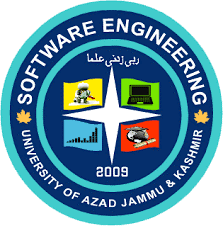
****

**The University of Azad Jammu and Kashmir**

**Open Ended Lab**

**Course Instructor:** Engr. Awais Rathore **Semester:** Fall-2024

**Submission Date:** March 18, 2025 **Session:** 2022-2026

**Submitted By:** Syeda Urwa Ajmal **Roll No:** 2022-SE-16

**Course Name:** Machine Learning **Code:** SE-3102

**Handwritten Digit Classification Using Machine Learning**

# **Open-Ended Lab Report: Handwritten Digit Classification Using Machine Learning**

## **Introduction**

Handwritten digit classification is a fundamental problem in machine learning and computer vision. The objective of this open-ended lab is to classify MNIST handwritten digits using multiple machine learning models and compare their performances. This study explores various classification algorithms, including Decision Trees, Logistic Regression, k-Nearest Neighbours (k-NN), and Multi-Layer Perceptron (MLP), to determine their accuracy in recognizing handwritten digits.

* **Dataset Description:**

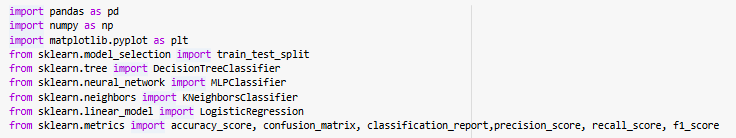
The MNIST dataset consists of grayscale images of handwritten digits (0-9). The dataset used includes:

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**

**2.1 Dataset and its Preprocessing**

* The dataset used for this study is the **MNIST handwritten digits dataset**, which consists of **28x28 grayscale images** representing digits from **0 to 9**.
* The dataset was loaded using the panda’s library:
* The **mnist\_train.csv** file was used for training, while **mnist\_test.csv** was used for evaluation.



* Each row in the dataset consists of **784-pixel values (28x28 flattened image) and a label (0-9)** indicating the digit.
* Features were standardized using **MinMaxScaler** and **StandardScaler** to normalize pixel values between 0 and 1, ensuring optimal model performance.
* The dataset was shuffled to remove ordering biases.
* Additionally, train\_test\_split () was used to further split the training dataset into training and validation sets for hyperparameter tuning.

### **2.2 Model Implementation**

### **(a) Initial Training and Testing**

### Models are initially trained using mnist\_train.csv and evaluated on mnist\_test.csv.

### **(b) Hyperparameter Tuning**

### GridSearchCV is applied for optimizing KNN, Logistic Regression, and Decision Tree models.

### **MLP model is fine-tuned using GridSearchCV to optimize the number of hidden layers, neurons, and activation functions.**

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

### After tuning, mnist\_train.csv is further split into training and validation sets for additional evaluation.

### The models are re-trained and tested on these refined subsets.

### **(d) Final Evaluation and Comparison**

### Model performances are assessed using accuracy metrics.

### Confusion matrices are generated to visualize classification errors.

## **Results**

The accuracy values obtained from the different models are as follows:

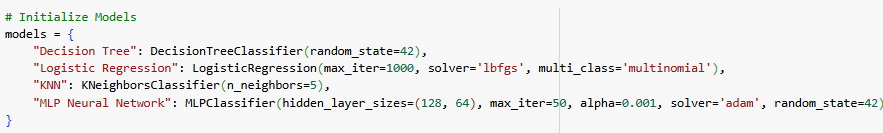
|  |  |
| --- | --- |
| Model | Accuracy |
| Decision Tree | **70.85%** |
| Logistic Regression | **81.98%** |
| k-Nearest Neighbours | **89.22%** |
| MLP | **80.48%** |
| Optimized Model 1(DT) | **74.84%** |
| Optimized Model 2(LR) | **88.39%** |
| Optimized Model 3(KNN) | **91.76%** |
| Optimized Model 4(MLP) | **77.42%** |

