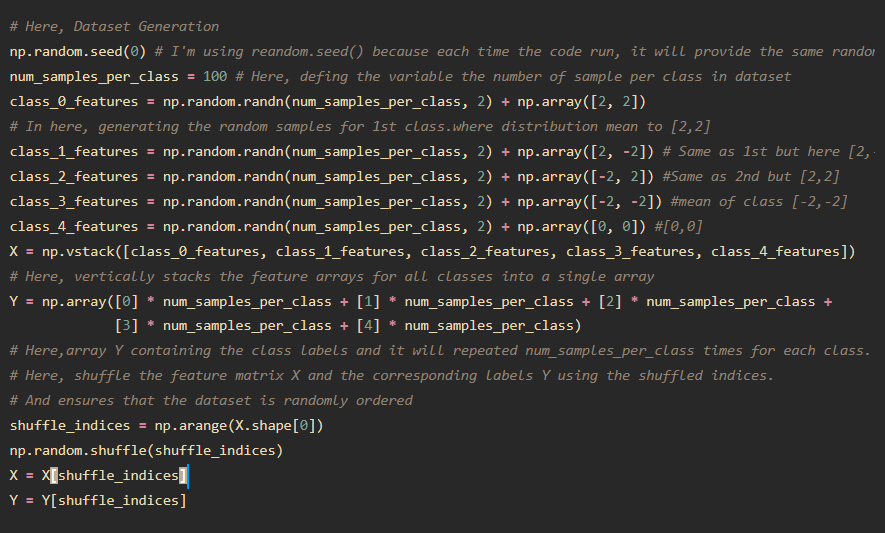
**Assignment: Implementation of a Three Hidden Layer Neural Network for Multi-Class Classification**

**Introduction:**

This assignment focuses on implementing a three hidden layer neural network for multi-class classification. The task involves modifying the provided codebase to accommodate the multi-class classification problem and evaluating the model's performance.

**Dataset Generation:**

A synthetic dataset consists of five distinct classes, with input features suitable for training a neural network.

