```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "345fe7e6",
   "metadata": {},
   "source": [
    "# IBM cloud deployment"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "a7dfa821",
   "metadata": {},
   "source": [
    "Team ID - PNT2022TMID23122"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "b1cbb86f",
   "metadata": {},
   "source": [
    "# Importing the required libraries"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 5,
   "id": "cc225ad3",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
```

```
"output_type": "stream",

"text": [

 "Requirement already satisfied: tensorflow in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.7.2)\n",

 "Collecting tensorflow\n",

 "  Downloading tensorflow-2.10.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (578.1 MB)\n",

 "\u001b[K     |████████████████████████████████| 578.1 MB 40 kB/s /s eta 0:00:01B 15.2 MB/s eta 0:00:34 |▌               | 84.6 MB 15.2 MB/s eta 0:00:33 |▉               | 90.0 MB 15.2 MB/s eta 0:00:33 |▋              | 177.6 MB 103.6 MB/s eta 0:00:04| 280.1 MB 104.4 MB/s eta 0:00:03/s eta 0:00:03\n",

 "\u001b[?25hRequirement already satisfied: termcolor>=1.1.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.1.0)\n",

 "Requirement already satisfied: flatbuffers>=2.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.0)\n",

 "Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.4.0)\n",

 "Requirement already satisfied: keras-preprocessing>=1.1.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.1.2)\n",

 "Requirement already satisfied: numpy>=1.20 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.20.3)\n",

 "Requirement already satisfied: grpcio<2.0,>=1.24.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.42.0)\n",

 "Collecting absl-py>=1.0.0\n",

 "  Downloading absl_py-1.3.0-py3-none-any.whl (124 kB)\n",

 "\u001b[K     |████████████████████████████████| 124 kB 83.1 MB/s eta 0:00:01\n",

 "\u001b[?25hRequirement already satisfied: astunparse>=1.6.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.6.3)\n",

 "Requirement already satisfied: six>=1.12.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.15.0)\n",

 "Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.23.1)\n",

 "Collecting keras<2.11,>=2.10.0\n",

 "  Downloading keras-2.10.0-py2.py3-none-any.whl (1.7 MB)\n",

 "\u001b[K     |████████████████████████████████| 1.7 MB 88.2 MB/s eta 0:00:01\n",

 "\u001b[?25hCollecting libclang>=13.0.0\n",

 "  Downloading libclang-14.0.6-py2.py3-none-manylinux2010_x86_64.whl (14.1 MB)\n",

 "\u001b[K     |████████████████████████████████| 14.1 MB 78.8 MB/s eta 0:00:01\n",

 "\u001b[?25hCollecting tensorboard<2.11,>=2.10\n",
```

" Downloading tensorboard-2.10.1-py3-none-any.whl (5.9 MB)\n",

"\u001b[K |████████████████████████████████| 5.9 MB 69.7 MB/s eta 0:00:01\n",

"\u001b[?25hRequirement already satisfied: opt-einsum>=2.3.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.3.0)\n",

"Requirement already satisfied: setuptools in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (58.0.4)\n",

"Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (21.3)\n",

"Requirement already satisfied: typing-extensions>=3.6.6 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (4.1.1)\n",

"Requirement already satisfied: google-pasta>=0.1.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (0.2.0)\n",

"Requirement already satisfied: protobuf<3.20,>=3.9.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.19.1)\n",

"Requirement already satisfied: h5py>=2.9.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (3.2.1)\n",

"Collecting tensorflow-estimator<2.11,>=2.10.0\n",

" Downloading tensorflow_estimator-2.10.0-py2.py3-none-any.whl (438 kB)\n",

"\u001b[K |████████████████████████████████| 438 kB 83.2 MB/s eta 0:00:01\n",

"\u001b[?25hRequirement already satisfied: wrapt>=1.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.12.1)\n",

"Requirement already satisfied: wheel<1.0,>=0.23.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from astunparse>=1.6.0->tensorflow) (0.37.0)\n",

"Requirement already satisfied: google-auth<3,>=1.6.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (1.23.0)\n",

"Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (0.4.4)\n",

"Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (0.6.1)\n",

"Requirement already satisfied: requests<3,>=2.21.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (2.26.0)\n",

"Requirement already satisfied: markdown>=2.6.8 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (3.3.3)\n",

"Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (1.6.0)\n",

"Requirement already satisfied: werkzeug>=1.0.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorboard<2.11,>=2.10->tensorflow) (2.0.2)\n",

"Requirement already satisfied: cachetools<5.0,>=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow) (4.2.2)\n",

"Requirement already satisfied: rsa<5,>=3.1.4 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow) (4.7.2)\n",

"Requirement already satisfied: pyasn1-modules>=0.2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow) (0.2.8)\n",

"Requirement already satisfied: requests-oauthlib>=0.7.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.11,>=2.10->tensorflow) (1.3.0)\n",

"Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow) (0.4.8)\n",

"Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow) (2022.9.24)\n",

"Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow) (2.0.4)\n",

"Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow) (3.3)\n",

"Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow) (1.26.7)\n",

"Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.11,>=2.10->tensorflow) (3.2.1)\n",

"Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging->tensorflow) (3.0.4)\n",

"Installing collected packages: absl-py, tensorflow-estimator, tensorboard, libclang, keras, tensorflow\n",

" Attempting uninstall: absl-py\n",

"   Found existing installation: absl-py 0.12.0\n",

"   Uninstalling absl-py-0.12.0:\n",

"   Successfully uninstalled absl-py-0.12.0\n",

" Attempting uninstall: tensorflow-estimator\n",

"   Found existing installation: tensorflow-estimator 2.7.0\n",

"   Uninstalling tensorflow-estimator-2.7.0:\n",

"   Successfully uninstalled tensorflow-estimator-2.7.0\n",

" Attempting uninstall: tensorboard\n",

"   Found existing installation: tensorboard 2.7.0\n",

"   Uninstalling tensorboard-2.7.0:\n",

"   Successfully uninstalled tensorboard-2.7.0\n",

" Attempting uninstall: keras\n",

"   Found existing installation: Keras 2.2.4\n",

"   Uninstalling Keras-2.2.4:\n",

```
"     Successfully uninstalled Keras-2.2.4\n",
"  Attempting uninstall: tensorflow\n",
"    Found existing installation: tensorflow 2.7.2\n",
"    Uninstalling tensorflow-2.7.2:\n",
"     Successfully uninstalled tensorflow-2.7.2\n",
"\u001b[31mERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.\n",
"tensorflow-text 2.7.3 requires tensorflow<2.8,>=2.7.0, but you have tensorflow 2.10.0 which is incompatible.\n",
"tensorflow-metadata 1.5.0 requires absl-py<0.13,>=0.9, but you have absl-py 1.3.0 which is incompatible.\n",
"autoai-ts-libs 1.1.9 requires tensorflow<2.8,>=2.7.0; python_version >= \"3.9\", but you have tensorflow 2.10.0 which is incompatible.\u001b[0m\n",
"Successfully installed absl-py-1.3.0 keras-2.10.0 libclang-14.0.6 tensorboard-2.10.1 tensorflow-2.10.0 tensorflow-estimator-2.10.0\n"
   ]
  }
 ],
 "source": [
  "!pip install tensorflow --upgrade"
 ]
 },
 {
 "cell_type": "code",
 "execution_count": 31,
 "id": "53a54029",
 "metadata": {},
 "outputs": [],
 "source": [
  "import numpy as np\n",
  "import tensorflow #open source used for both ML and DL for computation\n",
  "from tensorflow.keras.datasets import mnist #mnist dataset\n",
  "from tensorflow.keras.models import Sequential #it is a plain stack of layers\n",
  "from tensorflow.keras import layers #A Layer consists of a tensor- in tensor-out computat ion funct ion\n",
  "from tensorflow.keras.layers import Dense, Flatten #Dense-Dense Layer is the regular deeply connected r\n",
  "#faltten -used fot flattening the input or change the dimension\n",
```

```
  "from tensorflow.keras.layers import Conv2D #convolutional Layer\n",
  "from keras.utils import np_utils #used for one-hot encoding\n",
  "import matplotlib.pyplot as plt   #used for data visualization"
 ]
},
{
 "cell_type": "markdown",
 "id": "41612196",
 "metadata": {},
 "source": [
  "# Load data"
 ]
},
{
 "cell_type": "code",
 "execution_count": 32,
 "id": "3d416c13",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz\n",
    "11493376/11490434 [==============================] - 0s 0us/step\n",
    "11501568/11490434 [==============================] - 0s 0us/step\n"
   ]
  }
 ],
 "source": [
  "(x_train, y_train), (x_test, y_test)=mnist.load_data () #splitting the mnist data into train and test"
 ]
},
{
```

```
"cell_type": "code",
"execution_count": 33,
"id": "087fbafd",
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "(60000, 28, 28)\n",
   "(10000, 28, 28)\n"
  ]
 }
],
"source": [
 "print (x_train.shape)  #shape is used for give the dimens ion values #60000-rows 28x28-pixels\n",
 "print (x_test.shape)"
]
},
{
 "cell_type": "code",
 "execution_count": 34,
 "id": "4758cec8",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
     "         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
     "         0,   0],\n",
     "       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
     "         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
     "         0,   0],\n",
```

```
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,\n",
"         18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170,\n",
"        253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253,\n",
"        253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253,\n",
"        253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253,\n",
"        205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,\n",
"         90,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 139, 253,\n",
"        190,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  11, 190,\n",
"        253,  70,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"          0,   0],\n",
"       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  35,\n",
```

```
"      241, 225, 160, 108,   1,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"       81, 240, 253, 253, 119,  25,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,  45, 186, 253, 253, 150,  27,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0,  16,  93, 252, 253, 187,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0,   0,   0, 249, 253, 249,  64,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,  46, 130, 183, 253, 253, 207,   2,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  39,\n",
"      148, 229, 253, 253, 253, 250, 182,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  24, 114, 221,\n",
"      253, 253, 253, 253, 201,  78,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,   0,   0,  23,  66, 213, 253, 253,\n",
"      253, 253, 198,  81,   2,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,   0,   0,  18, 171, 219, 253, 253, 253, 253,\n",
"      195,  80,   9,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0,  55, 172, 226, 253, 253, 253, 253, 244, 133,\n",
"       11,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
"        0,   0],\n",
"     [  0,   0,   0,   0, 136, 253, 253, 253, 212, 135, 132,  16,   0,\n",
"        0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
```

```
       "          0,   0],\n",
       "       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0],\n",
       "       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0],\n",
       "       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,\n",
       "          0,   0]], dtype=uint8)"
      ]
     },
     "execution_count": 34,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "x_train[0]"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 35,
   "id": "ba3df521",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<matplotlib.image.AxesImage at 0x7f0004821880>"
      ]
     },
     "execution_count": 35,
```