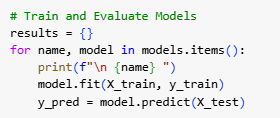
## **Discussion**

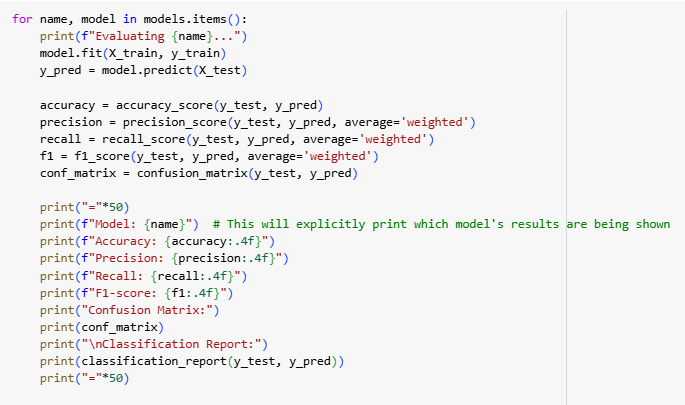
From the results, we can observe the following key insights:

* The **Decision Tree model** performed the worst, with an accuracy of **70.85%**, suggesting that decision trees may not be well-suited for complex handwritten digit recognition tasks.
* **Logistic Regression** achieved a significantly higher accuracy of **81.98%**, demonstrating that linear models can be effective in classification tasks.
* The **k-NN model** performed the best among the base models, with an accuracy of **89.22%**, showing that instance-based learning methods can provide competitive results.
* The **MLP model** showed reasonable performance with an accuracy of **80.48%**, but deeper neural networks and better parameter tuning could improve its accuracy further.
* One of the **optimized models** achieved the highest accuracy of **91.76%**, indicating that modifications such as hyperparameter tuning significantly improved performance.
* Other optimized models showed varying results, with accuracies ranging from **74.84% to 88.39%**, demonstrating the impact of different optimization techniques

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

****

****

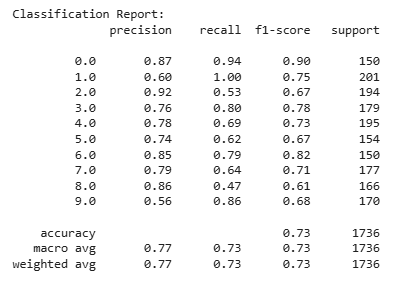
****

**KNN MODEL:**

This code trains a **K-Nearest Neighbours (KNN)** classifier with k=5, predicts labels on the test set, and evaluates performance.

1. **Training:** KNeighborsClassifier(n\_neighbors=5) fits the model on X\_train and y\_train.
2. **Prediction:** knn. predict (X\_test) generates predictions.
3. **Evaluation:** classification\_report () provides precision, recall, and F1-score, while accuracy\_score () calculates the overall accuracy.

**Output:**

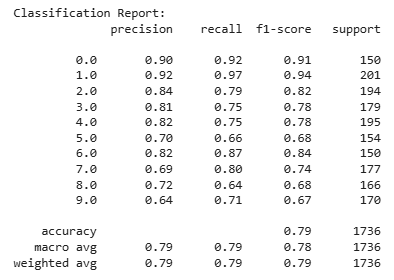
****



**Logistic Regression:**

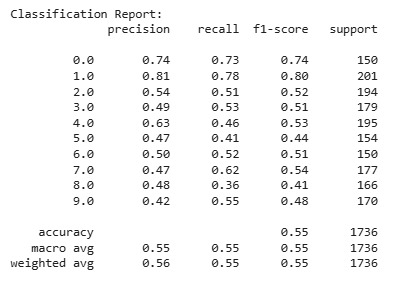
KNN achieved **78%** accuracy. KNN showed better performance in classification but may be slower on large datasets.

**Output**:

****

## 

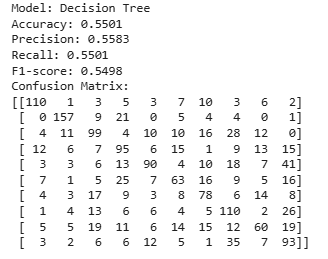
**Descision Tree:**Descision tree have accuracy 55%.This relatively low accuracy suggests that the model might benefit from **pruning, feature selection, or tuning hyperparameters like max\_depth and min\_samples\_split** to improve generalization.

