**ICP – 3**

**Tejaswini Pasupuleti**

**700755556**

**Github link: https://github.com/Tejaswinipasupuleti45/ICP\_3**

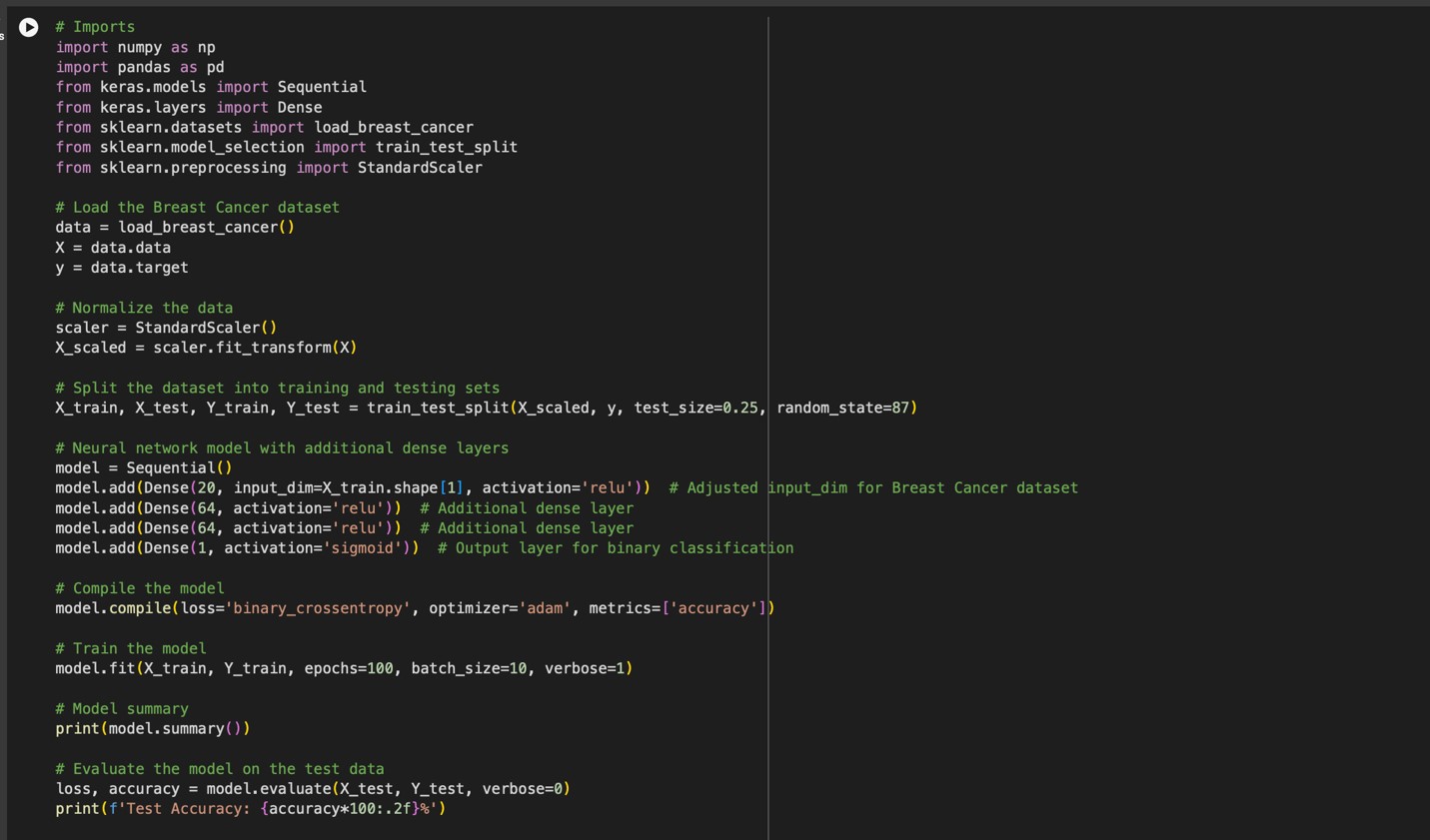
**Video Link: h-ps://drive.google.com/file/d/15UquAR5hBpA633bbkjgOfLn-5tJBJvVL/view?usp=sharing**

**Q**

**-**

**1:**

**CODE:**



**EXPLANATION:**

1. Dataset Loading: It loads the Breast Cancer dataset from scikit-learn, which contains features for breast cancer diagnos?c data and binary labels indica?ng malignancy.

1. Data Preprocessing: The features are normalized using `StandardScaler` to scale the data, ensuring that the neural network model receives input data within a manageable range, improving training stability and performance.

1. Dataset SpliJng: The normalized data is split into training and tes?ng sets, with 75% used for training and 25% for tes?ng, facilitated by the `train\_test\_split` method from scikit-learn.

