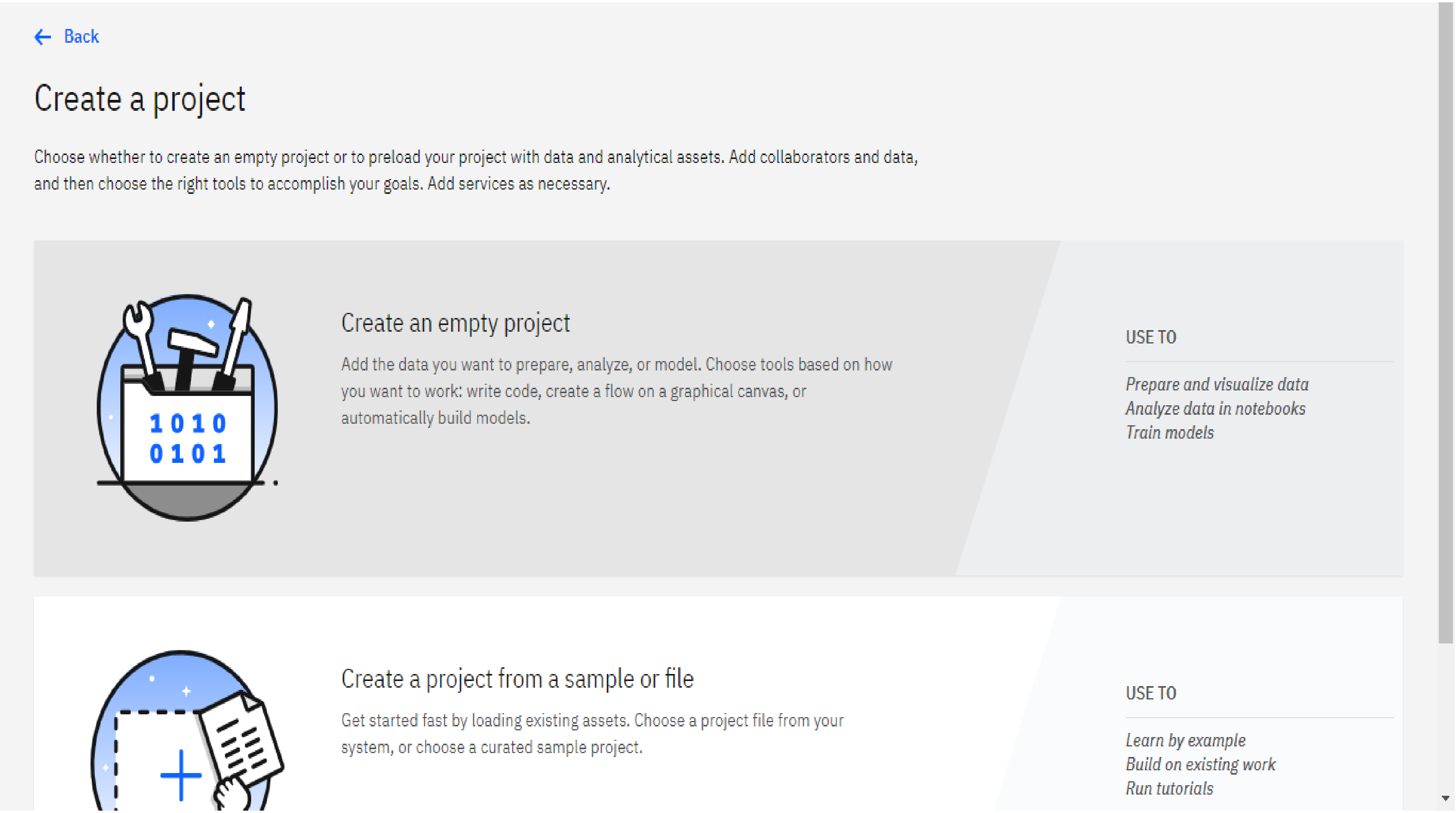
You will see your most recent notifications here.

Deployments ⓘ +

ⓘ No deployment spaces

After you create spaces, you'll see them here.