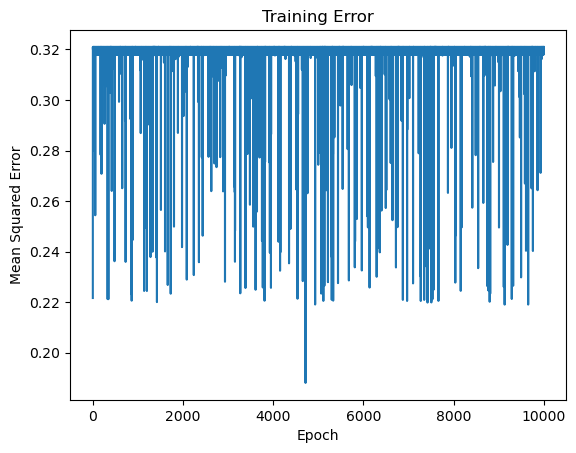


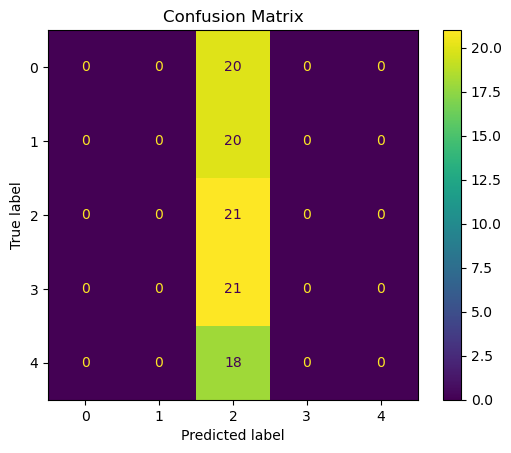
**Results and Analysis:**

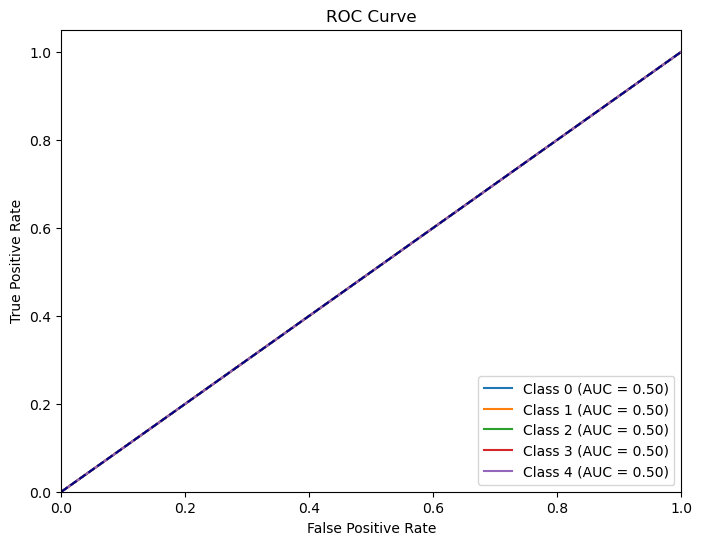
The performance of the neural network model was evaluated using different configurations, each varying in the number of hidden layers and learning rate.

**Configuration 1:**

* Hidden Layers: (10, 10, 10)
* Learning Rate: 0.2
* Accuracy: 0.21
* Precision: 0.0441
* Recall: 0.21
* F1 Score: 0.0729

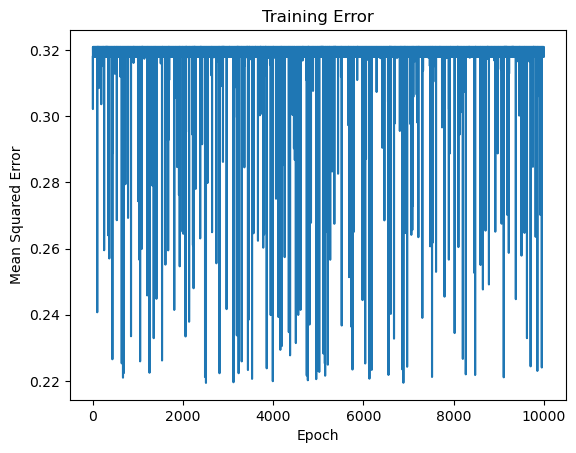


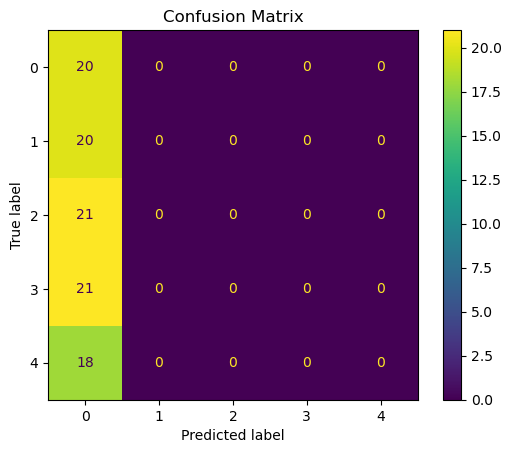


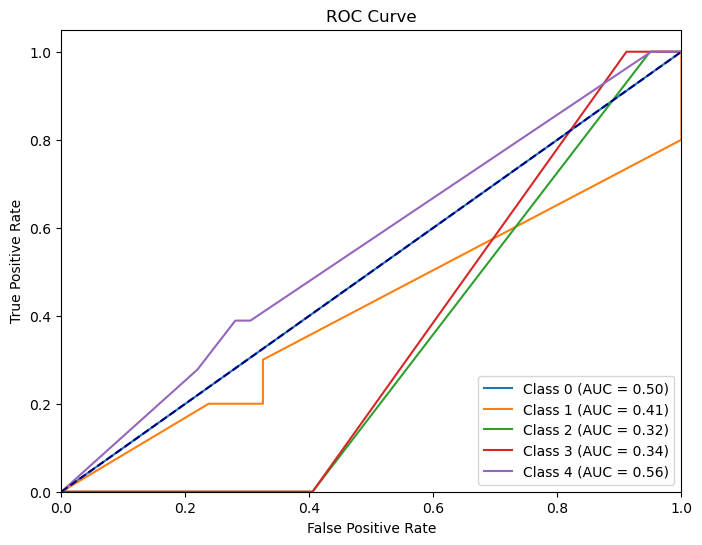


**Configuration 2:**

* Hidden Layers: (20, 20, 20)
* Learning Rate: 0.1
* Accuracy: 0.2
* Precision: 0.04
* Recall: 0.2
* F1 Score: 0.0667