1. Model Construc?on: A `Sequen?al` model is defined with four layers:

* An input dense layer with 20 neurons and ReLU ac?va?on, adjusted to match the number of features in the Breast Cancer dataset.
* Two addi?onal dense layers, each with 64 neurons and ReLU ac?va?on, to increase the model's capacity to learn complex pa[erns.
* An output layer with a single neuron and sigmoid ac?va?on to predict the binary outcome (malignant or benign).

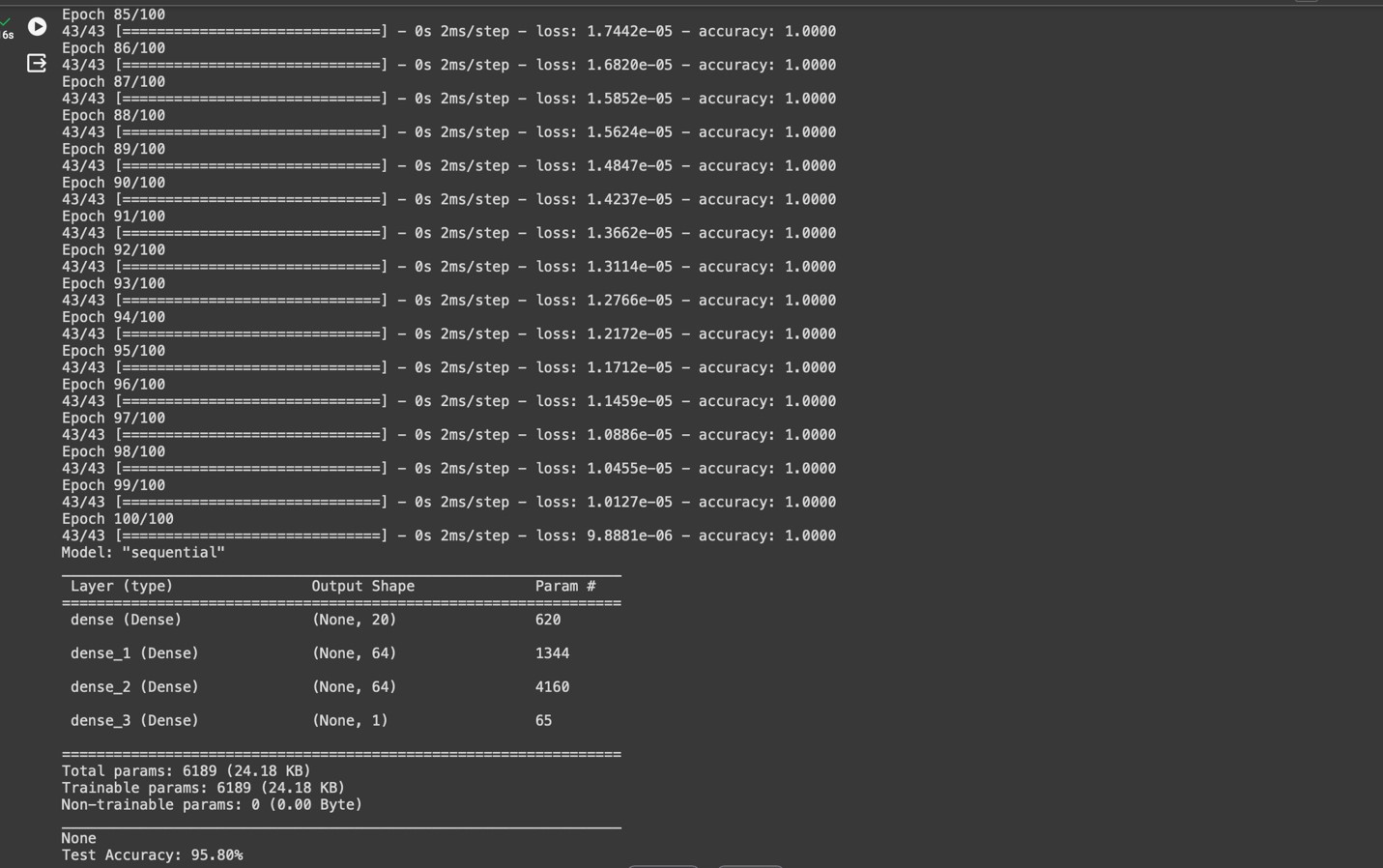
1. Model Compila?on: The model is compiled with the Adam op?mizer, binary crossentropy as the loss func?on, and accuracy as the metric to evaluate performance.

1. Model Training: The model is trained for 100 epochs with a batch size of 10 on the normalized and split training data, with verbosity set to 1 to display training progress.

