Saqlain Mushtaq

Saqlain.mushtaqbalti@gmail.com

Open Ended Lab

Machine Learning

**THE UNIVERSITY OF AZAD JAMMU AND KASHMIR**



***Name: Saqlain Mushtaq***

***Roll No: 2022-SE-33***

***Course code: SE-3105***

***Course Title: Machine Learning***

***Semester: 5th***

***Session: 2022-2026***

***Submitted To: Engr. Muhammad Awais***

***Open Ended Lab***

**Department Of Software Engineering**

**Classification of MNIST Handwritten Digits using Logistic Regression, and Artificial Neural Networks:**

**Abstract:**This report presents a comparative study on the classification of MNIST handwritten digits using two different machine learning models: Logistic Regression, and Artificial Neural Networks (ANN). Initially, the models are trained using mnist\_train.csv and tested on mnist\_test.csv. Hyperparameter tuning is performed, after which mnist\_train.csv is further split into training and testing sets for additional evaluation. Finally, the models' performances are compared using accuracy metrics and confusion matrices.

**Notebook source:**The complete implementation of this study, including data preprocessing, model training, hyperparameter tuning, and performance evaluation, is available in the Jupyter Notebook:

**Introduction:**Handwritten digit classification is a fundamental task in machine learning, often used as a benchmark for evaluating classification models. This study aims to implement and compare Logistic Regression, and ANN models to classify MNIST digits effectively.

**Dataset Description:**The MNIST dataset consists of grayscale images of handwritten digits (0-9). The dataset used includes:

1. mnist\_train.csv: Training dataset
2. mnist\_test.csv: Testing dataset

Each image is represented as a 28x28 pixel grid, flattened into a 784-dimensional feature vector.

**Methodology:**

**1. Data Preprocessing**The dataset is loaded using pandas. Features are standardized using StandardScaler or MinMaxScaler.

1. The dataset is shuffled to remove ordering biases.
2. Training and testing sets are created using train\_test\_split ().

**2. Model Implementation**

**(a) Initial Training and Testing**

1. Models are first trained using mnist\_train.csv and tested on mnist\_test.csv.

**(b) Hyperparameter Tuning**

1. GridSearchCV is used for Logistic Regression.
2. ANN model is tuned by adjusting the number of hidden layers and neurons.

**(c) Splitting mnist\_train.csv for Further Evaluation**

1. After hyperparameter tuning, mnist\_train.csv is split into new training and testing subsets.
2. Models are re-trained and tested on these new subsets.

**(d) Final Evaluation and Comparison**

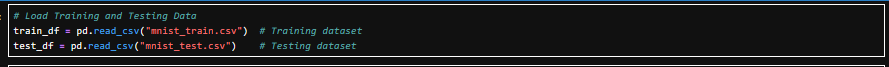
1. Model performances are compared using accuracy scores.
2. Confusion matrices are generated to analyze classification errors.

**Results and Discussion:**

1. **Initial Training** (on mnist\_train.csv → mnist\_test.csv): Achieved an accuracy.
2. **After Hyperparameter Tuning**: Improved accuracy.
3. **Training on Split mnist\_train.csv**: Final accuracy observed.
4. Confusion matrices illustrate misclassification patterns.

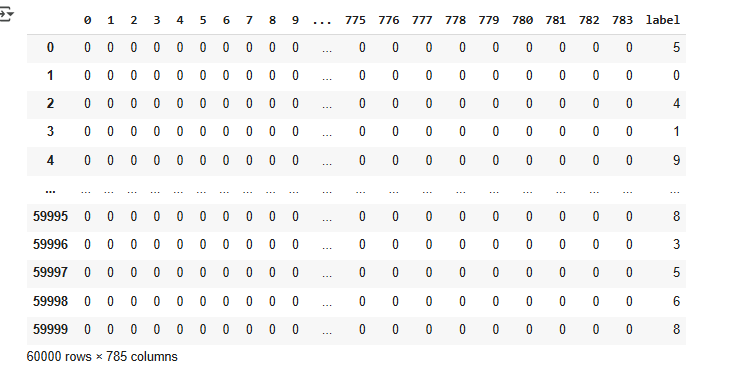
**Steps to perform the preprocessing, training and testing**

**Step 1:**load the dataset for further processing.



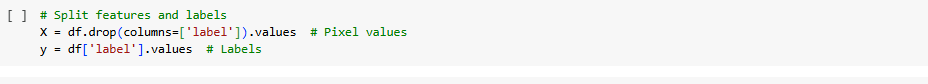
**Step 2:**print the first few rows of dataset for analysis.