New deployment space +



← 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

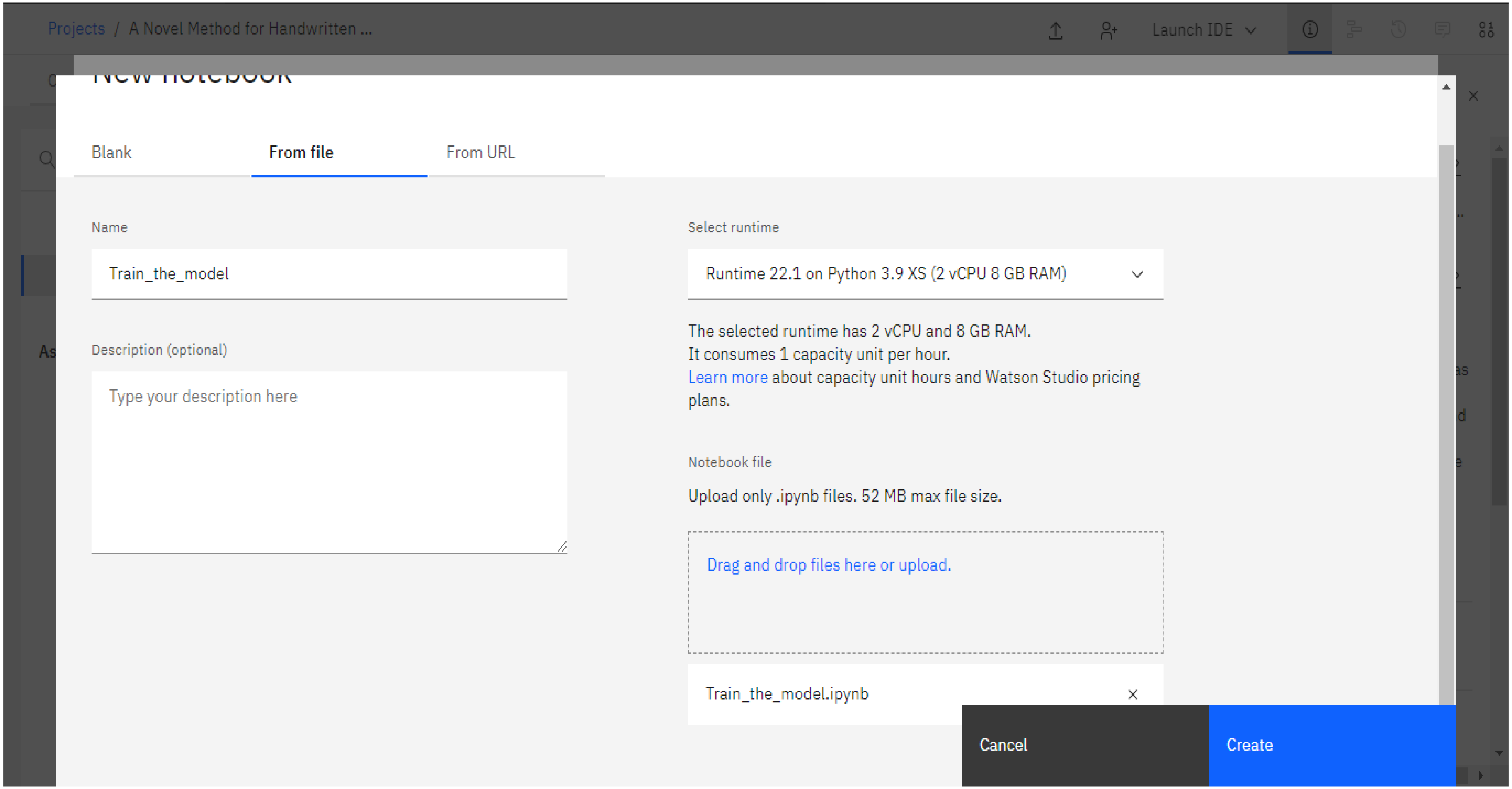
Prepare and visualize data
Analyze data in notebooks
Train models

Create a project from a sample or file

Get started fast by loading existing assets. Choose a project file from your system, or choose a curated sample project.