1. Evalua?on: A\_er training, the model's performance is evaluated on the test set, and the test accuracy is printed, indica?ng how well the model can predict breast cancer malignancy.

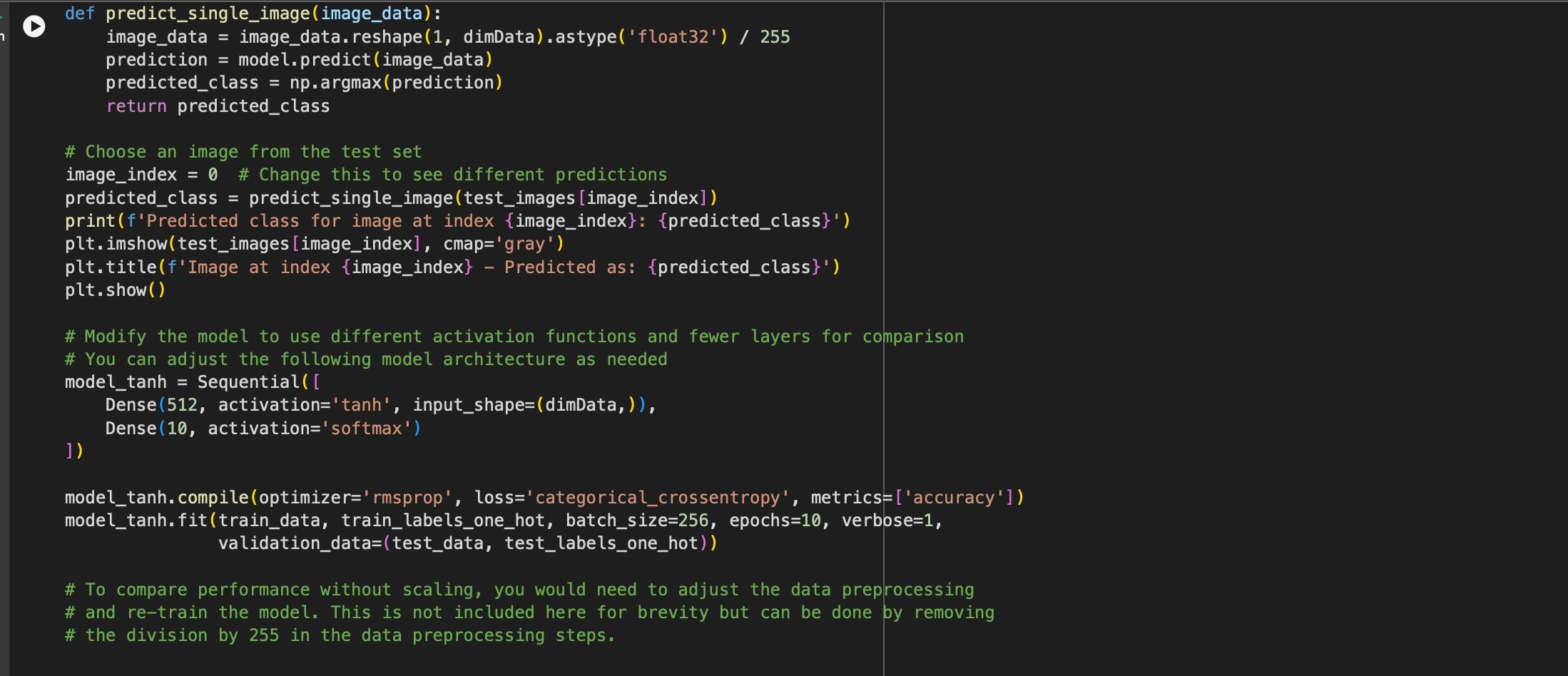
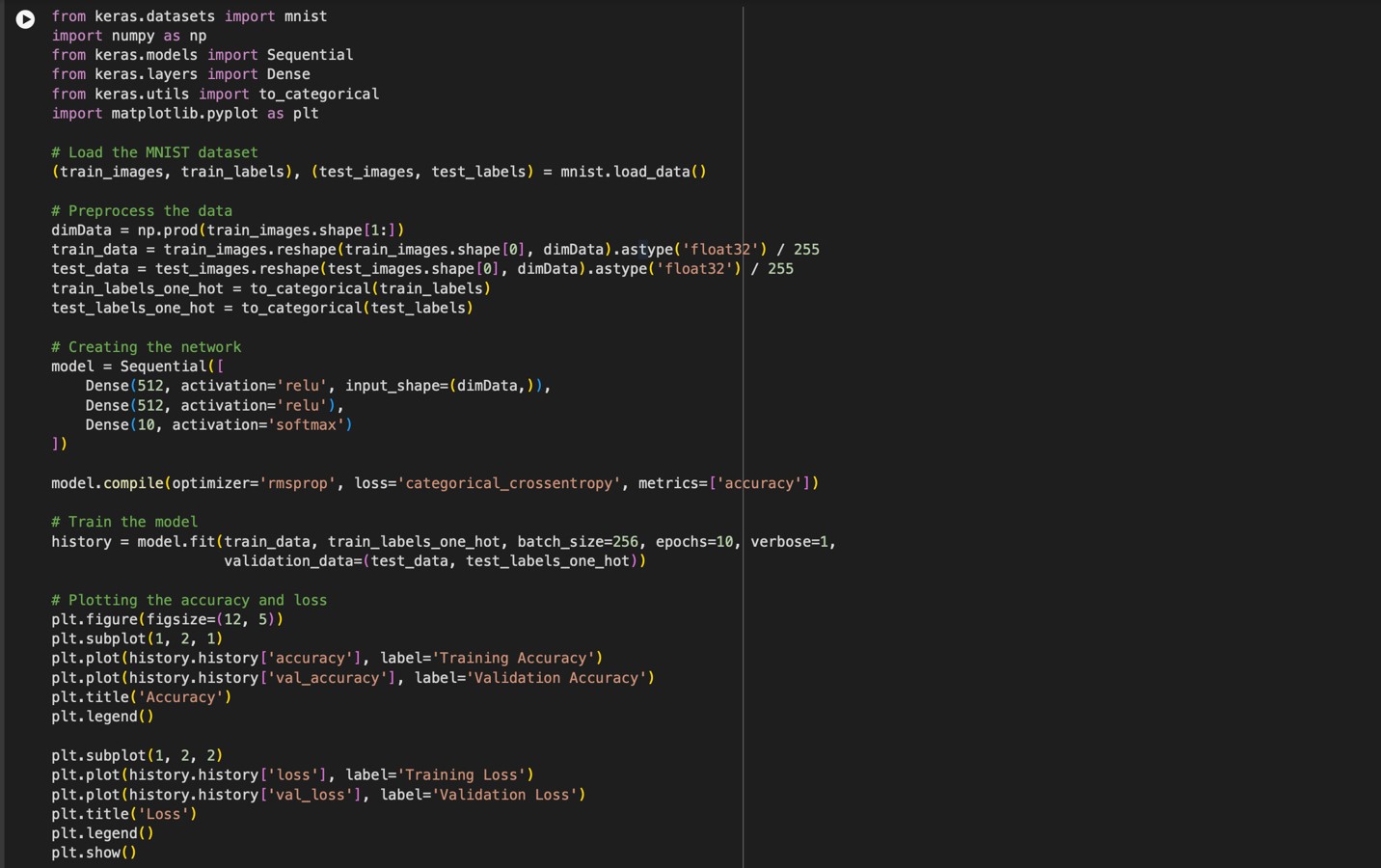
1. Summary Output: Finally, a summary of the model's architecture is printed, detailing the configura?on of each layer, including the number of parameters in the model.