**Output:**



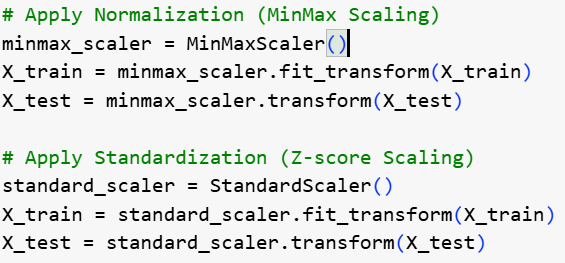
**Step 3:**This code split features (X) and labels (y) from training (train\_df) and testing (test\_df) datasets. It selects all columns except the last for features and the last column for labels using .iloc.

A computer code with text

AI-generated content may be incorrect.

**Step 4:**

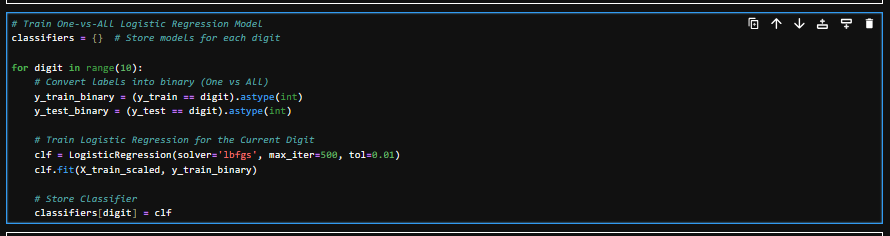
The code applies Min-Max Scaling to normalize features between 0 and 1 and Standardization to transform features to have mean 0 and standard deviation 1. fit\_transform(X\_train) fits and scales training data, while transform(X\_test) applies the same scaling to test data.

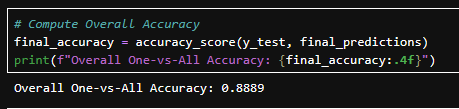


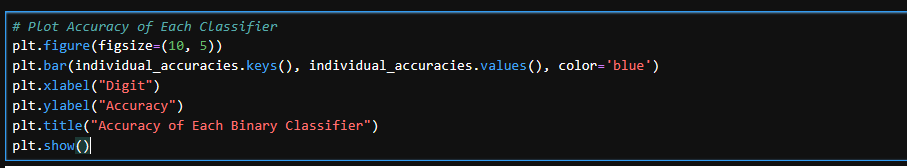
**Step 5:**

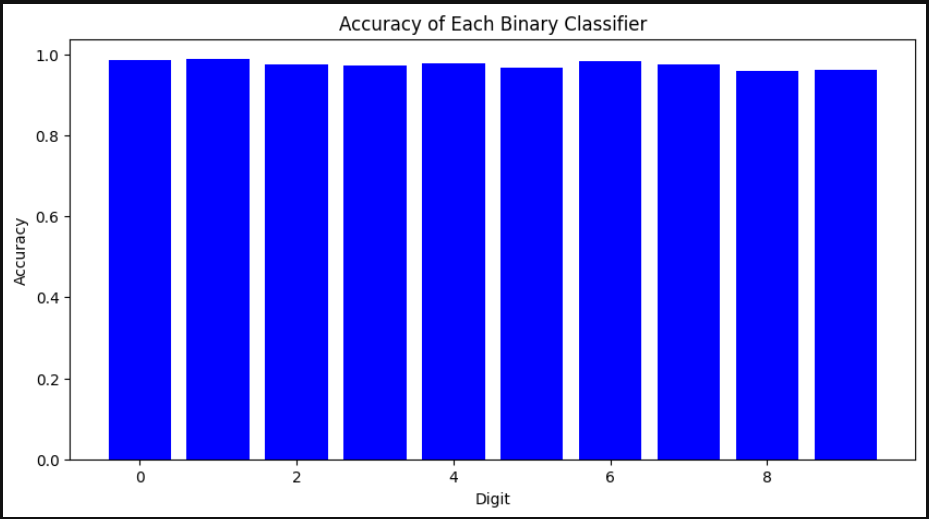
**Train the models using train\_mnsit.csv and test model using test\_mnsit.csv**

**Logistic Regression:**

Logistic Regression, which had **0.889%** accuracy on MNIST. KNN showed better performance in classification but may be slower on large datasets.

