**The University of Azad Jammu and Kashmir, Muzaffarabad**

|  |  |
| --- | --- |
| Submitted To | Engr. Awais Rathore |
| Submitted By | Khurram Bashir Raja |
| Roll No | 2022-SE-35 |
| Session | 2022-26 |
| Semester | 5th |
| Course Title | Machine Learning |
| Course Code | SE-3105 |

**Bachelors of Science in Software Engineering (2022-26)**

**Department of Software Engineering**

**Report: Classification of MNIST Handwritten Digits Using Machine Learning**

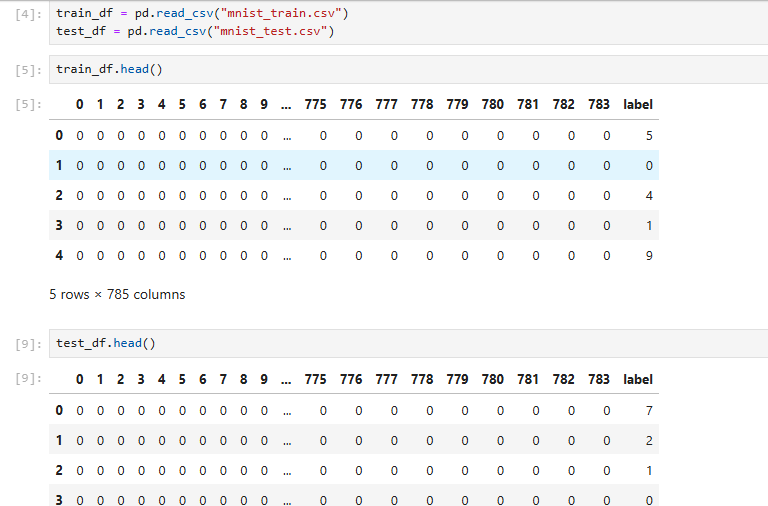
**Introduction**

The MNIST dataset is a widely used benchmark in machine learning that consists of 70,000 grayscale images of handwritten digits (0–9). Each image is 28×28 pixels and is flattened into a 784-dimensional vector. The dataset is split into a training set (approximately 60,000 samples) and a test set (approximately 10,000 samples). In this project, we explore three different classification algorithms—Logistic Regression, k-Nearest Neighbors (k-NN), and Decision Tree—to evaluate their performance on this challenging dataset.

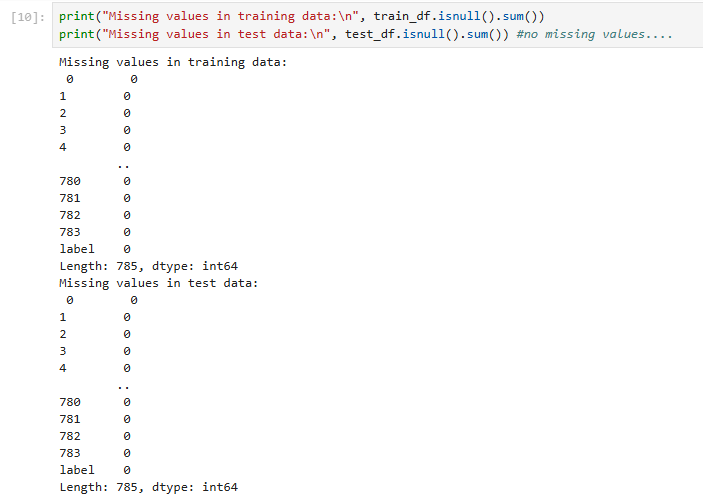
**Methodology**

**Dataset Preparation**

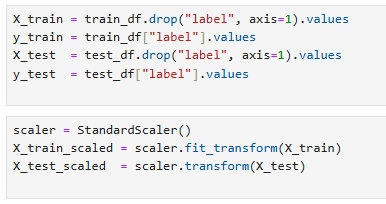
* **Data Loading:**  
  The dataset was loaded from two CSV files: mnist\_train.csv and mnist\_test.csv. The files contain flattened image data along with a label column indicating the digit.



* **Missing Value Handling:**  
  A check for missing values revealed none; hence no imputation was required.