"metadata": {},

"output_type": "execute_result"

},

{

"data": {

"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAPsAAAD4CAYAAAAq5pAIAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlc
nNpb24zLjUuMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy8/fFQqAAAACXBIWXMAAAsTAAALEwEAmpwYAAAOdEl
EQVR4nO3df6zV9X3H8deLyy+HPwo6lAGZlrFO107sbrUNtrExGkUzwM6mLLMsM6Pbyq5lf9S16SSLWbTxxx5bYamEy
6cl0TmvAxKylxMxqq/VimEKZhTlqEQq1LIoieOG+98c9Lrd6v59zPb/h/XwkN+ec7/t8z/fNyX3xPfd8vt/vxxEhAMe/cd1
uAEBnEHYgCcIOJEHYgSQQIO5DE+E5ubKInxWRN6eQmgVQO6U29HYc9Wq2psNu+TNI/SuqT9E8RcUvp+ZM1RRf44m
Y2CaDgmdhYWWv4Y7ztPklfl3S5pHMkLbF9TqvvOdK5kla2pi0ArdZM2GdK+tmIx7tZqy36Pkrx7tqy36F
7WW2B2wPDOpwE5sD0Ixmwj7alwDvOfY2IlZGRH9E9E/QpCY2B6AZzYR9l6TTIx7tPkrS7PkrS7uXYAtEszYX9W0lzbZ9meKOl
zkta3pi0Ardbw0FtEHLG9XNJ3NDz0tjoitrasMwAt1dQ4e0Q8JumxFvUCoI04BGEHkiDsQBIEHkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQ6eilp5HP4io9V1n7zYV1719xnP6SLGeDXt2IAnCDiRB2IEkCDuQBGEHkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQYZz8O7PjrOPbSW6W8Bd97Bv/RGEKhiDcSQxAS96xdSrfR
WJ7PvyfFze9OpsNveKemApKOSjkSfNqTg2ewAiCDiRB2IEkCDuQBGEHkiDsQBKEHUiCsQJNKxcTcgGcHkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQYZz8O7PjSOZW1dbPLl4KxXevWNBsX7mPS8W6+Wz3
dFJ7NmBJAg7kARhB5Ig7EAShB1Ig7EAShB1IgrADiRB2IAnCDiRB2IEkCDuQBGEHkmCcvQM8qeTzT9vxZ1Hm7eZe8c
xjEnh1IgrADiRB2IEkCDuQBGEHkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQYZz8O7PjSQeuHJesX7P7G8U69Vnqw87uu60ytr4Dw4W133Hh5g8U63/f/0ixvnjK94r1
0lj4kMpTUTfril97rbJ204rytS9Y1OJmekDdPbvt
1bb32d4yYtkK26/Y3WdDeNgE0aywf4++TdNkoy++iI+qQ0FNQ9kvZpDrjYlWQ5KCBMiOA2kuYrF4t1GNMJVtWc2xcCd2dcS
5yKwvQDMoFj7m0PHtA9BY/bc
aT8KmNIrJy8a+Y2RbVSz7O1sbDbaP/j
Qb/AWFcrcqPJdsTQ3
j42auecnh8nZstk+oHtQqfPB+SFg9srBeX+F5rsKxT2y9H1Ojz7iwqUhETrtbEGQ5cPH/kpz4Dx3vVfq4
U8JMRiQ1KhHj2HpqXiBxZxJVWXXfb
UoruWwSCqV7GDpQ/+ipnrkFj49
2J48UtQ4hU/AjXebqIZrHDHxEL3ixWZEhPQpS1ySM2e2sl/zx/Z1SF0cKi8KuYg0I1fv6vRiyPK+S7UVGz9Vn+s2rnQ+S8M4eTS3kTRpHDyM3o2ne0gKhz94
FmmE97uV8aFzswXsLEoqFNRTtBHdtP1ZkC82SRyhrlL6Dk/V4H8VbQ8OGxGe
/4M6Ujp0ZD2c1yH/7O69Sz7FnTPxprJO1qg3Rf8Ulcm3XtBK0FFR7/v8Tjqlwz
T1mT/cr6RvZ9h+lWj0K/Lz8/E6BM5i7Wja4d05RGm+e4wGmBBBQRQmE/OBjf6I
KMg0Tu8tRKZ3FWtFpo0I/l5OH8H2vQb6ABbz7U3nRTU4WQ7WmU58hL6Bm7B2YRBO73Z
GBQ1vrcY9w2i+ggcuyFtH7hrgVbJ+r4z35hjnd/7FBc4Tu2EWvN3E7PdqVWi1B/Pb
JdBG7N7f/3Zbu6j5W/J7NXuH5/Wc/W9ZGeZqRWTeTp9I2cYn+SVf07cVLY3l5bZdm+L7
BKCMeWWWWCd9Wuni/3aN0ytn6a7J7OWq1dTb7Df6TRJ7qvKKoC8sBDEH3CgJR2P7z4zeK/4yYhYwtvW7qc6NyyTV7DR3f8wJJGJcBNZLUgahv7R5LM+aSq/kG6dJQeyX
ZmcdmtC2OAAN5T9E7d1X6fQUKG/PeE8zBTmHSWyW2VLbO8Zsi7rOVaDfDvEsgHhzD4r1
RrA/V+RU5ovLlnv/5qwsra6c89HRxXbQWe3YgCcIOJEHYgSQIO5AEYQeSI
dcX67Ic4TbVXsGcHkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQYZz8O7PjSOZW1dbPLl4KXKxevWNBsX7mPS8W6+Wz3
dFJ7NmBJAg7kARhB5Ig7EAShB1IgrADSRB2IAnG2Y8Bv/yzTxrz11zZ6Fan1L5J4OHivXDVx4s1ocOHCjW0TvqsXzHsp2vebCdRjOQtTsocOHXuwKEYMGR5Z8Ks1X9
z/aWEcum2d5ge3vtdmp72CTQrLF8jf6epMtOWnaTpI0RMVfSxtpjAD2sMewR8aSk909avEjSmtrfayQtbm1bAFqt0R/oZkTEntrf70qaUfVE28tsD9ge
Go+IPZJUu51eNwYqnAF
sYYjfp0fK42D5XJlZh6Bl56
iv/rNl/PYmxGZLdiZmNwdk9W9RVY2T7KGEhXd2tXGb7h8dRC6dj/8W7VA8TWqFMu7C7F9rm3Ua6IGtALffG
3QnkH6tAqEO5dtenKGSWqm5TH6Yr0wa69fJqFq+yR1U5xzFWUd4J7bFDMXgP6aZbGQp0PdBA8QrZwOqa1zQQ2u2
5HXyzt7Xkn62XhF7IWRqyIYqnKmwmTMl
sAtTGGk6m1P2H22J8BWrpRE/KFQwX
ulUvYI7Sj5Hh9EiH9bryhJgQIrh4o5/fiBsyBM3gGhhbp0YS2AOi
cd+Cf1KxLEZe6Cg8gK6EOLZxt
0GOP5oTGEv4kZd6BbYNAn1JGPwgCOPkfDgtFlPLpBSXx
j41XHm07WJ3bXf
q3tMGZ8AQhK8ckB3qzJB4q7bvkA7
qBXmjTm8Ylti6A3OESq0k4S4pOxDpN9mrGZ8qRAHPXjnr47Fbc
3Ut6hjD4Hdq3uoTf9p0pfS6kOqVuaE3B
OSgXRgTzBRUXSmVbf8IC0dcrTMgaJAuxFNWmWHWx39lUeBvbJqSj6rmNcgBMlZhUFhNlAnY
4bXc1hrcSX3uGmsAW0h0wSOmD2l0
GJvTtgm8ne0w5W3cGy34eKS8K2bz
3F1tpzaRZM0UDvX9qXXTIL+pwKLzTp0d3ep
BvRNEbvl0bcKy4QOsqRsv91w8hj5vSxAeN7bHCHP6GkdC0Ipv9iIkwWw9AWrGA6aYw1VNhRhPhdNx5hEzhlDTOQkuq
cn7h1RSMpFmGf9nRxZYR3Jwb3kgbt82SHbqkr2Zp8gFDJcgCcN14G0K6ZrTSnL5H1d3sC8J3QiRxNOb0WA4ioWKJhAzGqMRn9
GF2cXWWC4HqVxz8hOB3W6AWcjJcHpr3s4VEqqgFoKBo3bxF8Z8E7Cm4LPpDeQI4SR41jmtfyRXLMXb8pFlMnx8HeGkzFX
SLfWl22DPxSgqkd3zU4bUrP2q8dJCsXRQtnPZm6nlF0yXgQ6lN00QyNu3B1X7hmsV4emLJJyLUZc9CAXBnC1A7EjtmCiUvmo
gX32NuJOfAexu5vi8dCNHtu1JdmHJ3vvT7fgWfE3TnNQ3zNTsm3gqh6s5bECJHm1IrJw55XUeh0wBuz2yClNZbpR95bEbG0F0M
zdHwFSRJH4sfmgtgjwvz0jP3vExOAaBoSt2AHkVb9HnAD6cvx0ytX0SD1Ws3owQKQXuqmA6w17TAJFcZjDc6ZP0u6z0rhqhxxOgy
czWrc6L3SVXy8nGLOWQAUz49kUjjTbT3Vv8wwjwJg6mxMDMkQDUvWd7BVAW6pyDWmmiC9PwHhlS6S1bKxEvtMTyJz9uGrgtO2
E5XQmwvt3SRzZXpxOgQBiSyqg5MIsVR4ZdIKfJCL9AFvKNEj5t1DSCnVhYZaB3YfGM9WE0XS1wO2uRmRnDbU9JhA9aBMAFGf2Ulp2
6zfZKkTbY31Gp3RsRt7WsPQKuMZX72PZL21O4fsL1N0sx2Nwagtd7X3+y2z5R0rbUyvWWWZ7w
PbAoA431y2Aho057LZPlPSwpOsj4nVJd0uaI2mehvf8f8t4+2XkSsjIj+iOifoEnNdwygIWMKu+0JGg762oj4tiRFxN6IOBoR