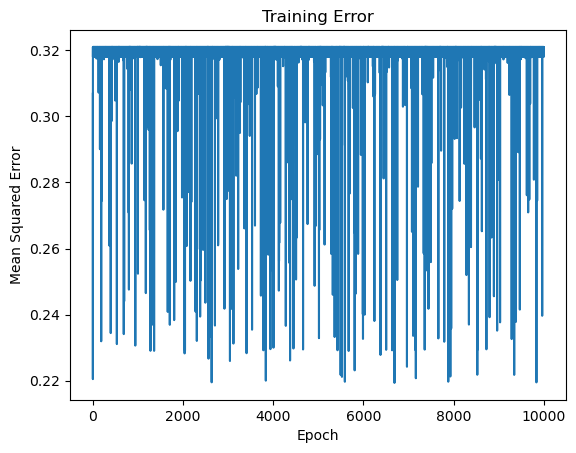


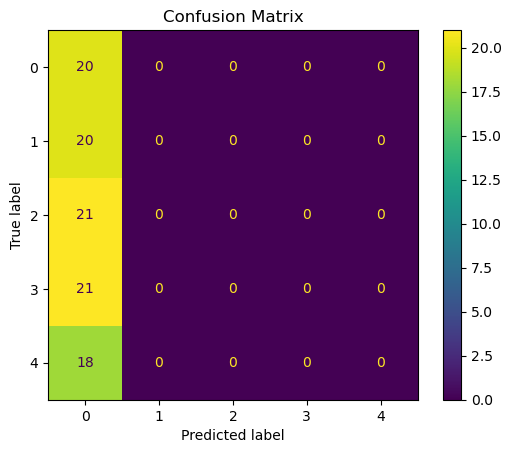


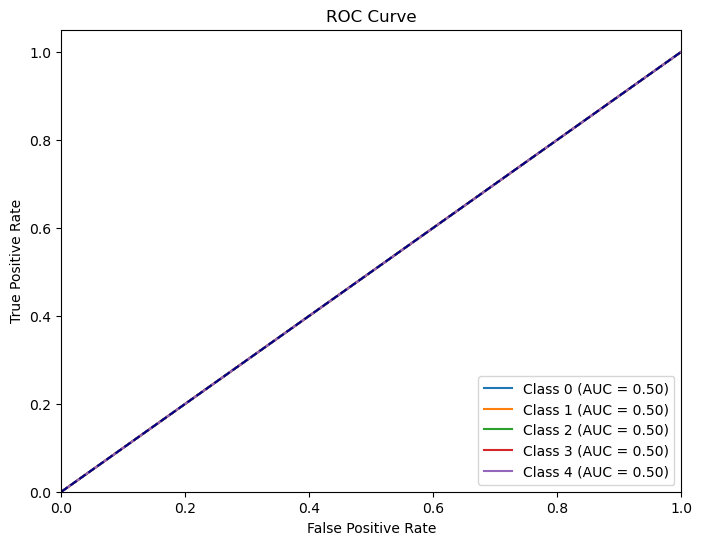


**Configuration 3:**

* Hidden Layers: (5, 5, 5)
* Learning Rate: 0.3
* Accuracy: 0.2
* Precision: 0.04
* Recall: 0.2
* F1 Score: 0.0667







After observation, the highest accuracy achieved was 0.21 with Configuration 1. However, the precision, recall, and F1 scores are low across all configurations, indicating deficient performance in correctly identifying positive samples. Configuration 1 with three hidden layers of 10 neurons each and a learning rate of 0.2 shows slightly better performance compared to other configurations.

**Challenges Encountered and Lessons Learned:**

Model performance fell below expectations, emphasizing the complexity of multi-class classification. Tuning hyperparameters, dataset complexity, and result interpretation were challenging but underscored the importance of experimentation, data quality, and thorough evaluation.