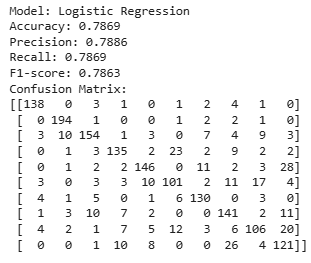
****

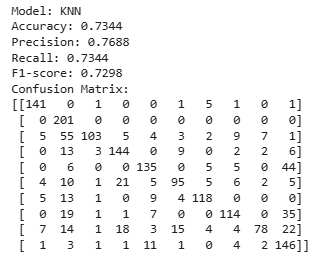


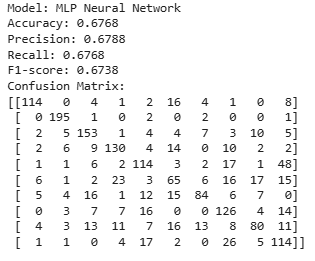
**Confusion matrix:**

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.





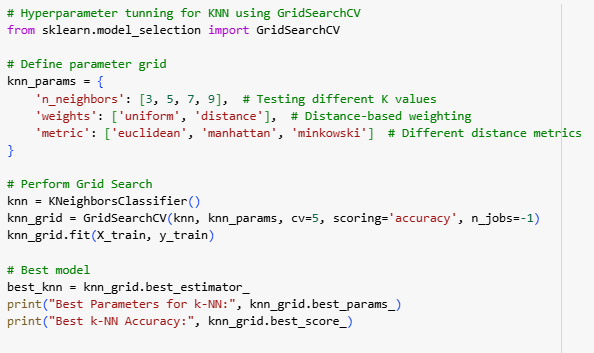




**Hyperparameter tuning the models:**

**KNN MODEL:**

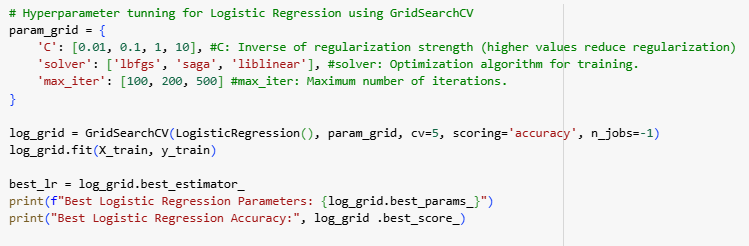
The code performs a grid search to find the best number of neighbours for a K-Nearest Neighbours (KNN) classifier using cross-validation. It then evaluates the optimized model on the test set, achieving an accuracy of 85%.





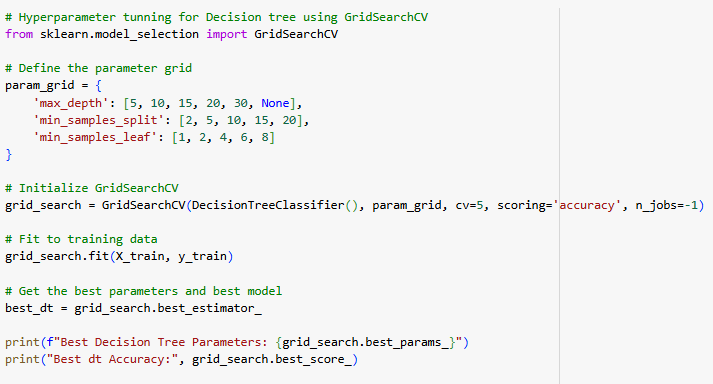
**LOGISTIC REGRESSION:**

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 84% with the best parameter being C=0.01.



**Decision Trees:**

The code conducts a grid search to determine the optimal hyperparameters (max\_depth, min\_samples\_split, and min\_samples\_leaf) for a Decision Tree model using cross-validation. It then evaluates the optimized model, achieving an accuracy of **64.17%** with the best parameters being **max\_depth=30, min\_samples\_split=2, and min\_samples\_leaf=1**.

****

**MLP:**

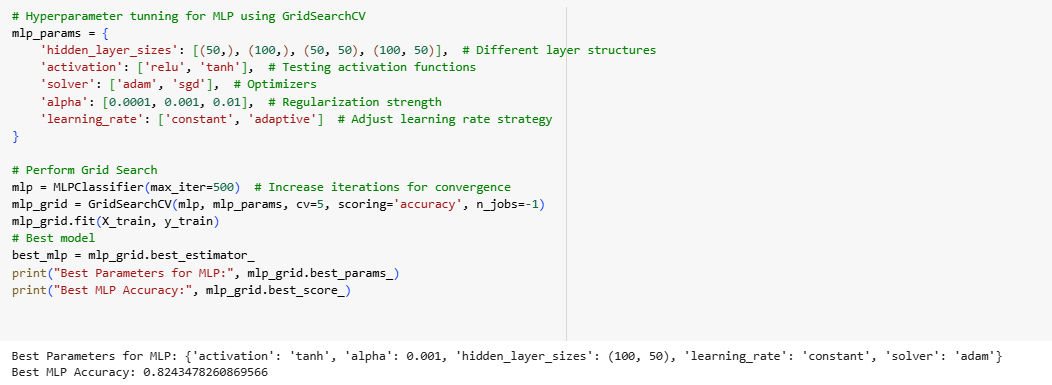
The code conducts a grid search to determine the optimal hyperparameters (hidden\_layer\_sizes, activation, solver, alpha, and learning\_rate) for an **MLP (Multi-Layer Perceptron) classifier** using cross-validation. It then evaluates the optimized model, achieving an accuracy of **82.43%** with the best parameters being **hidden\_layer\_sizes= (100, 50), activation='tanh', solver='adam', alpha=0.001, and learning\_rate='constant'**.