Q5JWSTq/fW0CaNZYvo23pHslbYuIO0YsnzHiaYslbWl9ewBaZSzfxs+XdI2kF2xvri37sqQltudJCkk7JX2hDf0BaJGxfBv/p
KTRrkP9WOvbAdAuHEEHJEHYgSQIO5AEYQeSIOxAEoQdSIKwA0kQdiAJwg4kQdiBJAg7kARhB5Ig7EAShB1IwhHRuY3
Zv5D00xGLTpP0ascaeH96tbde7Uuit0a1srffjIhfH63Q0bC/Z+P2QET0d62Bgl7trVf7kuitUZ3qjY/xQBKEHUii22Ff2eXtl/
Rqb73al0RvjepIb139mx1A53R7zw6gQwg7kERXwm77Mtsv2t5h+8Zu9FDF9k7bL9SmoR7oci+rbe+zvWXEsmm2N9je
XrsddY69LvXWE9N4F6YZ7+p71+3pzzv+N7vtPkk/kXSJpF2SnpW0JCJ+3NFGKtjeKak/Irp+AIbtT0l6Q9K3IuLDtWVfk7Q
/Im6p/Uc5NSK+1CO9rZD0Rren8a7NVjRj5DTjkhZJ+hN18b0r9PVZdeB968ae/XxJOyLipYh4W9IDkhZ2oY+eFxFPSNr/r
sULJa2p3V+j4V+WjqvorSdExJ6IeK52/4Ckd6YZ7+p7V+irI7oR9pmSfjbi8S711nzvIem7tjfZXtbtZkZxekTskYZ/eSRN73I/
71Z3Gu9Oetc04z3z3jUy/XmzuhH20aaS6qXxv/kR8VFJl0v6Yu3jKsZmTNN4d8oo04z3hEanP29WN8K+S9LsEY9nSdrd
hT5GFRG7a7f7JD2i3puKeu87M+jWbvd1uZ//10vTeI82zbh64L3r5vTn3Qj7s5Lm2j7L9kRJn5O0vgt9vIftybUDJ7I9WdJl6r2pqNdLWlK7v0TSd3ZaG+mMzG3YtssSy/1XWHQFMwE3iMX2b+lwvHmAp7gFjcoQr3PfICA7PKbYtXhEEzgZVdVYWOE6VPMvfbxbf+XvO0/fmSOpQK3IAAABRVRqVEHpQ+SUaucyzAMlKpyQ4NTXM2Ym7v05Ra7nJlZwF+Y6++bSxmNhtjmfRvWp3eSpb7mU+rBxsjW7ZhTlh/AqiK7PFm5hfo/1UMY88HY+AdI6A1RGNpN
hT5GFRG7a7f7JD2i3puKeu87M+jWbvd1uZ//10vTeI82zbh64L3r5vTn3Qj7s5Lm2j7L9kRJn5O0vgt9vIftKbUvTmR7iqR
L1XtTUa+XtLR2f6mkdV3s5Vf0yjTeVdOMq8vvXdenP4+Ijv9IWqDhb+T/W9JXutFDRV8flPSftZ+t3e5N0v0a/lg3qOFPR
NdKOlXSRknba7fTeqi3f5H0gqTnNRysGV3q7UIN/2n4vKTNtZ8F3X7vCn115H3jcFkgCY6gA5Ig7EAShB1IgrADSRB2IAn
CDiRB2IEk/g/lX1RCIuK9nwAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "plt.imshow(x_train[6000])   #ploting the index=image"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 36,
   "id": "8f87b22d",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "0"
      ]
     },

```
  "execution_count": 36,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "np.argmax(y_train[6000])"
]
},
{
"cell_type": "markdown",
"id": "a6bd217a",
"metadata": {},
"source": [
 "# Reshaping Dataset"
]
},
{
"cell_type": "code",
"execution_count": 37,
"id": "d298102b",
"metadata": {},
"outputs": [],
"source": [
 "#Reshaping to format which CNN expects (batch, height, width, channels)\n",
 "x_train=x_train.reshape (60000, 28, 28, 1).astype('float32')\n",
 "x_test=x_test.reshape (10000, 28, 28, 1).astype ('float32')"
]
},
{
"cell_type": "markdown",
"id": "113e9a6a",
"metadata": {},
"source": [
```

```
  "# Applying One Hot Encoding"
 ]
},
{
 "cell_type": "code",
 "execution_count": 38,
 "id": "76ba7afe",
 "metadata": {},
 "outputs": [],
 "source": [
  "number_of_classes = 10  #storing the no of classes in a variable"
 ]
},
{
 "cell_type": "code",
 "execution_count": 39,
 "id": "ff17626c",
 "metadata": {},
 "outputs": [],
 "source": [
  "y_train = np_utils.to_categorical (y_train, number_of_classes) #converts the output in binary format\n",
  "y_test = np_utils.to_categorical (y_test, number_of_classes)"
 ]
},
{
 "cell_type": "markdown",
 "id": "be53ea02",
 "metadata": {},
 "source": [
  "# Add CNN Layers"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 40,
 "id": "119a34ab",
 "metadata": {},
 "outputs": [],
 "source": [
  "#create model\n",
  "model=Sequential ()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 41,
 "id": "3e49c070",
 "metadata": {},
 "outputs": [],
 "source": [
  "#adding modeL Layer\n",
  "model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))\n",
  "model.add(Conv2D(32, (3, 3), activation = 'relu'))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 42,
 "id": "8f94369b",
 "metadata": {},
 "outputs": [],
 "source": [
  "#flatten the dimension of the image\n",
  "model.add(Flatten())"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 43,
 "id": "84765fae",
 "metadata": {},
 "outputs": [],
 "source": [
  "#output layer with 10 neurons\n",
  "model.add(Dense(number_of_classes,activation = 'softmax'))"
 ]
},
{
 "cell_type": "markdown",
 "id": "2048aef5",
 "metadata": {},
 "source": [
  "# Compiling the model"
 ]
},
{
 "cell_type": "code",
 "execution_count": 44,
 "id": "856304e2",
 "metadata": {},
 "outputs": [],
 "source": [
  "#Compile model\n",
  "model.compile(loss= 'categorical_crossentropy', optimizer=\"Adam\", metrics=['accuracy'])"
 ]
},
{
 "cell_type": "code",
 "execution_count": 45,
 "id": "f0468d09",
 "metadata": {},
 "outputs": [],
```

```
  "source": [
   "x_train = np.asarray(x_train)\n",
   "y_train = np.asarray(y_train)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "974dd9c8",
  "metadata": {},
  "source": [
   "# Train the model"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 46,
  "id": "bcde8241",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Epoch 1/5\n",
     "1875/1875 [==============================] - 126s 67ms/step - loss: 0.2690 - accuracy: 0.9514 - val_loss: 0.0884 - val_accuracy: 0.9728\n",
     "Epoch 2/5\n",
     "1875/1875 [==============================] - 125s 66ms/step - loss: 0.0676 - accuracy: 0.9789 - val_loss: 0.0803 - val_accuracy: 0.9788\n",
     "Epoch 3/5\n",
     "1875/1875 [==============================] - 125s 67ms/step - loss: 0.0458 - accuracy: 0.9852 - val_loss: 0.0791 - val_accuracy: 0.9788\n",
     "Epoch 4/5\n",
     "1875/1875 [==============================] - 125s 67ms/step - loss: 0.0387 - accuracy: 0.9883 - val_loss: 0.1079 - val_accuracy: 0.9759\n",
```

```
    "Epoch 5/5\n",
    "1875/1875 [==============================] - 125s 67ms/step - loss: 0.0280 - accuracy: 0.9909 - val_loss: 0.0991 - val_accuracy: 0.9774\n"
   ]
  },
  {
   "data": {
    "text/plain": [
     "<keras.callbacks.History at 0x7f0004e4a640>"
    ]
   },
   "execution_count": 46,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "#fit the model\n",
  "model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5, batch_size=32)"
 ]
},
{
 "cell_type": "markdown",
 "id": "81738cb4",
 "metadata": {},
 "source": [
  "# Observing the metrics"
 ]
},
{
 "cell_type": "code",
 "execution_count": 47,
 "id": "00a1c2dc",
 "metadata": {},
```

```
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Metrics (Test loss &Test Accuracy) : \n",
    "[0.09910603612661362, 0.977400004863739]\n"
   ]
  }
 ],
 "source": [
  "# Final evaluation of the model\n",
  "metrics = model.evaluate(x_test, y_test, verbose=0)\n",
  "print(\"Metrics (Test loss &Test Accuracy) : \")\n",
  "print(metrics)"
 ]
},
{
 "cell_type": "markdown",
 "id": "b3add595",
 "metadata": {},
 "source": [
  "# Test The Model"
 ]
},
{
 "cell_type": "code",
 "execution_count": 48,
 "id": "0508c462",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
```