**Potential Improvements and Further Experiments:**

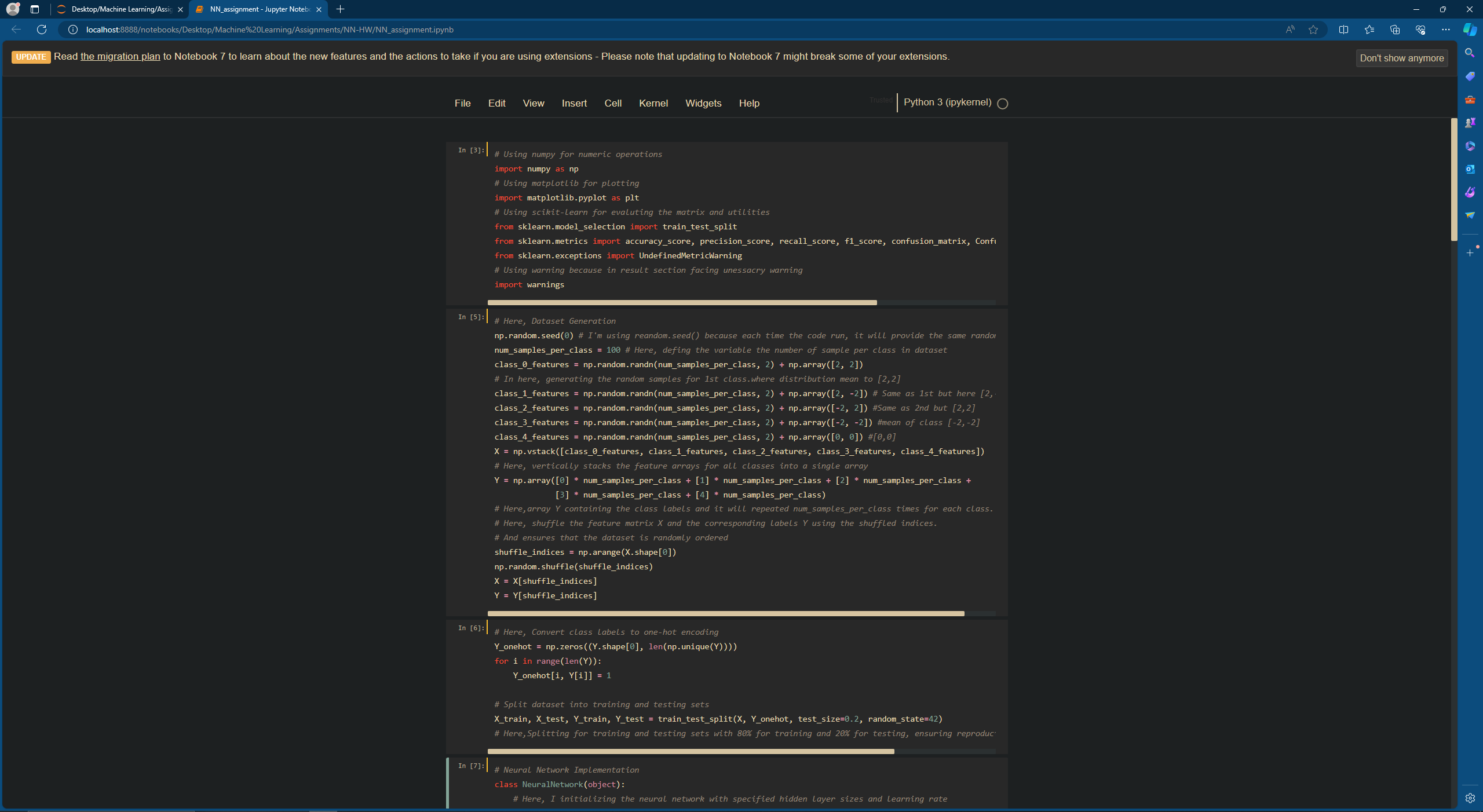
Explore data augmentation, advanced architectures like CNNs or RNNs, and automated hyperparameter optimization. Incorporate regularization techniques and ensemble