**Output**:

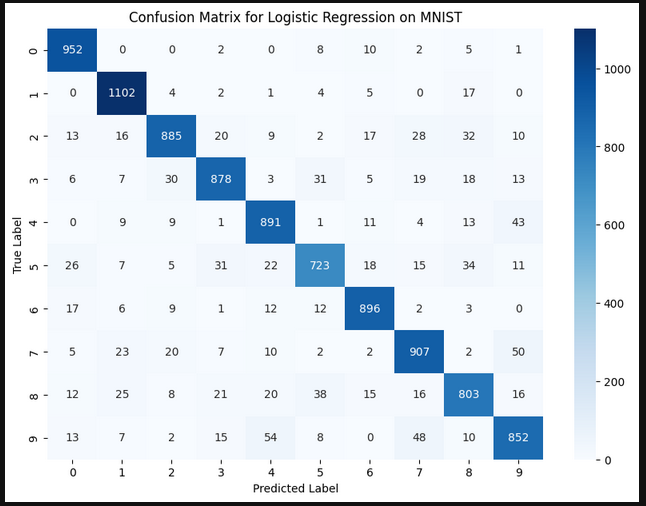
**Accuracy of Each Classifier:**

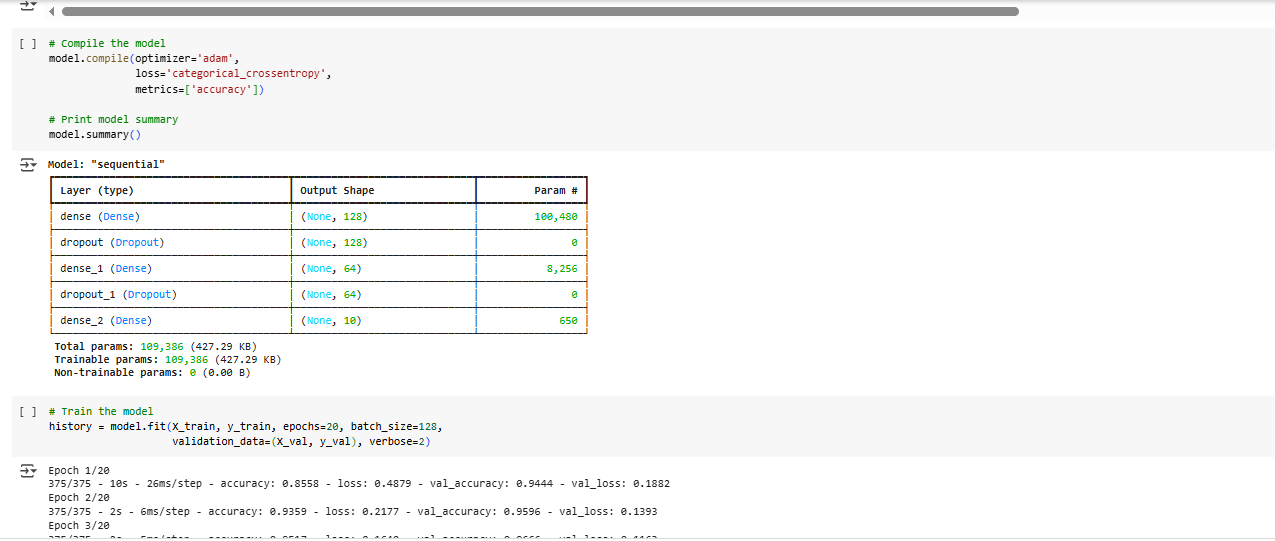
**Output:**

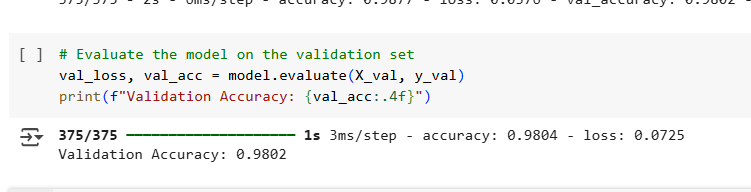
A black screen with colorful text

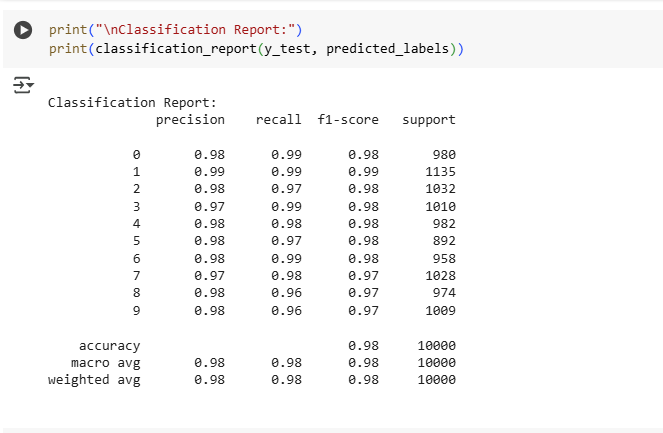
AI-generated content may be incorrect.**confusion matrix:**

**output:**



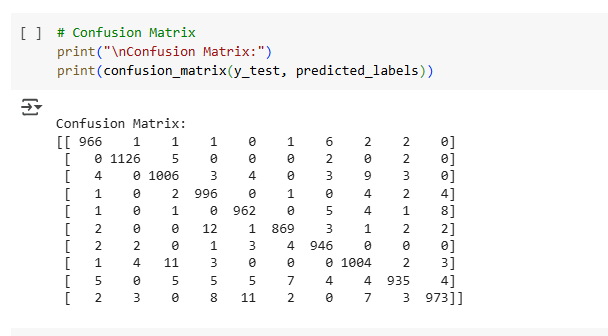
**ANN MODEL:**The logistic regression model achieved a Validation Accuracy: 0.9802%, while the artificial neural network (ANN) Testing an accuracy of 0.9783%, indicating that ANN is effective for the MNIST classification task.