"text": [
"[[9.1516389e-13 8.1778777e-19 2.4542002e-14 1.7823329e-07 5.2257418e-04\n",
" 5.8763407e-09 6.2800168e-17 3.1880148e-07 6.3142506e-03 9.9316275e-01]]\n"
]
}
],
"source": [
"prediction=model.predict(x_test[6000:6001])\n",
"print(prediction)"
]
},
{
"cell_type": "code",
"execution_count": 49,
"id": "45a591fc",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"<matplotlib.image.AxesImage at 0x7f00044a3370>"
]
},
"execution_count": 49,
"metadata": {},
"output_type": "execute_result"
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAPsAAAD4CAYAAAAq5pAIAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlc
nNpb24zLjUuMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy8/fFQqAAAACXBIWXMAAAsTAAALEwEAmpwYAAANXkl
EQVR4nO3df6xX9X3H8ddLdkGHaPglQyTVMTZna0R3q3Msi4uuUZYWmqUd/tGyHKeCNiiAAWBEOpnLABNvNtfQmZmQ5cnARMjTpfLUR8tO68mWpOzjGtidk1O1f6
W4AQHcQdiAJwg4kQdiBJAg7kMQvdHNlkz0lLtbUbq4SSOU9/a9OxHGPVasVdtt3SvqKpEmS/jEiHi3Nf7Gm6hbfXme
VAAo2xYaWtbZ3421PkvT3ku6SdJ2ku21f1+77AeisOn+z3yzp9Yh4IyJOSHpW0tJm2gLQtDphnydydp72N7pe0

