Name: Chinmay Arvind
Batch Code: LISUM18
Submission Date: March 12, 2023

Week 4: Flask Deployment
Stage 1: Model building with Python, TensorFlow, and MNIST dataset

In [2]:
```
# Source code adapted from: https://docs.ray.io/en/latest/serve/tutorials/serve-ml-models.html
# Importing necessary libraries
import pickle
import tensorflow as tf
```

In [3]:
```
# Load MNIST dataset
dataset = tf.keras.datasets.mnist
(training_X_data, training_Y_data), (Validation_X_data, Validation_Y_data) = dataset.load_data()

# Preprocessing the data by normalizing it to a range between 0 and 1
training_X_data = training_X_data / 255.0
Validation_X_data = Validation_X_data / 255.0

# Defining the Neural Network's architecture
neural_net_model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(5, activation='softmax')
])

# Compiling the model with the poisson loss and training the model
neural_net_model.compile(optimizer='sgd', loss='poisson', metrics=['accuracy'])
neural_net_model.fit(training_X_data, training_Y_data, epochs=20)

# Running an evaluation and checking how well the model performs on the validation data
neural_net_model.evaluate(Validation_X_data, Validation_Y_data)

# Saving the model into a pickle file by writing to it
with open('mnist_predictive_model.pkl', 'wb') as filename:
    pickle.dump(neural_net_model, filename)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 1s 0us/step
Epoch 1/20

2023-03-12 15:12:31.292994: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operat
ions:  SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.

1875/1875 [==============================] - 3s 2ms/step - loss: 7.3794 - accuracy: 0.0879
Epoch 2/20
1875/1875 [==============================] - 3s 2ms/step - loss: 7.3725 - accuracy: 0.0951
Epoch 3/20
```

Stage 2: Deployment of model to Flask

```
Keras model archive saving:
File Name                                                 Modified             Size
config.json                                         2023-03-12 15:35:18         1956
metadata.json                                       2023-03-12 15:35:18           64
variables.h5                                        2023-03-12 15:35:18      2168656
```

Order of execution: run flask_deployment.py, app.py, and then requests.py

Send a request using Postman:
1. Create a new request in Postman
2. Select the HTTP method as 'POST'
3. Enter the URL as "http://localhost:8080/predict"
4. In the "Body" tab, select the "raw" option and choose "JSON" as the data type.
5. Send a json array of numbers between 0 and 1, and the model should show a prediction.