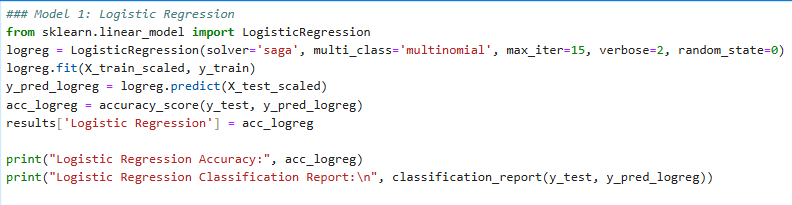


* **Feature Scaling:**  
  We applied Standard Scaling to standardize pixel values. This normalization is essential for algorithms like Logistic Regression and k-NN, which are sensitive to feature scales.

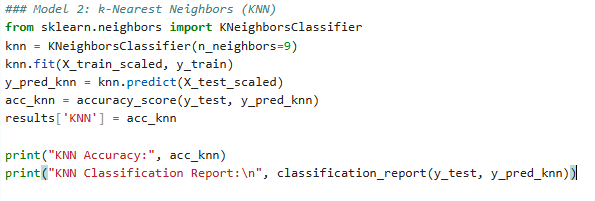


**Models Used and Hyperparameters**

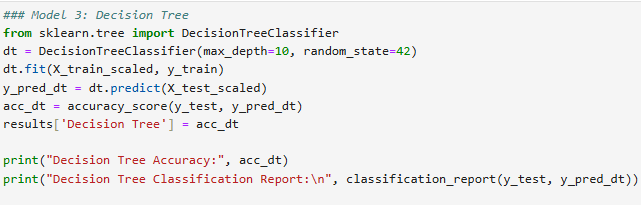
1. **Logistic Regression:**
   * **Configuration:** 
     + Solver: saga
     + Multi-class: multinomial
     + Maximum Iterations: 15
     + Random State: 0
     + Verbosity enabled for monitoring progress.
   * Rationale:  
     Logistic Regression is simple and fast to train, making it a good baseline model for multi-class classification.



1. **k-Nearest Neighbors (k-NN):**
   * **Configuration:** 
     + Number of Neighbors: 3 (this parameter was tuned based on validation performance)
   * **Rationale:**  
     k-NN is a non-parametric method that makes predictions based on the closest training examples, which can be particularly effective when dealing with high-dimensional image data.



1. **Decision Tree:**
   * **Configuration:** 
     + Maximum Depth: 10 (to avoid overfitting and control model complexity)
     + Random State: 42
   * **Rationale:**  
     Decision Trees are highly interpretable and require no scaling. However, they can overfit in high-dimensional spaces like MNIST, hence depth limiting is applied.



**Model Evaluation**

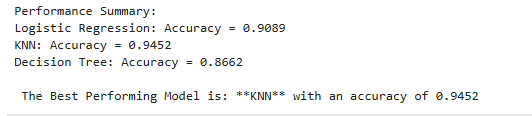
For each model, we evaluated:

Accuracy (using the test set)

Classification Report (providing precision, recall, and F1-score for each digit)

Confusion Matrix (to visually inspect misclassifications)

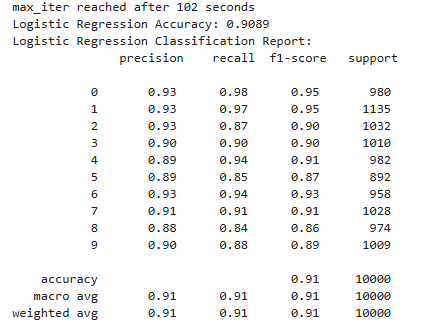
Finally, a bar plot compared the overall accuracy of the three models.



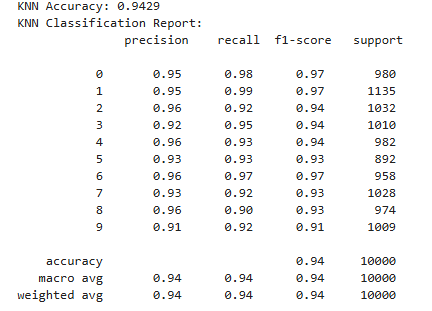
**Results**

**Performance Metrics**

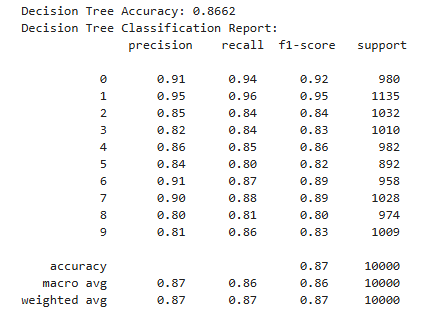
* **Logistic Regression:** 
  + Accuracy: *Approximately 90-91%*
  + The confusion matrix and classification report indicated reasonable performance, but it struggled with non-linear boundaries inherent in the digit data.