**OUTPUT:**



**Q-2:**

**CODE:**



**EXPLANATION:**

The provided script is a comprehensive Python program for training a neural network model on the MNIST dataset for handwri[en digit recogni?on. It highlights key steps in data preprocessing, model training, evalua?on, and experimenta?on with model architecture. Here's a summary in a few points:

1. MNIST Dataset: The script loads the MNIST dataset, which consists of 28x28 pixel grayscale images of handwri[en digits (0 through 9) and their associated labels.

1. Data Preprocessing: The images are fla[ened from a 2D 28x28 format to a 1D 784 vector and normalized so that pixel values are in the range [0, 1]. Labels are converted to one-hot encoded vectors for classifica?on.

1. Model Architecture: A `Sequen?al` model with two dense layers of 512 neurons each (using ReLU ac?va?on), followed by a so\_max output layer for classifying the digits into 10 categories, is defined and compiled with RMSprop op?mizer and categorical crossentropy loss.

1. Training and Valida?on: The model is trained for 10 epochs with a batch size of 256, using both training and valida?on data to monitor performance and avoid overfiJng.

1. Performance Visualiza?on: Training and valida?on accuracy and loss are plo[ed using matplotlib to visually assess the model's learning progress over epochs.

1. Predic?on on Test Data: The script includes a func?on to predict the digit class for a single image from the test set, demonstra?ng the model's inferencing capability.

1. Model Experimenta?on: A variant of the model using tanh ac?va?on and a simplified architecture (one hidden layer) is trained to explore the impact of different network configura?ons on performance.

1. Note on Scaling: A comment men?ons the significance of scaling image pixel values for model performance, sugges?ng an experiment to compare results with and without this preprocessing step.

This script serves as a complete workflow for neural network training on image data, encompassing data handling, model building, training, evalua?on, and experimenta?on with architecture modifica?ons.

**OUTPUT:**