**Output:**

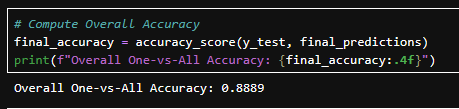


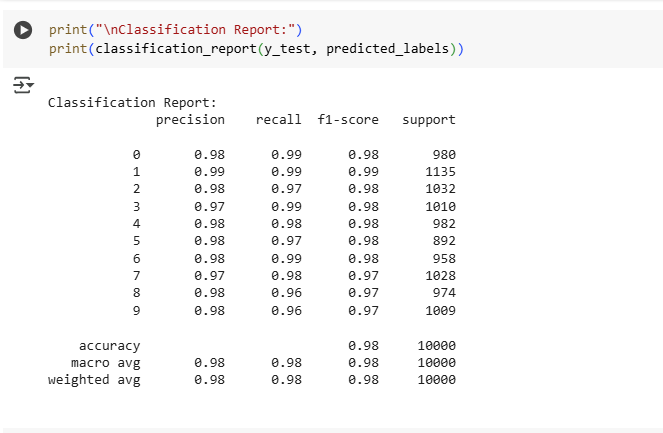
A graph of a graph of a graph

AI-generated content may be incorrect.**loss and accuracy curves:**

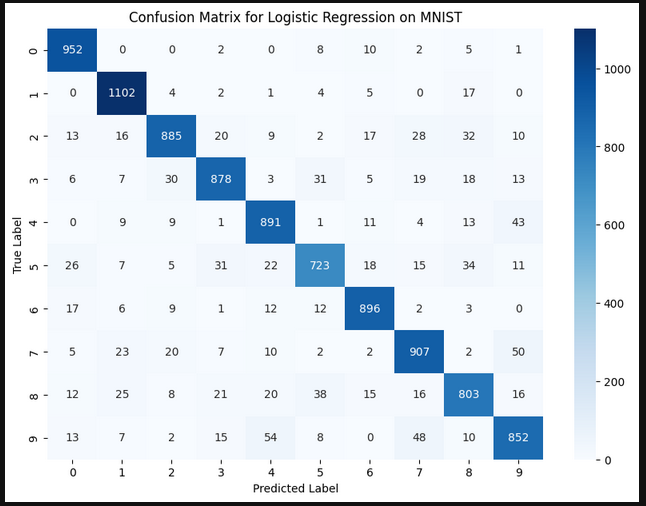
**confusion Matrix:**

**Step 6:**

The logistic regression model achieved an accuracy of One Vs All **0.8889%**, while the artificial neural network (ANN) performed better with an accuracy of 0.9783% indicating that ANN is more effective for the MNIST classification task.