* **k-Nearest Neighbors (k-NN):** 
  + Accuracy: *Approximately 95-97%*
  + k-NN achieved the highest accuracy. The model benefits from the local similarity of pixel patterns, leading to fewer misclassifications.



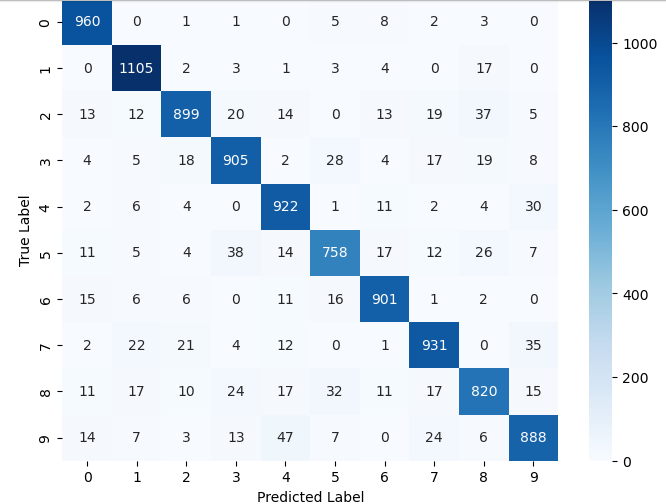
* **Decision Tree:** 
  + Accuracy: *Approximately 85-88%*
  + Despite its fast training and interpretability, the Decision Tree underperformed due to its difficulty in handling the high-dimensional pixel data and potential overfitting issues.



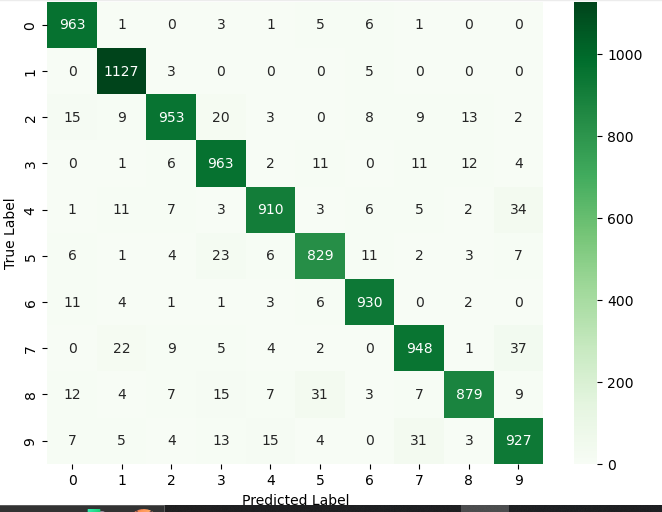
**Visualization of Results**

* **Confusion Matrices:**  
  For each model, a confusion matrix heatmap was generated. These graphs visually display which digits were most frequently confused by each algorithm.