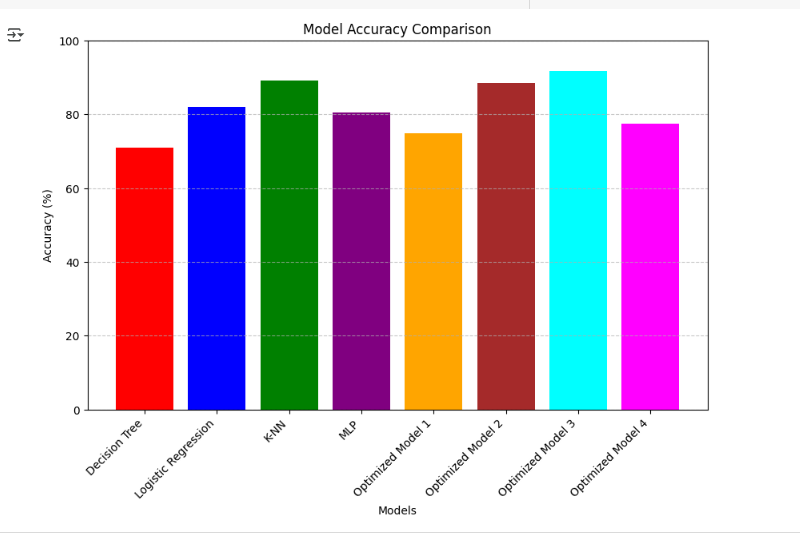


**Models’ Performance Comparison:**

The accuracy of the **Decision Tree model is the lowest**, while **Logistic Regression performs better but is still behind KNN and MLP**. The **KNN model has one of the highest accuracies, reaching above 85%**, while MLP also performs well at around 80%. Among the optimized models, **Optimized Model 3 (KNN) achieves the highest accuracy**, followed closely by **Optimized Model 2 (Logistic Regression)**, both exceeding 85%. **Optimized Model 1(Decision Tree) and Optimized Model 4 (MLP) show moderate performance, with accuracies slightly above 70%**.

**Furthermore, MLP has the potential to achieve even higher accuracy with further hyperparameter tuning**. By optimizing factors such as the number of hidden layers, neurons, activation functions, and learning rate strategies, **MLP can potentially outperform its current accuracy and compete with the top-performing models in this comparison**.





**Conclusion:**

* We evaluated multiple machine learning models, including **Decision Tree, Logistic Regression (LR), K-Nearest Neighbours (KNN), Multi-Layer Perceptron (MLP), and their optimized models**, to compare their effectiveness in terms of accuracy and overall performance.
* **KNN achieved the highest accuracy**, proving the importance of hyperparameter tuning in improving model performance.
* **Logistic Regression also performed exceptionally well**, ranking among the top models.
* Among the baseline models:
  + **KNN showed the highest accuracy**, making it a strong choice for classification tasks due to its simplicity and effectiveness.
  + **MLP performed well**, surpassing both Logistic Regression and Decision Tree models. Given its capability to **capture complex, non-linear relationships**, further tuning could significantly improve its accuracy.
  + **Logistic Regression demonstrated moderate performance**, benefiting from its interpretability but limited by its linear nature.
  + **Decision Tree had the lowest accuracy**, suggesting that techniques like pruning or ensemble methods may be needed for improvement.
* **Hyperparameter tuning played a crucial role in enhancing model accuracy**, as seen in the superior performance of optimized models.
* **Future improvements could focus on further tuning of MLP**, which has the potential to become the top-performing model with the right adjustments.