h20PDOl5jdQDqqBP2sb4E+MC5txGxKiIGI2JwQFNqrA5AHXXCvk/S/FGvr5K0v147ADqlTtg3S1po+xrbkyUtl7S2mbYA
NK3tQ28RcdL2fZL+WSOH3p6KiJ2NdQagUbWOs0fEOknrGuoFQAdxuiyQBGEHkiDsQBKEHUiCsANJEHYgCcIOJEHYgS
QIO5AEYQeSIOxAEoQdSIKwA0kQdiAJwg4kQdiBJAg7kARhB5Ig7EAShB1IgrADSRB2IAnCDiRB2IEkCDuQBGEHkiDsQB
KEHUiCsANJEHYgiVpDNtveI+ldSacknYyIwSaaAtC8WmGv/G5EHG7gfQB0ELvxQB0wx6Svmd7i+2VY81ge6XtIdtDwz
pec3UA2lV3N35xROy3fYWk9bZ/EhEbR88QEaskrZKkyzwjaq4PQJtqbdkjYn/1eEjSC5JubqIpAM1rO+y2p9qedua5pI9J
2tFUYwCaVWc3fo6kF2yfeZ9vRsR3G+kKQOPaDntEvCHhphgZ7AdBBHHoDkiDsQBKEHUiCsANJEHYgCcIOJEHYgSQIO5
AEYQeSIOxAEoQdSIKwA0kQdiCJJm44iZpeW/3RYv3W614v1m+b8W/fe8WrL2srL97fV08/fe8eyYv2tV+cU67/69E9b1t540
MVlp2y9tFi/8rGXi3W8H1t2IAnCDiRB2IEkCDuQBGEHkiDsQBKEHUjCEd0bpOU+yz4+caf5wdmjfvvXmbxpT04ZFqrYrrm
DbKerdg6faFn78ED53/XSewBmJbB5dfn6nXHYJvLW05a/JSSWuq52skLWu2LQBNa/cLujkRcUCSqscrWs1oe6XtIdtDwzre5uoA1NXxb+MjYlVE
DEbE4ICmdHp1AFpoN+wHbc+VpOrxUHMtAeiEdsO+VtKK6vkKSS820w6AThn3OLvtZyTdJmmW7b5X2SvijpUUnP2b5H
0luSPPtXJJs93H7r6nVrL/9XhjxTrL31hccvamx8v/z6ftntSsT7j998u1p+/9pvF+vSLLinWS2aOs+yxeeVj+Je3veYL07hhj4i7
W5Ty3YUCOI9xuiyQBGEHkiDsQBKEHUiCsANJcIlrH3j7VOthjSXp5XsHi/XJ/XJ/765ZW3hd9tq6f/9bbl8x5/8abG+5aEnaja
AprBlB5Ig7EAShB1IgrADSRB2IAnCDiRB2IEkOM7eB+ZN+sVi/dee2FWs/9Pm1kNC+0T59/kl/1muX/XXL5eXP3y6y6WEf/
YMsOJEHYgSQIO5AEYQeSIOxAEoQdSIKwA0lwnL0LDv/v/rleUZPz53U7/XL7HbH+F+rX376z1/i/iXbTz6/iDovO7LyLg6f72ay0lP5+X/nj6TRFrXk1fuMxNbQCiehHwGtMu
wwX6zM37ivWTzbZZzAWALTuQBGEHkiDsQBKEHUiCsANJEHYgCcIOJMFx9i6Yv+Isf9v6lWL9zZtefyFXU/Hq4+vQN/6vrbLcC/Df6aRmYi/5nnFuIGSFGtCYJ3fCeUOsu3XbVt4laH0pKm83eO9d2U+2MSzNgA+yazdJQkEPOU5VS2FkeEHAt0zbzUHVSv
q/uKyzxz9lWL9xu/sLdb/cvaPivXV/zO/Ze1zl5ff+9PPPlCsX7P3h8U63m/cLbvtp2wfsr1j1LRHbL9te1KoayK78
V+TdOcY0x+PiEXVz7pm2wLQtHHHDhrA5AHe2G/auSFkhaJOmjMrjMjjDMDmhKm6sDUFdbYY+PiEXVkXEYMDmtJOjwAAMKW28o1gtqfXtTQH0hbbCbnvuqJefHkkKiJOS1otqfXtTQH0hbbCbnvuqJefHkfILSj1bwA+oMjojyD/Yyk2yTNkn
RQ0her14skhaQ9ku6NiAPjrewyz4+aruS2aXx4av68Sbl7asvbr8H4rLfvyOPyzWT+3a
3VZPF7JNsUFH44jHqo17Uk1EjHXGR/ksDwB9h9NlgSQIO5AEYQeSIOxAEoQdSIJLXM8D/mH5D/mH5MtIr+/hKz0mzZ7cuLi8v
+5M/n1asL/xsGw0lxpYdSIKwA0kQdiAJwg4kQdiBJAg7kARhB5LgODv61urFa4rABDv61urFa4r1x3R9lzq5MLBlzBDv61urFa4r1x3R9lzq5MLBlB5Ig7EAShB1IgrADSR
B2IAnCDiRB2IEkOM6OzrpozA6ozrpozLsaT8if/fgPivVZeq3t9ILTuQBGEHkiDsQBKEHUiCsANJEHYgCcIOJMFxdnW6fKQ4CV/
vKB8Q/y1mtn2e2c07pbd9nzb37e9y/ZO2fQ
eg2gT40b9og4EBFbq+fvStolaZ6kkpZLO3DdojaRlHeoQRAPAPO6Qs621dLuHSJkizIuKANPILQdlVLZZZaXvI9tCwjtdsF0C7J
hx225dK+pakByLCi6hEYEyMDmtJOjwAaMKGw2x7QSNC/ERHfriYftD23qs+VdKgzLQJowriH3mxb0pOSdkXEl
0eV1kpaIenR6vHFHFjnSItK6dsr9Y59DbuZnIcfbFkj4jabvtbdW0hzUS8uds3yPpLUmf6kiHAHABoxbtgj4geGjvNy9ZE2wH
QKZwuCyRB2IEkCDuQBGEHkiDsQBJc4orOWrt1UQuJ1rU9gQ4UzfJnlKQqxrXxgSzgpvlzIzjqtbraq/TYs6m1V2cgcp372XsvS88FK46svn/bfTXe
T2kROqjkp6cGI2Gp7mqQttdXtXtccj2L+xX
FsustD1ke2hYx+t1C6BtEw677UslfFVAxFxVNJXJS2QtEgjW/4vjbQiMGBzQlPodA2jLhMJue0AjQf9GRHxbkiLi
YESciojTkZ2LurlzbQKoa9yw2/Y0SVPOPJf0MUk7mmoMQLPq7MbPkfSC7TPv882I+G4jXQGNYvKojzb41Yp92bQn9OzRRPDAn7Qb5FkRo1X0GUnbbW+rpj0bW7G1lzq
T+L3bTxfvGWbh8iSvOzUS+jf+BpLEuSuaYOnAe4Qw6IAnCDiRB2IEkCDuQBGEHkiDsQBLcShoddeq/jrSsfXbniuKy/3bD
c023kxpbdiAJwg4kQdiBJAg7kARhB5Ig7EAShB1IwhHRvZXZ70h6c9SkWZIOd62Bj6Zfe+vXviR6m6gme1sQEZePVehq2D+0cnsoIga0r0uMs1B6/f6Qa5gQEIsQR4AYuTAmbk3pyHQOvt9J+fTD5TP/3T/3FfbTbx63DM6jlZsfE8zxXEv2+6Thmhz8DMjpYJhnSnS2qwKcbpwAqB/bg6Sf3TP9+8rj+yP/+B
B9YuT0UEYM9a6CgX3vr174kemtXt3pjNx5IgrADSfQ67Kt6vP6Sfu2tX/uS6G2ietZbrz+zA+gBjsiNrq4wkOZbNq5PJcRp7djMbh9OMrqmoJKWAqUNLnkWR6+W7e29vi9mw1u62iWjUn0Q7u3KyF73PRj29JHTqvGN67r9i6qa7wzD9yVZjY06b2Xd7fOdsEbsPjN1vr3O8nB8kKrS7MVi9lG+u
9p+1Xbr9t+qBc9tGJ7j+3ttrfZHupxL0/ZPmR7x6hpM2yvt727+jy9yvt727ehxzjL0e9faI7berz26b7SU96m2+7e/b3mV7p+37q+k9/
ewKfXXlc+v63+y2J0l6TdLvSdonabOkuyPix11tpAXbeyQNRKTxPtl+E0Ck+9kU5PSK+0Ce9PSLp
WK+H8a5GK5o7ephxScsk/ZF6+NkV+vq0uvC59WLLfrOk1yPijYg4kYg4IelZSUt70Effi4i4nN6ks6+1ctSSWuq52s08p+l61r01hci4
kBEbK2evyvpzDDjPf3sCn11RS/CPk/S3lGv96m/xnsPSd+zvcX2yl43M4Y5EXFAGvnlkTS7x/3cY25vPrpRdjHGkqqn47/LY6ImyTdJenz1e4o11dJ2t+DPsYUEfurx0OSXtD/DUV98
MwIutXjoR7383P9NIz3WMOMqw8+u14Of96LsG+WtND2lbYnSvq0pLGGi+JUpiSsSV/NgDP2WT+nMbVlXSpfgA4O05ke6qkj6n/hqJeK+nMbVlXS
Hqxh728T78M491qmHH1+LPr+fDnEdH1H0LNPKN/H9I+ote9NCir1+W9KPqZ2eve5P0jEZ264Y1skd0j6SZkjZZI2l09zuij
3r4uabukVzQSrlk96u1rN/S/3r92RX66srnxumyQBKcQQckQdiBJAg7kARhB5Ig7EAShB1IgrADSfwfeAvlI2T
seF0AAAAASUVORK5CYII=\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

```
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
 "source": [
  "plt.imshow(x_test[6000])"
 ]
},
{
 "cell_type": "code",
 "execution_count": 50,
 "id": "4cf45ba6",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "[9]\n"
   ]
  }
 ],
 "source": [
  "import numpy as np\n",
  "print(np.argmax(prediction, axis=1)) #printing our Labels from first 4 images"
 ]
},
{
 "cell_type": "code",
 "execution_count": 51,
 "id": "81f007d9",
 "metadata": {},
```

```
 "outputs": [
  {
   "data": {
    "text/plain": [
     "9"
    ]
   },
   "execution_count": 51,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "np.argmax(y_test[6000:6001]) #printing the actual labels"
 ]
},
{
 "cell_type": "markdown",
 "id": "0e2f8106",
 "metadata": {},
 "source": [
  "# Save The model"
 ]
},
{
 "cell_type": "code",
 "execution_count": 52,
 "id": "8fc89e72",
 "metadata": {},
 "outputs": [],
 "source": [
  "# Save the model\n",
  "model.save('models/mnistCNN.h5')"
 ]
```

```
},
{
 "cell_type": "code",
 "execution_count": 53,
 "id": "d3fe4109",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "/home/wsuser/work/models\n"
   ]
  }
 ],
 "source": [
  "cd models"
 ]
},
{
 "cell_type": "code",
 "execution_count": 54,
 "id": "502acee5",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "mnistCNN.h5\r\n"
   ]
  }
 ],
 "source": [
```

```
   "!tar -zcvf handwritten-digit-recognition-model_new.tgz mnistCNN.h5"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 55,
  "id": "86720654",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Collecting watson-machine-learning-client\n",
     "  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)\n",
     "\u001b[K     |████████████████████████████████| 538 kB 21.5 MB/s eta 0:00:01\n",
     "\u001b[?25hRequirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.3.3)\n",
     "Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.18.21)\n",
     "Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (4.62.3)\n",
     "Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.3.4)\n",
     "Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.8.9)\n",
     "Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2022.9.24)\n",
     "Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.26.7)\n",
     "Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0)\n",
     "Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.26.0)\n",
     "Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (1.21.41)\n",
     "Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.5.0)\n",
```

"Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.10.0)\n",

"Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (2.8.2)\n",

"Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (1.15.0)\n",

"Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)\n",

"Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)\n",

"Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (2.0.4)\n",

"Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (3.3)\n",

"Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (2021.3)\n",

"Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (1.20.3)\n",

"Installing collected packages: watson-machine-learning-client\n",

"Successfully installed watson-machine-learning-client-1.0.391\n"

]

}

],

"source": [

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

]

},

{

"cell_type": "markdown",

"id": "bd05ddda",

"metadata": {},

"source": [

"# Cloud deploy"

]

},

{

"cell_type": "code",

```
 "execution_count": 56,
 "id": "20abb86b",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "<ibm_watson_machine_learning.client.APIClient at 0x7f00044af820>"
    ]
   },
   "execution_count": 56,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "from ibm_watson_machine_learning import APIClient\n",
  "credentials ={\n",
  "    \"url\":\"\",\n",
  "    \"apikey\":\"\"\n",
  "}\n",
  "client = APIClient(credentials)\n",
  "client"
 ]
},
{
 "cell_type": "code",
 "execution_count": 57,
 "id": "30ad9b4a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
```

"{'resources': [{'entity': {'compute': [{'crn': 'crn:v1:bluemix:public:pm-20:jp-tok:a/53f9f6400d0d44889534e8abcd2dfe39:0f4376b6-c944-4b27-b23e-48b54d8f4bbd::',\n",
" 'guid': '0f4376b6-c944-4b27-b23e-48b54d8f4bbd',\n",
" 'name': 'Watson Machine Learning-sp',\n",
" 'type': 'machine_learning'}],\n",
" 'description': '',\n",
" 'name': 'digitrecognition',\n",
" 'scope': {'bss_account_id': '53f9f6400d0d44889534e8abcd2dfe39'},\n",
" 'stage': {'production': False},\n",
" 'status': {'state': 'active'},\n",
" 'storage': {'properties': {'bucket_name': '63888f6f-d1ef-475c-a8d8-a2e4957bb673',\n",
" 'bucket_region': 'jp-tok-standard',\n",
" 'credentials': {'admin': {'access_key_id': '834b3358ebb945fb9ebbb4020cd2bf0e',\n",
" 'api_key': '2JONUuuPfYzZzPGzTp1J7dwwjNTpkOsyxdW5gx_vml3m',\n",
" 'secret_access_key': '1ed5b29fdd6c65b48ca72963b6177133ce51a7b23acdcaa5',\n",
" 'service_id': 'ServiceId-a2495f73-f36b-4fa1-9991-976f110c1a4f'},\n",
" 'editor': {'access_key_id': 'b56d445c54794369b2a4e0115e166605',\n",
" 'api_key': 'wcwCBLp8z4xpgnsEDeUCOZquAovrWhXu2wcF9Kz5Vhpe',\n",
" 'resource_key_crn': 'crn:v1:bluemix:public:cloud-object-storage:global:a/53f9f6400d0d44889534e8abcd2dfe39:d8fa8aee-cd61-4757-9543-a61f55971074::',\n",
" 'secret_access_key': '84b0b128f52e57c025e6517604a06212b8d19f0b349eeea3',\n",
" 'service_id': 'ServiceId-4e1f87ab-27bc-4654-b6ea-667a8640c7e0'},\n",
" 'viewer': {'access_key_id': '558109e942fb4b1eb020c881f04d8588',\n",
" 'api_key': 'zWS-VZ_d9GfkDt1XnCmWoOA6liYXNnGtrPwJt2fI0UI5',\n",
" 'resource_key_crn': 'crn:v1:bluemix:public:cloud-object-storage:global:a/53f9f6400d0d44889534e8abcd2dfe39:d8fa8aee-cd61-4757-9543-a61f55971074::',\n",
" 'secret_access_key': '3e2d27ab9d4041707cfa721daa638d1ad57f42ab8df94c09',\n",
" 'service_id': 'ServiceId-93177c88-86e2-470d-b5bf-3aed99d093a8'}},\n",
" 'endpoint_url': 'https://s3.jp-tok.cloud-object-storage.appdomain.cloud',\n",
" 'guid': 'd8fa8aee-cd61-4757-9543-a61f55971074',\n",
" 'resource_crn': 'crn:v1:bluemix:public:cloud-object-storage:global:a/53f9f6400d0d44889534e8abcd2dfe39:d8fa8aee-cd61-4757-9543-a61f55971074::'},\n",
" 'type': 'bmcos_object_storage'}},\n",
" 'metadata': {'created_at': '2022-10-31T10:33:07.575Z',\n",
" 'creator_id': 'IBMid-667000CZ2Y',\n",

```
    "    'id': 'aa24227a-9f01-493f-90e6-1b6132057fc6',\n",
    "    'updated_at': '2022-10-31T10:33:25.148Z',\n",
    "    'url': '/v2/spaces/aa24227a-9f01-493f-90e6-1b6132057fc6'}}]}"
   ]
  },
  "execution_count": 57,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "client.spaces.get_details()"
]
},
{
 "cell_type": "code",
 "execution_count": 58,
 "id": "36192134",
 "metadata": {},
 "outputs": [],
 "source": [
  "def guid_from_space_name(client,deploy):\n",
  "  space = client.spaces.get_details()\n",
  "  return (next(item for item in space['resources'] if item['entity']['name']==deploy)['metadata']['id'])"
 ]
},
{
 "cell_type": "code",
 "execution_count": 59,
 "id": "8db09ec8",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
```

```
   "output_type": "stream",
   "text": [
    "Space UID = aa24227a-9f01-493f-90e6-1b6132057fc6\n"
   ]
  }
 ],
 "source": [
  "space_uid = guid_from_space_name(client,'digitrecognition')\n",
  "print(\"Space UID = \" + space_uid)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 60,
 "id": "dd499fbd",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "'SUCCESS'"
    ]
   },
   "execution_count": 60,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "client.set.default_space(space_uid)"
 ]
},
{
 "cell_type": "code",
```

```
"execution_count": 61,
"id": "b1037d4f",
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "------------------------------ ------------------------------------ ----\n",
   "NAME                           ASSET_ID                             TYPE\n",
   "default_py3.6                  0062b8c9-8b7d-44a0-a9b9-46c416adcbd9 base\n",
   "kernel-spark3.2-scala2.12      020d69ce-7ac1-5e68-ac1a-31189867356a base\n",
   "pytorch-onnx_1.3-py3.7-edt     069ea134-3346-5748-b513-49120e15d288 base\n",
   "scikit-learn_0.20-py3.6        09c5a1d0-9c1e-4473-a344-eb7b665ff687 base\n",
   "spark-mllib_3.0-scala_2.12     09f4cff0-90a7-5899-b9ed-1ef348aebdee base\n",
   "pytorch-onnx_rt22.1-py3.9      0b848dd4-e681-5599-be41-b5f6fccc6471 base\n",
   "ai-function_0.1-py3.6          0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda base\n",
   "shiny-r3.6                     0e6e79df-875e-4f24-8ae9-62dcc2148306 base\n",
   "tensorflow_2.4-py3.7-horovod   1092590a-307d-563d-9b62-4eb7d64b3f22 base\n",
   "pytorch_1.1-py3.6              10ac12d6-6b30-4ccd-8392-3e922c096a92 base\n",
   "tensorflow_1.15-py3.6-ddl      111e41b3-de2d-5422-a4d6-bf776828c4b7 base\n",
   "runtime-22.1-py3.9             12b83a17-24d8-5082-900f-0ab31fbfd3cb base\n",
   "scikit-learn_0.22-py3.6        154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base\n",
   "default_r3.6                   1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base\n",
   "pytorch-onnx_1.3-py3.6         1bc6029a-cc97-56da-b8e0-39c3880dbbe7 base\n",
   "kernel-spark3.3-r3.6           1c9e5454-f216-59dd-a20e-474a5cdf5988 base\n",
   "pytorch-onnx_rt22.1-py3.9-edt  1d362186-7ad5-5b59-8b6c-9d0880bde37f base\n",
   "tensorflow_2.1-py3.6           1eb25b84-d6ed-5dde-b6a5-3fbdf1665666 base\n",
   "spark-mllib_3.2                20047f72-0a98-58c7-9ff5-a77b012eb8f5 base\n",
   "tensorflow_2.4-py3.8-horovod   217c16f6-178f-56bf-824a-b19f20564c49 base\n",
   "runtime-22.1-py3.9-cuda        26215f05-08c3-5a41-a1b0-da66306ce658 base\n",
   "do_py3.8                       295addb5-9ef9-547e-9bf4-92ae3563e720 base\n",
   "autoai-ts_3.8-py3.8            2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base\n",
   "tensorflow_1.15-py3.6          2b73a275-7cbf-420b-a912-eae7f436e0bc base\n",
```