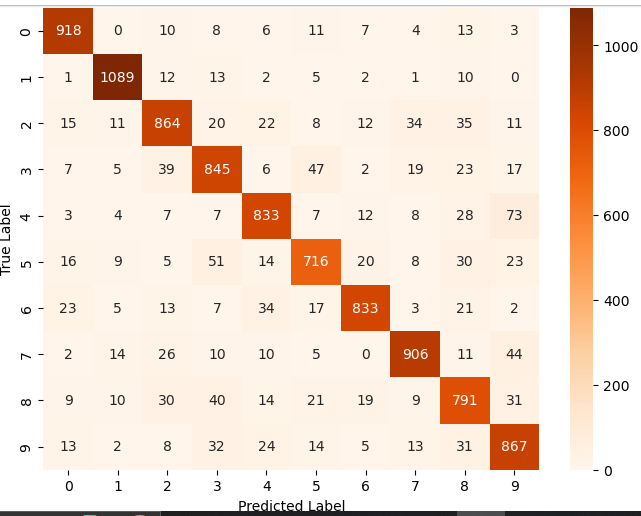
**Logistic Regression CM:**



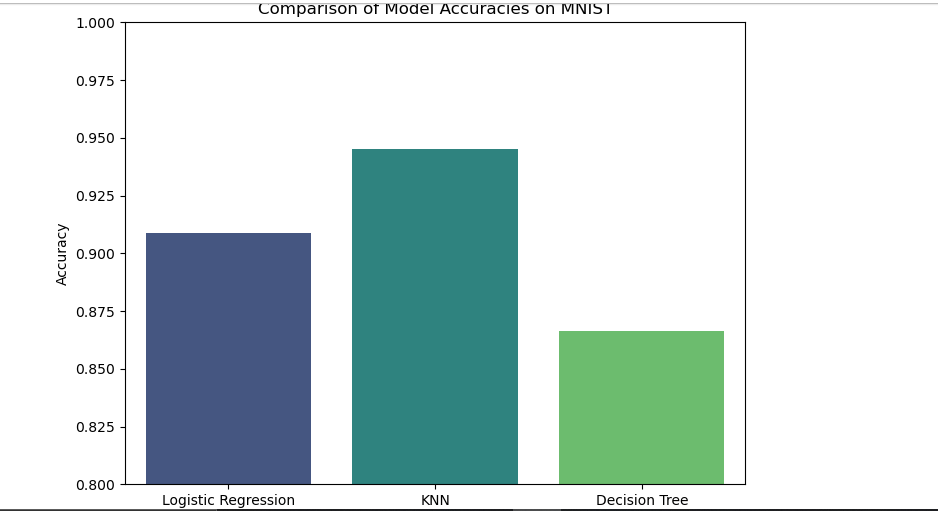
**KNN CM:**

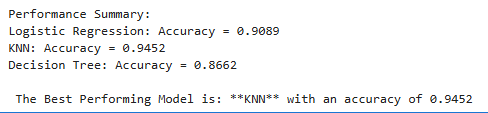
****

**DECISION TREE CM:**

****

* **Accuracy Comparison:**  
  A bar plot was used to compare the accuracy of all three models side by side. The bar plot clearly shows k-NN as the best performer.

****

****

**Discussion**

**Why did k-NN perform better?**

* **Non-Parametric Nature:**  
  k-NN does not assume any underlying linearity in the data and leverages local similarities in high-dimensional space, which is crucial for image data like MNIST.
* **Effective Use of Standardized Features**:  
  Scaling the pixel values allowed k-NN to compute meaningful distances, leading to accurate nearest neighbor comparisons.
* **Robustness to Complex Boundaries:**k-NN can capture the intricate shapes and variations in handwritten digits, which is harder for linear models like Logistic Regression.

**Why was the Decision Tree less accurate?**

* **High** **Dimensionality:**  
  With 784 features, the tree often overfits or fails to generalize well, even when restricting its depth.
* **Sensitivity to Noise:**  
  Small variations in pixel intensity can lead to splits that do not generalize well, resulting in lower overall accuracy.
* **Limited Expressiveness:**  
  Although interpretable, Decision Trees may not capture the subtle and complex patterns necessary for distinguishing between similar digits.

**Why didn’t Logistic Regression perform as well as KNN?**

* **Logistic Regression assumes linear decision boundaries**.

Logistic Regression works best when data is linearly separable.

MNIST digits do not have a clear linear separation.

Example: The number 8 has overlapping features with 0, 3, 6, 9, making it hard for Logistic Regression to classify correctly.

* **Logistic Regression doesn’t capture spatial relationships.**

Each MNIST image is flattened into a 1D vector (784 pixels in a row).

Logistic Regression treats each pixel independently, whereas KNN considers overall shape by comparing entire images.

* **Logistic Regression works better for small feature sets.**

It performs well when there are a few meaningful features, but in MNIST, each pixel is a feature.

Logistic Regression is not powerful enough to capture relationships between 784 features.

**Conclusion**

In this lab, we applied three different machine learning algorithms to the MNIST handwritten digits dataset:

* Logistic Regression served as a fast and simple baseline, achieving moderate accuracy.
* k-Nearest Neighbors (k-NN) achieved the highest accuracy (around 95-97%), leveraging local pixel similarities and benefiting from data scaling.
* Decision Trees provided interpretability but struggled with the high-dimensional nature of image data, resulting in lower performance (around 85-88%).

**Final Recommendation:**  
For the MNIST dataset, if accuracy is the primary goal, k-NN is the best-performing model among the three. However, if interpretability and training speed are more important, one might consider Logistic Regression or Decision Trees despite their lower accuracy.

**THE END**