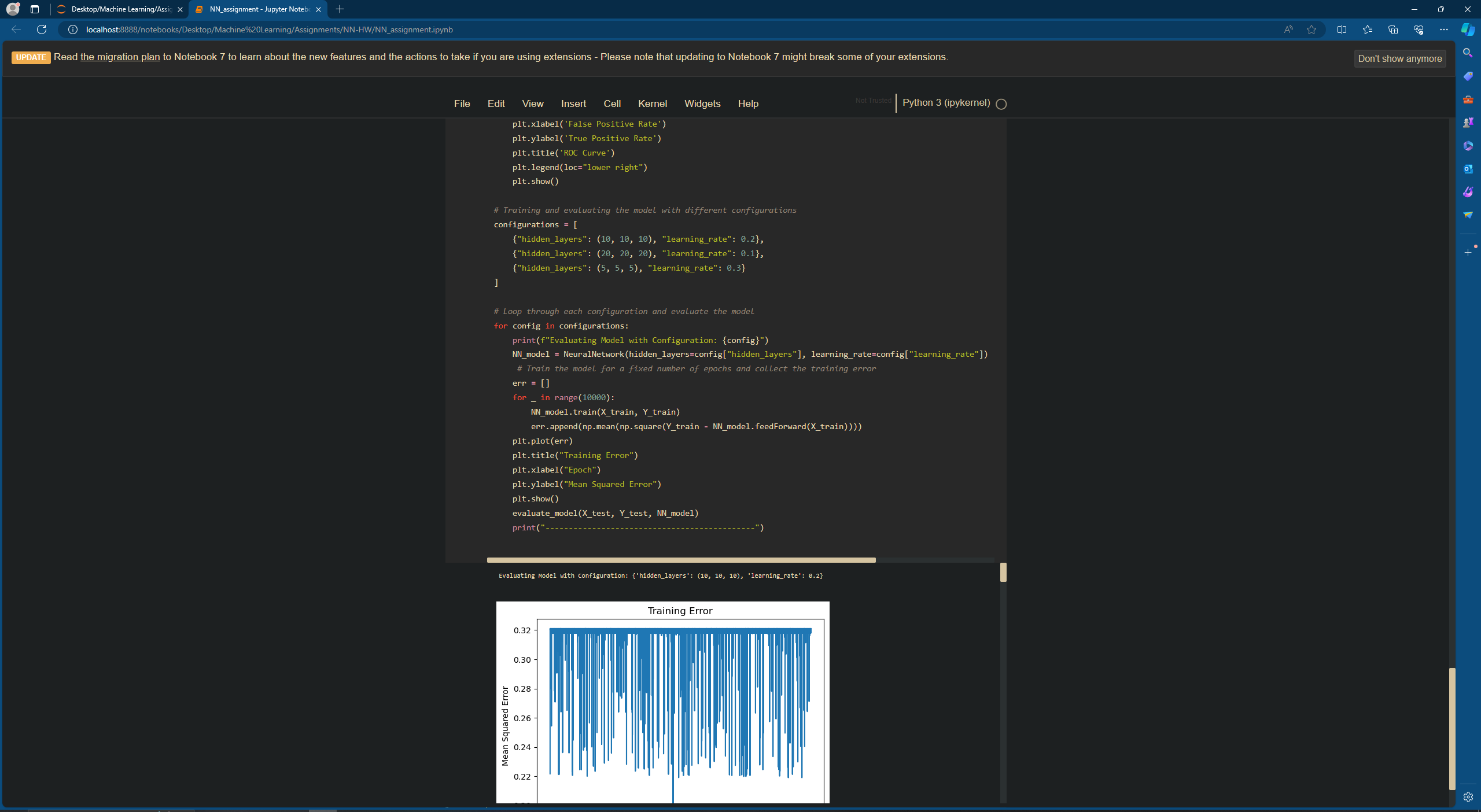
methods for improved model robustness and performance. Continued refinement and exploration are vital for advancing multi-class classification models.

**Installation and Run:**

It runs perfectly in anaconda (Jupiter Notebook) and VS Code with Jupiter extension and other modules are installed.