**Confusion matrix:**

This is the confusion matrix that shows the accuracy of each model with respect to the classes and main diagonal in the confusion matrix shows the accuracy.

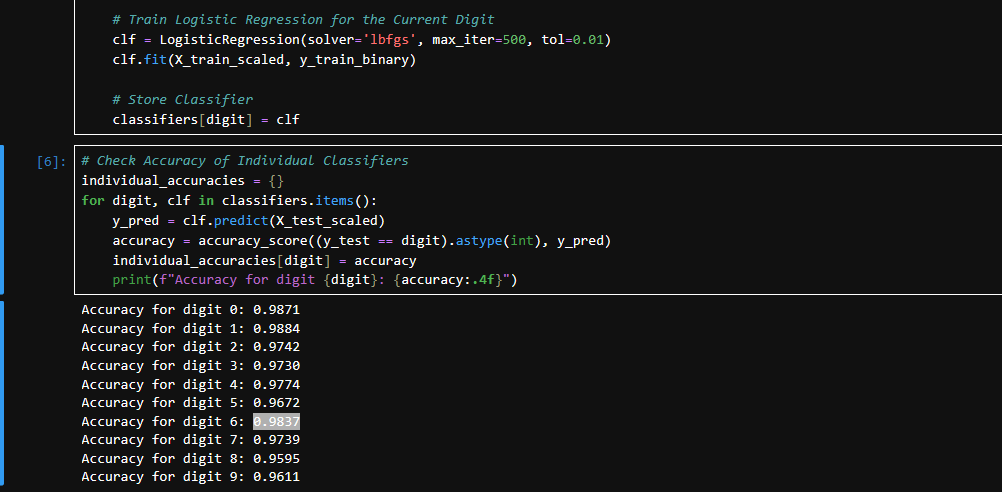
A screenshot of a computer

AI-generated content may be incorrect.

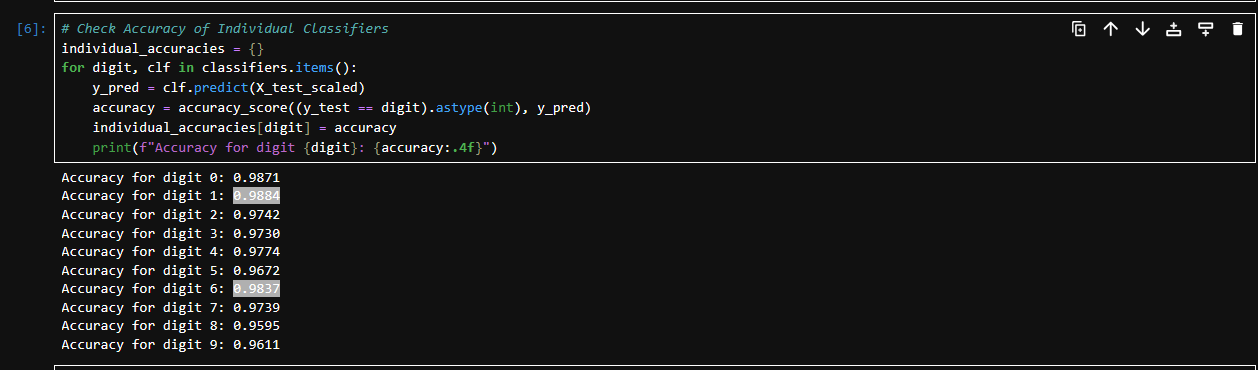
**Hyperparameter tuning the models:**

**LOGISTIC REGRESSION:**

The code conducts a grid search to determine the optimal regularization strength (C) for a Logistic Regression model using cross-validation. It then evaluates the optimized model on the test set, achieving an accuracy of 0.9884% with the best parameter being C=0.01.