USE TO

Learn by example
Build on existing work
Run tutorials



Name: Train_the_model

Description (optional): Type your description here

Select runtime: Runtime 22.1 on Python 3.9 XS (2 vCPU 8 GB RAM)

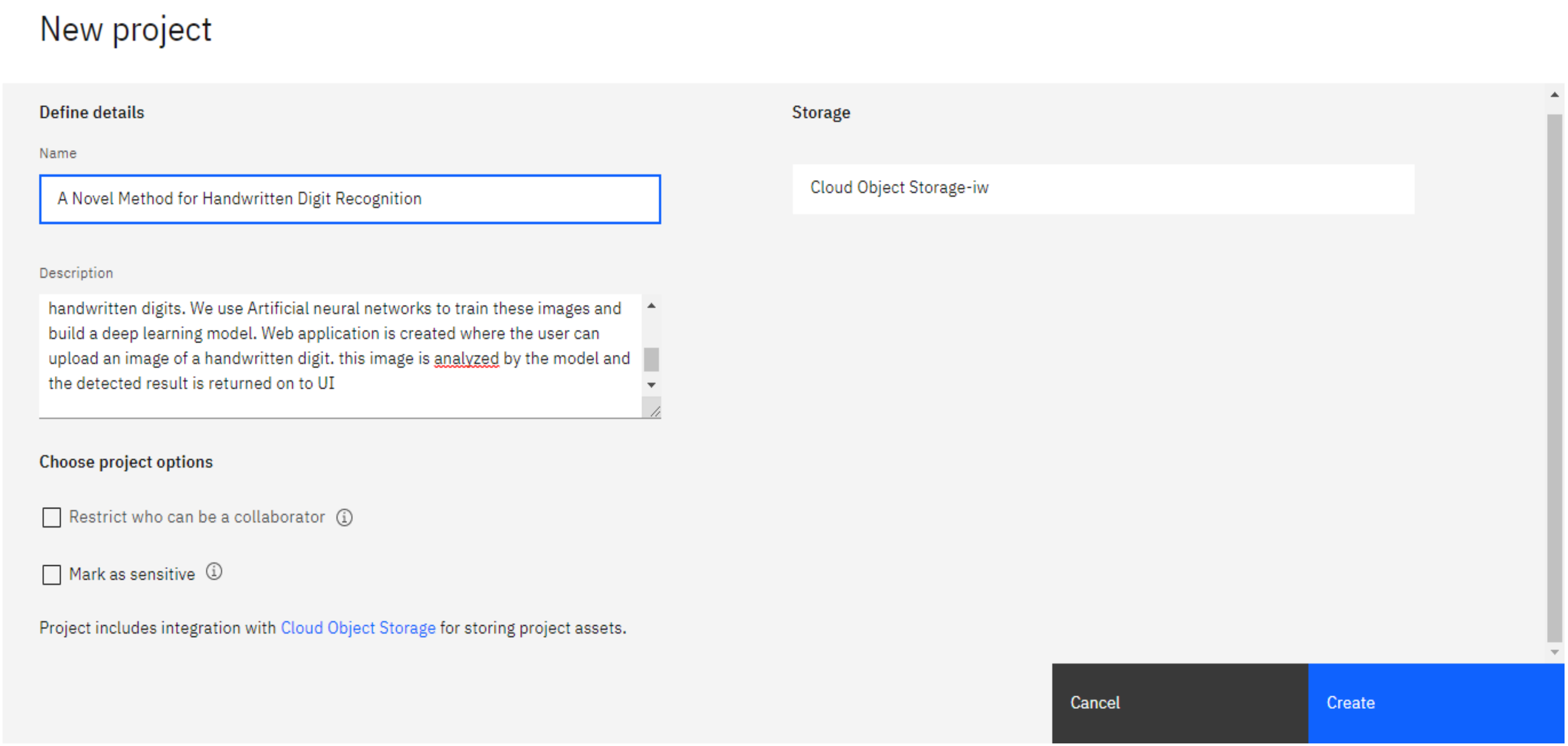
The selected runtime has 2 vCPU and 8 GB RAM.
It consumes 1 capacity unit per hour.
Learn more about capacity unit hours and Watson Studio pricing plans.