"kernel-spark3.3-py3.9 2b7961e2-e3b1-5a8c-a491-482c8368839a base\n",

"pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1 base\n",

"spark-mllib_2.3 2e51f700-bca0-4b0d-88dc-5c6791338875 base\n",

"pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-dde874a8d67e base\n",

"spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base\n",

"spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base\n",

"xgboost_0.82-py3.6 39e31acd-5f30-41dc-ae44-60233c80306e base\n",

"pytorch-onnx_1.2-py3.6-edt 40589d0e-7019-4e28-8daa-fb03b6f4fe12 base\n",

"default_r36py38 41c247d3-45f8-5a71-b065-8580229facf0 base\n",

"autoai-ts_rt22.1-py3.9 4269d26e-07ba-5d40-8f66-2d495b0c71f7 base\n",

"autoai-obm_3.0 42b92e18-d9ab-567f-988a-4240ba1ed5f7 base\n",

"pmml-3.0_4.3 493bcb95-16f1-5bc5-bee8-81b8af80e9c7 base\n",

"spark-mllib_2.4-r_3.6 49403dff-92e9-4c87-a3d7-a42d0021c095 base\n",

"xgboost_0.90-py3.6 4ff8d6c2-1343-4c18-85e1-689c965304d3 base\n",

"pytorch-onnx_1.1-py3.6 50f95b2a-bc16-43bb-bc94-b0bed208c60b base\n",

"autoai-ts_3.9-py3.8 52c57136-80fa-572e-8728-a5e7cbb42cde base\n",

"spark-mllib_2.4-scala_2.11 55a70f99-7320-4be5-9fb9-9edb5a443af5 base\n",

"spark-mllib_3.0 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9 base\n",

"autoai-obm_2.0 5c2e37fa-80b8-5e77-840f-d912469614ee base\n",

"spss-modeler_18.1 5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b base\n",

"cuda-py3.8 5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e base\n",

"autoai-kb_3.1-py3.7 632d4b22-10aa-5180-88f0-f52dfb6444d7 base\n",

"pytorch-onnx_1.7-py3.8 634d3cdc-b562-5bf9-a2d4-ea90a478456b base\n",

"spark-mllib_2.3-r_3.6 6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c base\n",