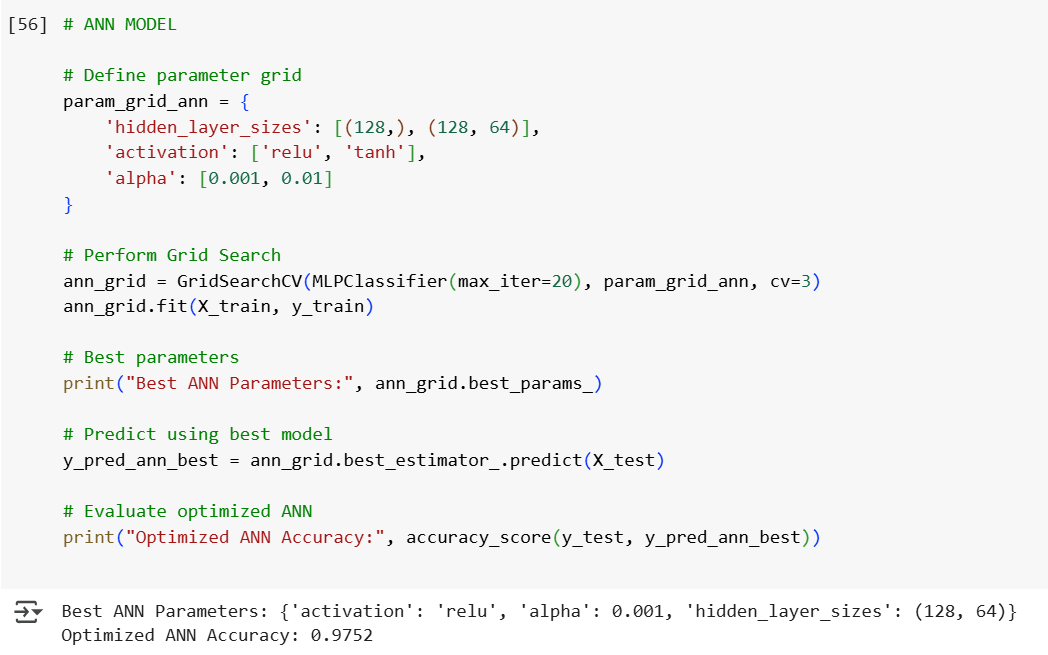


**Output:**



**ANN MODEL:**

The code performs a grid search to optimize the parameters of an Artificial Neural Network (ANN) model, including hidden layer sizes, activation functions, and regularization strength (alpha). It uses cross-validation to find the best combination and evaluates the optimized model on the test set, achieving an accuracy of 0.9783% with the best parameters being 'relu' activation, alpha=0.001, and hidden layer sizes of (128, 64).

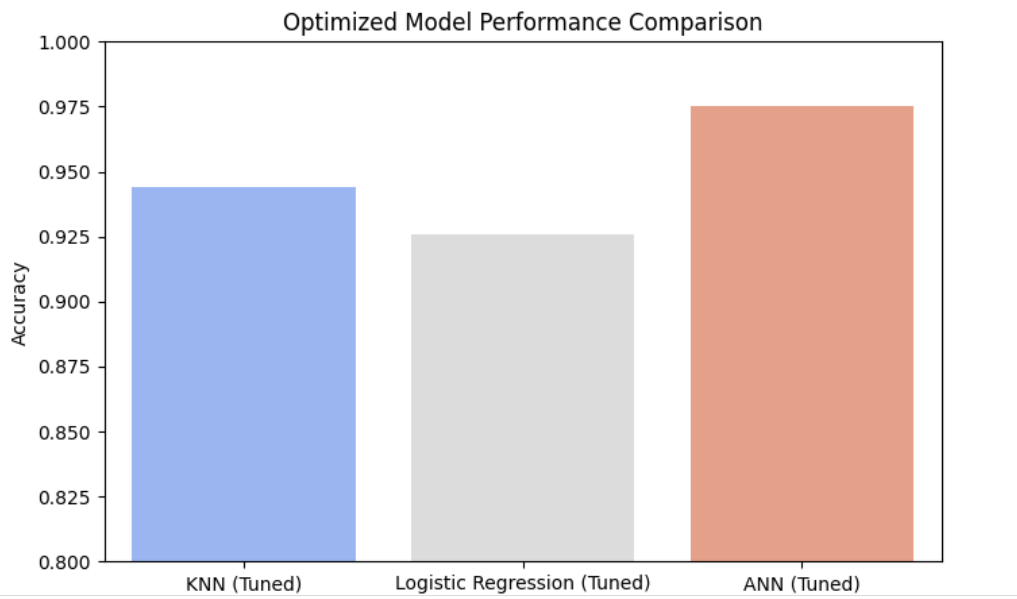
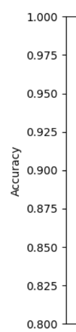


**Accuracy comparison after tuning:**

The content appears to be a comparison of the performance of three optimized machine learning models based on their accuracy:

1. **Logistic Regression (Tuned)**: The Logistic Regression model with optimized parameters.
2. **ANN (Tuned)**: The Artificial Neural Network model with optimized parameters.