Notebook file

Upload only .ipynb files. 52 MB max file size.

Drag and drop files here or upload.

Train_the_model.ipynb ×



New project

Define details

Name

A Novel Method for Handwritten Digit Recognition

Description

handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI

Choose project options

□ Restrict who can be a collaborator

□ Mark as sensitive

Project includes integration with Cloud Object Storage for storing project assets.

Storage

Cloud Object Storage-iw

IBM cloud deployment

Team ID - PNT2022TMID44334

Importing the required libraries

```
!pip install tensorflow --upgrade
```

```
Requirement already satisfied: tensorflow in /opt/conda/envs/Python-3.9/lib/python3.9
Collecting tensorflow
  Downloading tensorflow-2.10.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.
    |████████████████████████████████| 578.1 MB 40 kB/s /s eta 0:00:01B 15.2 MB/s et
Requirement already satisfied: termcolor>=1.1.0 in /opt/conda/envs/Python-3.9/lib/pyt
Requirement already satisfied: flatbuffers>=2.0 in /opt/conda/envs/Python-3.9/lib/pyt
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /opt/conda/envs/Python-3.9/lib/
Requirement already satisfied: keras-preprocessing>=1.1.1 in /opt/conda/envs/Python-3
Requirement already satisfied: numpy>=1.20 in /opt/conda/envs/Python-3.9/lib/python3.
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /opt/conda/envs/Python-3.9/lib/
Collecting absl-py>=1.0.0
  Downloading absl_py-1.3.0-py3-none-any.whl (124 kB)
    |████████████████████████████████| 124 kB 83.1 MB/s eta 0:00:01
Requirement already satisfied: astunparse>=1.6.0 in /opt/conda/envs/Python-3.9/lib/py
Requirement already satisfied: six>=1.12.0 in /opt/conda/envs/Python-3.9/lib/python3.
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /opt/conda/env
Collecting keras<2.11,>=2.10.0
  Downloading keras-2.10.0-py2.py3-none-any.whl (1.7 MB)
    |████████████████████████████████| 1.7 MB 88.2 MB/s eta 0:00:01
Collecting libclang>=13.0.0
  Downloading libclang-14.0.6-py2.py3-none-manylinux2010_x86_64.whl (14.1 MB)
    |████████████████████████████████| 14.1 MB 78.8 MB/s eta 0:00:01
Collecting tensorboard<2.11,>=2.10
  Downloading tensorboard-2.10.1-py3-none-any.whl (5.9 MB)
    |████████████████████████████████| 5.9 MB 69.7 MB/s eta 0:00:01
Requirement already satisfied: opt-einsum>=2.3.2 in /opt/conda/envs/Python-3.9/lib/py
Requirement already satisfied: setuptools in /opt/conda/envs/Python-3.9/lib/python3.9
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/
Requirement already satisfied: typing-extensions>=3.6.6 in /opt/conda/envs/Python-3.9
Requirement already satisfied: google-pasta>=0.1.1 in /opt/conda/envs/Python-3.9/lib/
Requirement already satisfied: protobuf<3.20,>=3.9.2 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: h5py>=2.9.0 in /opt/conda/envs/Python-3.9/lib/python3.
Collecting tensorflow-estimator<2.11,>=2.10.0
  Downloading tensorflow_estimator-2.10.0-py2.py3-none-any.whl (438 kB)
    |████████████████████████████████| 438 kB 83.2 MB/s eta 0:00:01
Requirement already satisfied: wrapt>=1.11.0 in /opt/conda/envs/Python-3.9/lib/python
Requirement already satisfied: wheel<1.0,>=0.23.0 in /opt/conda/envs/Python-3.9/lib/p
Requirement already satisfied: google-auth<3,>=1.6.3 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /opt/conda/envs/Py
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /opt/conda/en
Requirement already satisfied: requests<3,>=2.21.0 in /opt/conda/envs/Python-3.9/lib/
Requirement already satisfied: markdown>=2.6.8 in /opt/conda/envs/Python-3.9/lib/pyth
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /opt/conda/envs/Pytho
Requirement already satisfied: werkzeug>=1.0.1 in /opt/conda/envs/Python-3.9/lib/pyth
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /opt/conda/envs/Python-3.9/l
Requirement already satisfied: rsa<5,>=3.1.4 in /opt/conda/envs/Python-3.9/lib/python
Requirement already satisfied: pyasn1-modules>=0.2.1 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: requests-oauthlib>=0.7.0 in /opt/conda/envs/Python-3.9
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /opt/conda/envs/Python-3.9/lib
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Python-3.9/lib/p
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/envs/Python-3.9/lib/pyth
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9
Installing collected packages: absl-py, tensorflow-estimator, tensorboard, libclang,
  Attempting uninstall: absl-py
```

```
import numpy as np
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.datasets import mnist #mnist dataset
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A Layer consists of a tensor- in tensor-out computat ion
from tensorflow.keras.layers import Dense, Flatten #Dense-Dense Layer is the regular deeply c
#faltten -used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D #convolutional Layer
from keras.utils import np_utils #used for one-hot encoding
import matplotlib.pyplot as plt   #used for data visualization
```