"tensorflow_2.4-py3.7 65e171d7-72d1-55d9-8ebb-f813d620c9bb base\n",

"spss-modeler_18.2 687eddc9-028a-4117-b9dd-e57b36f1efa5 base\n",

"pytorch-onnx_1.2-py3.6 692a6a4d-2c4d-45ff-a1ed-b167ee55469a base\n",

"spark-mllib_2.3-scala_2.11 7963efe5-bbec-417e-92cf-0574e21b4e8d base\n",

"spark-mllib_2.4-py37 7abc992b-b685-532b-a122-a396a3cdbaab base\n",

"caffe_1.0-py3.6 7bb3dbe2-da6e-4145-918d-b6d84aa93b6b base\n",

"pytorch-onnx_1.7-py3.7 812c6631-42b7-5613-982b-02098e6c909c base\n",

"cuda-py3.6 82c79ece-4d12-40e6-8787-a7b9e0f62770 base\n",

"tensorflow_1.15-py3.6-horovod 8964680e-d5e4-5bb8-919b-8342c6c0dfd8 base\n",

"hybrid_0.1 8c1a58c6-62b5-4dc4-987a-df751c2756b6 base\n",

"pytorch-onnx_1.3-py3.7 8d5d8a87-a912-54cf-81ec-3914adaa988d base\n",

"caffe-ibm_1.0-py3.6 8d863266-7927-4d1e-97d7-56a7f4c0a19b base\n",

"spss-modeler_17.1 902d0051-84bd-4af6-ab6b-8f6aa6fdeabb base\n",

"do_12.10 9100fd72-8159-4eb9-8a0b-a87e12eefa36 base\n",

"do_py3.7 9447fa8b-2051-4d24-9eef-5acb0e3c59f8 base\n",

"spark-mllib_3.0-r_3.6 94bb6052-c837-589d-83f1-f4142f219e32 base\n",

"cuda-py3.7-opence 94e9652b-7f2d-59d5-ba5a-23a414ea488f base\n",

"nlp-py3.8 96e60351-99d4-5a1c-9cc0-473ac1b5a864 base\n",

"cuda-py3.7 9a44990c-1aa1-4c7d-baf8-c4099011741c base\n",

"hybrid_0.2 9b3f9040-9cee-4ead-8d7a-780600f542f7 base\n",

"spark-mllib_3.0-py38 9f7a8fc1-4d3c-5e65-ab90-41fa8de2d418 base\n",

"autoai-kb_3.3-py3.7 a545cca3-02df-5c61-9e88-998b09dc79af base\n",

"spark-mllib_3.0-py39 a6082a27-5acc-5163-b02c-6b96916eb5e0 base\n",

"runtime-22.1-py3.9-do a7e7dbf1-1d03-5544-994d-e5ec845ce99a base\n",

"default_py3.8 ab9e1b80-f2ce-592c-a7d2-4f2344f77194 base\n",

"tensorflow_rt22.1-py3.9 acd9c798-6974-5d2f-a657-ce06e986df4d base\n",

"kernel-spark3.2-py3.9 ad7033ee-794e-58cf-812e-a95f4b64b207 base\n",

"autoai-obm_2.0 with Spark 3.0 af10f35f-69fa-5d66-9bf5-acb58434263a base\n",

"default_py3.7_opence c2057dd4-f42c-5f77-a02f-72bdbd3282c9 base\n",

"tensorflow_2.1-py3.7 c4032338-2a40-500a-beef-b01ab2667e27 base\n",

"do_py3.7_opence cc8f8976-b74a-551a-bb66-6377f8d865b4 base\n",

"spark-mllib_3.3 d11f2434-4fc7-58b7-8a62-755da64fdaf8 base\n",

"autoai-kb_3.0-py3.6 d139f196-e04b-5d8b-9140-9a10ca1fa91a base\n",

"spark-mllib_3.0-py36 d82546d5-dd78-5fbb-9131-2ec309bc56ed base\n",

"autoai-kb_3.4-py3.8 da9b39c3-758c-5a4f-9cfd-457dd4d8c395 base\n",

"kernel-spark3.2-r3.6 db2fe4d6-d641-5d05-9972-73c654c60e0a base\n",

"autoai-kb_rt22.1-py3.9 db6afe93-665f-5910-b117-d879897404d9 base\n",

"tensorflow_rt22.1-py3.9-horovod dda170cc-ca67-5da7-9b7a-cf84c6987fae base\n",

"autoai-ts_1.0-py3.7 deef04f0-0c42-5147-9711-89f9904299db base\n",

"tensorflow_2.1-py3.7-horovod e384fce5-fdd1-53f8-bc71-11326c9c635f base\n",

"default_py3.7 e4429883-c883-42b6-87a8-f419d64088cd base\n",

"do_22.1 e51999ba-6452-5f1f-8287-17228b88b652 base\n",

"autoai-obm_3.2 eae86aab-da30-5229-a6a6-1d0d4e368983 base\n",

"do_20.1 f686cdd9-7904-5f9d-a732-01b0d6b10dc5 base\n",

```
   "scikit-learn_0.19-py3.6         f963fa9d-4bb7-5652-9c5d-8d9289ef6ad9  base\n",
   "tensorflow_2.4-py3.8            fe185c44-9a99-5425-986b-59bd1d2eda46  base\n",
   "------------------------------  ------------------------------------  ----\n"
  ]
 }
],
"source": [
 "client.software_specifications.list(limit=100)"
]
},
{
 "cell_type": "code",
 "execution_count": 62,
 "id": "841ad2ab",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "'acd9c798-6974-5d2f-a657-ce06e986df4d'"
    ]
   },
   "execution_count": 62,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "software_space_uid = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')\n",
  "software_space_uid"
 ]
},
{
 "cell_type": "code",
```

```
  "execution_count": 63,
  "id": "82c978e1",
  "metadata": {},
  "outputs": [],
  "source": [
   "model_details = client.repository.store_model(model='handwritten-digit-recognition-model_new.tgz',meta_props={\n",
   "    client.repository.ModelMetaNames.NAME:\"CNN Digit recognition model\",\n",
   "    client.repository.ModelMetaNames.TYPE:\"tensorflow_2.7\",\n",
   "    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid\n",
   "})"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 64,
  "id": "12700c5d",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "{'entity': {'hybrid_pipeline_software_specs': [],\n",
      "  'software_spec': {'id': 'acd9c798-6974-5d2f-a657-ce06e986df4d',\n",
      "   'name': 'tensorflow_rt22.1-py3.9'},\n",
      "  'type': 'tensorflow_2.7'},\n",
      " 'metadata': {'created_at': '2022-11-01T10:15:40.847Z',\n",
      "  'id': '97d463b1-45ee-47f7-b8af-aed338794ce1',\n",
      "  'modified_at': '2022-11-01T10:15:44.197Z',\n",
      "  'name': 'CNN Digit recognition model',\n",
      "  'owner': 'IBMid-667000CZ2Y',\n",
      "  'resource_key': '84636ddb-9fa8-47e4-8fa4-3c36731e2fe6',\n",
      "  'space_id': 'aa24227a-9f01-493f-90e6-1b6132057fc6'},\n",
      " 'system': {'warnings': []}}"
```

```
   ]
  },
  "execution_count": 64,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "model_details"
]
},
{
"cell_type": "code",
"execution_count": 65,
"id": "edd79f04",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "'97d463b1-45ee-47f7-b8af-aed338794ce1'"
   ]
  },
  "execution_count": 65,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "model_id = client.repository.get_model_id(model_details)\n",
 "model_id"
]
},
{
```

```
 "cell_type": "code",
 "execution_count": 66,
 "id": "0a26ad43",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Successfully saved model content to file: 'DigitRecog_IBM_model.tar.gz'\n"
   ]
  },
  {
   "data": {
    "text/plain": [
     "'/home/wsuser/work/models/DigitRecog_IBM_model.tar.gz'"
    ]
   },
   "execution_count": 66,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "client.repository.download(model_id,'DigitRecog_IBM_model.tar.gz')"
 ]
},
{
 "cell_type": "code",
 "execution_count": 67,
 "id": "e3b67528",
 "metadata": {},
 "outputs": [
  {
```

```
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "DigitRecog_IBM_model.tar.gz                mnistCNN.h5\r\n",
   "handwritten-digit-recognition-model_new.tgz\r\n"
  ]
 }
],
"source": [
 "ls"
]
},
{
 "cell_type": "markdown",
 "id": "62f82a21",
 "metadata": {},
 "source": [
  "# TEST MODEL"
 ]
},
{
 "cell_type": "code",
 "execution_count": 286,
 "id": "b3bc5a3d",
 "metadata": {},
 "outputs": [],
 "source": [
  "from tensorflow.keras.models import load_model\n",
  "from keras.preprocessing import image\n",
  "from PIL import Image\n",
  "import numpy as np"
 ]
},
{
```

```
 "cell_type": "code",
 "execution_count": 287,
 "id": "0e416ac2",
 "metadata": {},
 "outputs": [],
 "source": [
  "model = load_model(\"mnistCNN.h5\")"
 ]
},
{
 "cell_type": "code",
 "execution_count": 288,
 "id": "d9d565ad",
 "metadata": {},
 "outputs": [],
 "source": [
  "\n",
  "import os, types\n",
  "import pandas as pd\n",
  "from botocore.client import Config\n",
  "import ibm_boto3\n",
  "\n",
  "def __iter__(self): return 0\n",
  "\n",
  "# @hidden_cell\n",
  "# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.\n",
  "# You might want to remove those credentials before you share the notebook.\n",
  "cos_client = ibm_boto3.client(service_name='s3',\n",
  "    ibm_api_key_id='is_QZGPyU8oxZr3W-td-LCHXS3QPMaWArILi18FdSyGT',\n",
  "    ibm_auth_endpoint=\"https://iam.cloud.ibm.com/oidc/token\",\n",
  "    config=Config(signature_version='oauth'),\n",
  "    endpoint_url='https://s3.private.ap.cloud-object-storage.appdomain.cloud')\n",
  "\n",
  "bucket = 'handwrittenimagerecognition-donotdelete-pr-8tlrnykut46vpi'\n",
```

"object_key = 'mnist-dataset-1024x424 (2).png'\n",
"\n",
"streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']\n",
"\n",
"# Your data file was loaded into a botocore.response.StreamingBody object.\n",
"# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.\n",
"# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/\n",
"# pandas documentation: http://pandas.pydata.org/\n"
]
},
{
"cell_type": "code",
"execution_count": 289,
"id": "0af4c026",
"metadata": {},
"outputs": [],
"source": [
"img = Image.open(streaming_body_1).convert(\"L\") # convert image to monochrome\n",
"img = img.resize((28,28)) # resizing of input image"
]
},
{
"cell_type": "code",
"execution_count": 290,
"id": "758a186b",
"metadata": {},
"outputs": [
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAABwAAAAcCAAAAABXZoBIAAABUUlEQVR4nGNgGGSAEcFklrW+ftfCjeHH2vP/ISIs
CEmhovB7V+zUGBi4H77FMCTlzf//v3+f2v3+TRYbupz6tv9///4/bao28ftNT0ZUOcbKH////X9fw8AgPvdTBROqJP+8/9
/+fmgVZmBgSHh5VhJFjsn18ZcpByYJMDAwMMQ8/5SCYq7swv837J3EGRgYGBgSXv6eyYLsFTV3hkPXX0HY9x4IQOy
E2cwnzvDiPZR9aP4nVMn/DC9u/IPZcforK4rkP4ZHV/7CJFl+HvuHJMmqwPD2C9x1f67uRJYUcPl/7jFMjtP91DNkY38+

YxThgkm6Rv/4jexNxuj/b6pEWVhZWQXcZlzerge1G+rYd1+Ei7SeMDH8k7URO9VymQEFiBWe+f3/37//f748Xm0MCz
t4GLKppVmz/Gf8uufg1Zf/0SUZGJgYGRgYGP79Z6A1AABiMn1gl3jf8wAAAABJRU5ErkJggg==\n",

"text/plain": [

"<PIL.Image.Image image mode=L size=28x28 at 0x7EFFC445F430>"

]

},

"execution_count": 290,

"metadata": {},

"output_type": "execute_result"

}

],

"source": [

"img"

]

},

{

"cell_type": "code",

"execution_count": 291,

"id": "ad56c441",

"metadata": {},

"outputs": [],

"source": [

"im2arr = np.array(img) #converting to image\n",

"im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement"

]

},

{

"cell_type": "code",

"execution_count": 292,

"id": "4ffcdfc3",

"metadata": {},

"outputs": [

{

"name": "stdout",

```
   "output_type": "stream",
   "text": [
    "[[1.0000000e+00 5.3912803e-17 3.9648812e-11 2.0051219e-16 5.1053910e-18\n",
    "  2.9315760e-12 7.0849349e-13 2.0999634e-16 2.9204243e-09 7.4729778e-11]]\n"
   ]
  }
 ],
 "source": [
  "pred = model.predict(im2arr)\n",
  "print(pred)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 293,
 "id": "81258a25",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "[0]\n"
   ]
  }
 ],
 "source": [
  "print(np.argmax(pred, axis=1)) #printing our Labels"
 ]
}
],
"metadata": {
 "kernelspec": {
  "display_name": "Python 3.9",
```

```
  "language": "python",
  "name": "python3"
 },
 "language_info": {
  "codemirror_mode": {
   "name": "ipython",
   "version": 3
  },
  "file_extension": ".py",
  "mimetype": "text/x-python",
  "name": "python",
  "nbconvert_exporter": "python",
  "pygments_lexer": "ipython3",
  "version": "3.9.13"
 }
},
"nbformat": 4,
"nbformat_minor": 5
}
```