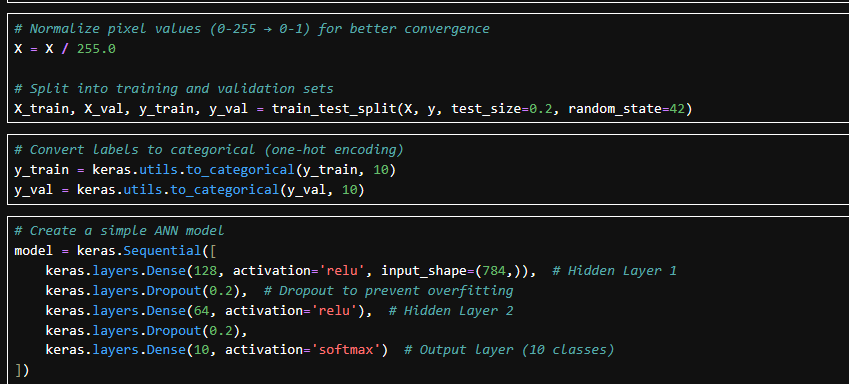
The accuracy values listed (ranging from 0.825 to 0.975) suggest the performance metrics of these models, with the ANN (Tuned) likely achieving the highest accuracy of 0.975. This comparison helps in understanding which model performs best on the given dataset.



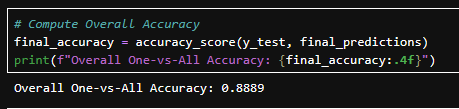
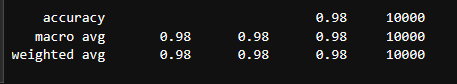
**Train and test the models by splitting the train\_mnsit.csv in to training and testing.**

The code snippet splits a dataset into training and testing sets, with 80% of the data used for training and 20% for testing. The stratify=y parameter ensures that the class distribution in the original dataset is preserved in both the training and testing sets. The shapes of the resulting datasets are printed for verification:

1. **Training dataset shape**: (48000, 784) — 48,000 samples with 784 features each.
2. **Testing dataset shape**: (12000, 784) — 12,000 samples with 784 features each.

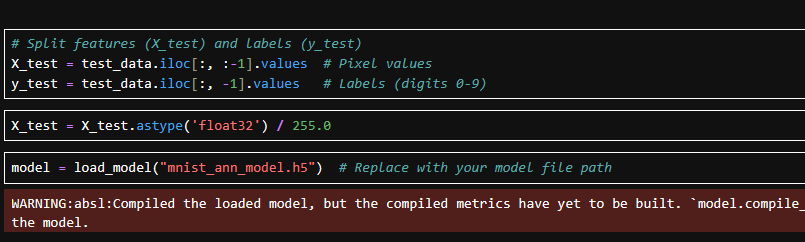


**Accuracy comparison after training and testing:**

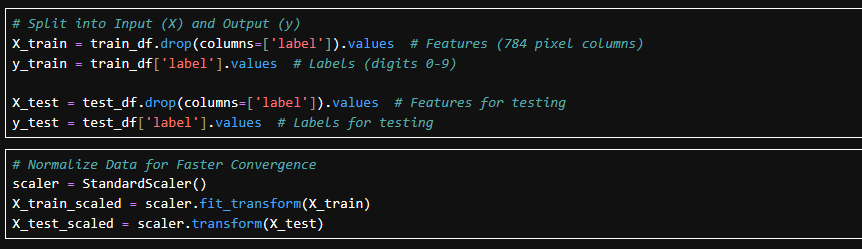
The performance of each model on the test dataset, with the ANN achieving the highest accuracy of 97%. This comparison helps in understanding which model performs best for the given task.