Load data

```
(x_train, y_train), (x_test, y_test)=mnist.load_data () #splitting the mnist data into train
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
```

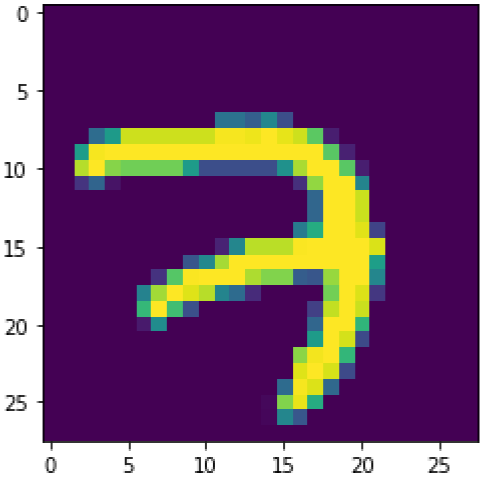
```
print (x_train.shape)  #shape is used for give the dimens ion values #60000-rows 28x28-pixels
print (x_test.shape)
```

```
(60000, 28, 28)
(10000, 28, 28)
```

```
x_train[0]
```

```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
  18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170,
 253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253,
 253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253,
 253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253,
 205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,
  90,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 139, 253,
 190,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  11, 190,
 253,  70,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  35,
 241, 225, 160, 108,   1,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
  81, 240, 253, 253, 119,  25,   0,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,  45, 186, 253, 253, 150,  27,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,  16,  93, 252, 253, 187,   0,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0, 249, 253, 249,  64,   0,   0,   0,   0,   0,
   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,  46, 130, 183, 253, 253, 207,   2,   0,   0,   0,   0,   0,
   0,   0],
plt.imshow(x_train[5000])     #ploting the index=image
```

```
<matplotlib.image.AxesImage at 0x7fb302596d00>
```



```
np.argmax(y_train[5000])
```

```
0
```

Reshaping Dataset

```
#Reshaping to format which CNN expects (batch, height, width, channels)
x_train=x_train.reshape (60000, 28, 28, 1).astype('float32')
x_test=x_test.reshape (10000, 28, 28, 1).astype ('float32')
```

Applying One Hot Encoding

```
number_of_classes = 10  #storing the no of classes in a variable
```

```
y_train = np_utils.to_categorical (y_train, number_of_classes) #converts the output in binary
y_test = np_utils.to_categorical (y_test, number_of_classes)
```

Add CNN Layers

```
#create model
model=Sequential ()
```

```
#adding modeL Layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation = 'relu'))


#flatten the dimension of the image
model.add(Flatten())


#output layer with 10 neurons
model.add(Dense(number_of_classes,activation = 'softmax'))
```

Compiling the model

```
#Compile model
model.compile(loss= 'categorical_crossentropy', optimizer="Adam", metrics=['accuracy'])
```

```
x_train = np.asarray(x_train)
y_train = np.asarray(y_train)
```