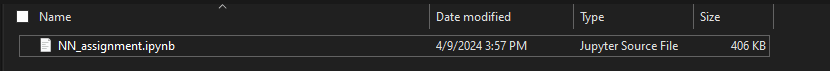
In Jupiter Notebook, [I write the code in Jupiter]



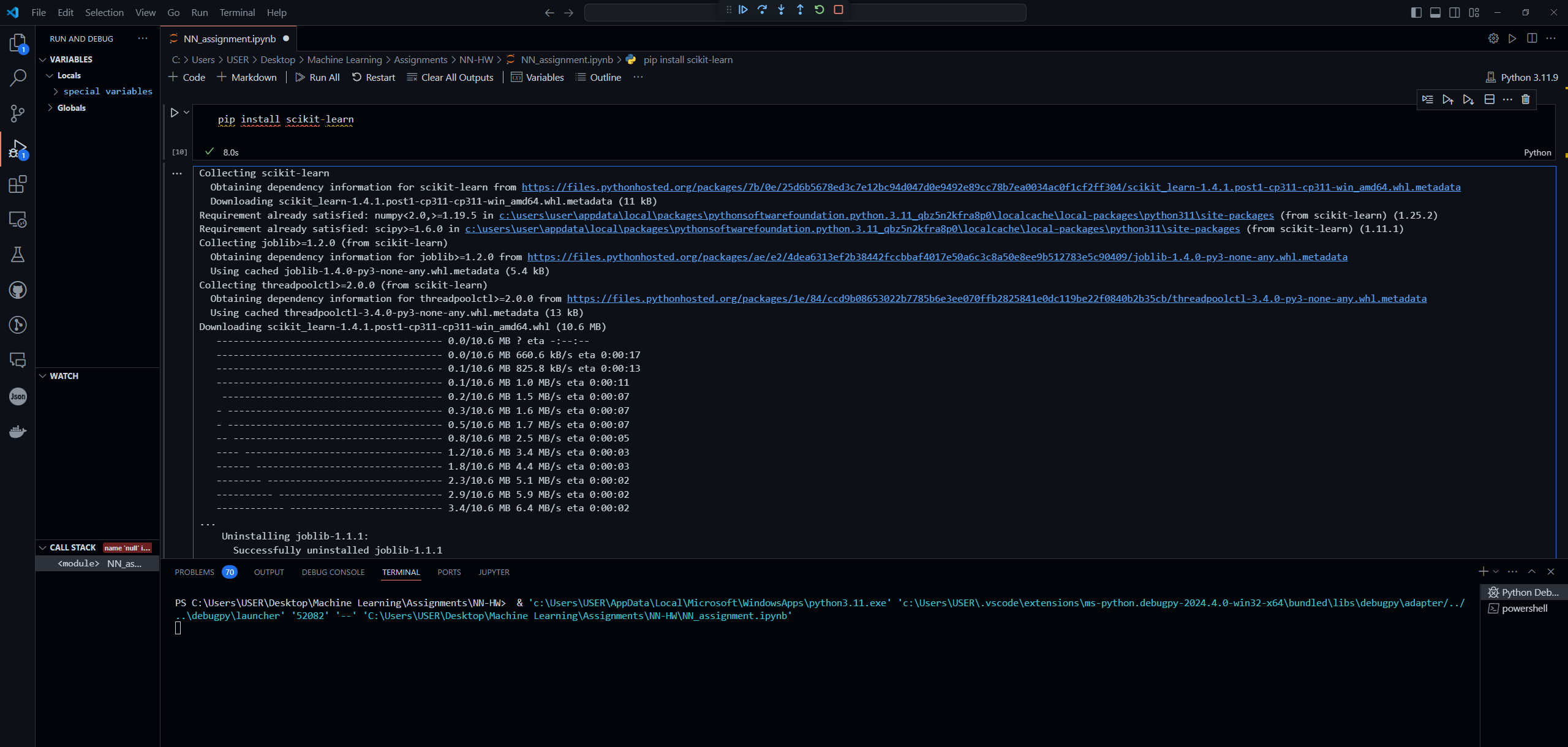


In VS Code,

1. Click and open the file



1. Run the code



1. It will show the results