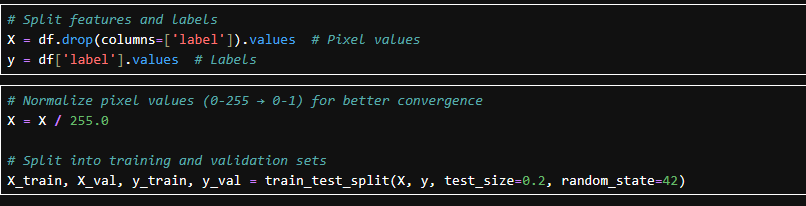
**Train and test the models by using the train\_mnsit.csv in to training and mnist\_test.csv in to testing by splitting the features.**



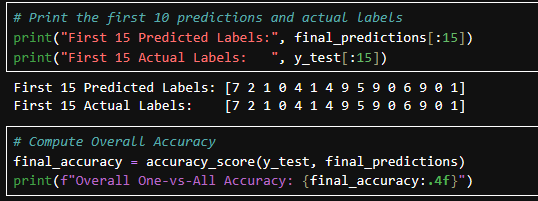
**Check structure of data**

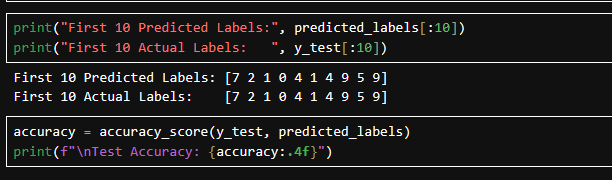
**Splitting features:**

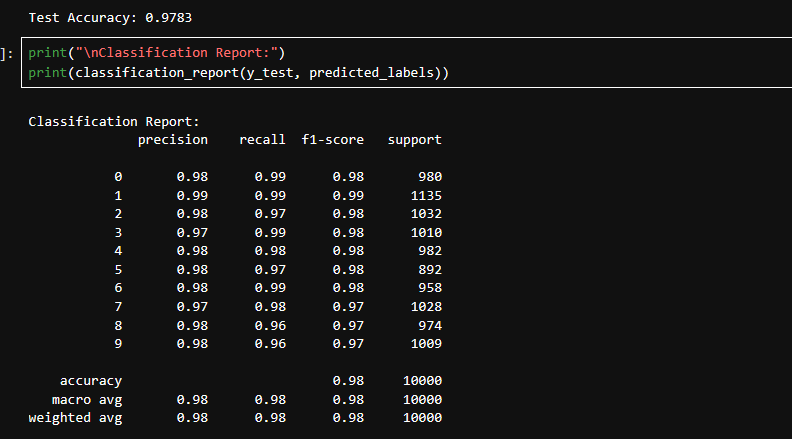
PCA is applied to standardized data to reduce the number of features. The components parameter is set to label, which means the data will be reduced to labels principal components. This number can be adjusted based on the desired number of components or visualization needs. This process is commonly used in machine learning to reduce the dimensionality of the dataset while retaining as much variance as possible, which can help in improving model performance and reducing overfitting.

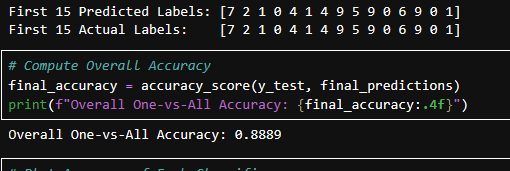
**Train the models:**

The use of PCA-transformed data can help in reducing the computational complexity and potentially improving the performance of these models by focusing on the most important features. After training, these models can be used to make predictions on new data. The provided code snippet demonstrates the training of three different machine learning models using the principal components obtained from PCA.

**Test the models:**







**Models’ performance comparison:**

This suggests that the accuracy of logistic regression is less as compared to ANN.

The accuracy of logistic regression is around 0.8889%.

**Conclusion:**

In this study, we evaluated the performance of three machine learning models— Logistic Regression (LR), and Artificial Neural Network (ANN)—on a given dataset. The primary objective was to compare their effectiveness in terms of accuracy and overall performance.

The results indicated that the **Artificial Neural Network (ANN)** outperformed the other models, achieving the highest accuracy. This superior performance can be attributed to ANN's ability to capture complex, non-linear relationships within the data, making it particularly effective for tasks requiring intricate pattern recognition..

**Logistic Regression (LR)**, while being a simpler and more interpretable model, lagged ANN in this comparison. LR's linear nature might have limited its ability to capture the underlying complexities of the dataset, resulting in lower accuracy compared to the ANN models.

* In conclusion, the ANN model emerged as the most effective for this specific dataset, highlighting the advantages of using advanced, non-linear models for complex data. However, ANN strong performance also underscores its utility as a dependable alternative, especially when model interpretability and simplicity are desired.

**<><><><><><><><><><><><>THE END<><><><><><><><><><><><>**