Train the model

```
#fit the model
model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5, batch_size=32)
```

```
Epoch 1/5
1875/1875 [==============================] - 107s 57ms/step - loss: 0.2935 - accuracy: 0
Epoch 2/5
1875/1875 [==============================] - 106s 57ms/step - loss: 0.0787 - accuracy: 0
Epoch 3/5
1875/1875 [==============================] - 107s 57ms/step - loss: 0.0533 - accuracy: 0
Epoch 4/5
1875/1875 [==============================] - 107s 57ms/step - loss: 0.0409 - accuracy: 0
Epoch 5/5
1875/1875 [==============================] - 106s 56ms/step - loss: 0.0333 - accuracy: 0
<keras.callbacks.History at 0x7fb3023f6820>
```

Observing the metrics

```
# Final evaluation of the model
metrics = model.evaluate(x_test, y_test, verbose=0)
```

```
print("Metrics (Test loss &Test Accuracy) : ")
print(metrics)
```

```
Metrics (Test loss &Test Accuracy) :
[0.12009724229574203, 0.9739000201225281]
```

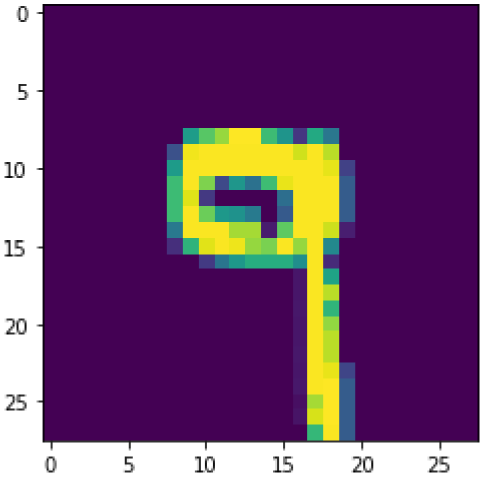
Test The Model

```
prediction=model.predict(x_test[6000:6001])
print(prediction)
```

```
[[1.3093596e-15 1.6657296e-18 1.0302369e-14 6.9182353e-09 3.4993657e-06
  2.2733215e-09 5.9939021e-18 2.8354838e-08 8.0210708e-07 9.9999559e-01]]
```

```
plt.imshow(x_test[6000])
```

```
<matplotlib.image.AxesImage at 0x7fb300207dc0>
```



```
import numpy as np
print(np.argmax(prediction, axis=1)) #printing our Labels from first 4 images
```

```
[9]
```

```
np.argmax(y_test[6000:6001]) #printing the actual labels
```

```
9
```

Save The model

```
# Save the model
model.save('models/mnistCNN.h5')
```

```
cd models
```

```
/home/wsuser/work/models
```

```
!tar -zcvf handwritten-digit-recognition-model_new.tgz mnistCNN.h5
```

```
mnistCNN.h5
```

```
!pip install watson-machine-learning-client --upgrade
```

```
Requirement already satisfied: watson-machine-learning-client in /opt/conda/envs/Python-
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-p
Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-p
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/s
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-pa
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-pac
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/l
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/li
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/l
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9
```

Cloud deploy

```
from ibm_watson_machine_learning import APIClient
credentials ={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"oHvops4PuUNbcBbzCyzOWKSt2bKQVD0bj2EAlZIvrGdv"
}
client = APIClient(credentials)
```

```
<ibm_watson_machine_learning.client.APIClient at 0x7fb2ecbdbc40>
```

```
client = APIClient(credentials)
```

```
def guid_from_space_name(client,deploy):
  space = client.spaces.get_details()
  return (next(item for item in space['resources'] if item['entity']['name']==deploy)['metada

space_uid = guid_from_space_name(client,'handwritten digit recognition')
print("Space UID = " + space_uid)
```

```
Space UID = 0a096f80-7603-4d8e-be47-c9a264c3b445
```

```
client.set.default_space(space_uid)
```

```
client.software_specifications.list(limit=100)
```

```
-------------------------------  ------------------------------------  ----
NAME                             ASSET_ID                              TYPE
default_py3.6                    0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12        020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt       069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6          09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12       09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9        0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6            0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                       0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod     1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6                10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl        111e41b3-de2d-5422-a4d6-bf776828c4b7  base
autoai-kb_rt22.2-py3.10          125b6d9a-5b1f-5e8d-972a-b251688ccf40  base
runtime-22.1-py3.9               12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6          154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                     1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6           1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6             1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt    1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6             1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                  20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod     217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda          26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                         295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8              2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6            2b73a275-7cbf-420b-a912-eae7f436e0bc  base
kernel-spark3.3-py3.9            2b7961e2-e3b1-5a8c-a491-482c8368839a  base
pytorch_1.2-py3.6                2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                  2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt       32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37             36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                  390d21f8-e58b-4fac-9c55-d7ceda621326  base
autoai-ts_rt22.2-py3.10          396b2e83-0953-5b86-9a55-7ce1628a406f  base
xgboost_0.82-py3.6               39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt       40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
pytorch-onnx_rt22.2-py3.10       40e73f55-783a-5535-b3fa-0c8b94291431  base
default_r36py38                  41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9           4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                   42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
```

```
pmml-3.0_4.3                    493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6           49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6              4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6          50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
autoai-ts_3.9-py3.8             52c57136-80fa-572e-8728-a5e7cbb42cde  base
spark-mllib_2.4-scala_2.11      55a70f99-7320-4be5-9fb9-9edb5a443af5  base
spark-mllib_3.0                 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9  base
autoai-obm_2.0                  5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1               5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
cuda-py3.8                      5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e  base
runtime-22.2-py3.10-xc          5e8cddff-db4a-5a6a-b8aa-2d4af9864dab  base
autoai-kb_3.1-py3.7             632d4b22-10aa-5180-88f0-f52dfb6444d7  base
pytorch-onnx_1.7-py3.8          634d3cdc-b562-5bf9-a2d4-ea90a478456b  base
spark-mllib_2.3-r_3.6           6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c  base
tensorflow_2.4-py3.7            65e171d7-72d1-55d9-8ebb-f813d620c9bb  base
spss-modeler_18.2               687eddc9-028a-4117-b9dd-e57b36f1efa5  base
pytorch-onnx_1.2-py3.6          692a6a4d-2c4d-45ff-a1ed-b167ee55469a  base
```

```
software_space_uid = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9'
software_space_uid
```

```
'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

```
model_details = client.repository.store_model(model='handwritten-digit-recognition-model_new.
    client.repository.ModelMetaNames.NAME:"CNN Digit recognition model",
    client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
}
```

```
  File "/tmp/wsuser/ipykernel_164/3885713015.py", line 6
    model_id = client.repository.get_model_uid(model_details)
    ^
SyntaxError: invalid syntax
```

SEARCH STACK OVERFLOW

```
model_id
```

```
client.repository.download(model_id,'DigitRecog_IBM_model.tar.gz')
```

```
---------------------------------------------------------------------------
```

```
ls
```

```
----> 1 client.repository.download(model_id,'DigitRecog_IBM_model.tar.gz')
```

TEST MODEL

SEARCH STACK OVERFLOW

```
from tensorflow.keras.models import load_model
from keras.preprocessing import image
from PIL import Image
import numpy as np
```

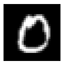
```
model = load_model("mnistCNN.h5")
```

"@hidden_cell" is not an allowed annotation - allowed values include [@param, @title, @markdown].

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your crede
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='is_QZGPyU8oxZr3W-td-LCHXS3QPMaWArILi18FdSyGT',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.ap.cloud-object-storage.appdomain.cloud')

bucket = 'handwrittenimagerecognition-donotdelete-pr-8tlrnykut46vpi'
object_key = 'mnist-dataset-1024x424 (2).png'

streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

```
img = Image.open(streaming_body_1).convert("L") # convert image to monochrome
img = img.resize( (28,28) ) # resizing of input image
```

```
img
```



```
im2arr = np.array(img) #converting to image
im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement
```

```
pred = model.predict(im2arr)
print(pred)
```

```
[[1.0000000e+00 2.3963600e-15 2.4757642e-12 1.2317031e-13 1.0567615e-17
  1.4570932e-12 7.9845136e-10 4.4574440e-15 6.1912351e-09 2.7461991e-11]]
```

```
print(np.argmax(pred, axis=1)) #printing our Labels
```

```
[0]
```