DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF E&ME, NUST, RAWALPINDI

EC- 350 AI & Decision Support System

Assignment Report: Classification of MNIST Handwritten Digits

Course Instructor: Dr. Farhan Hussain

Student Name: Waseem Ghulam (409600)

Degree/ Syndicate: 44/B

1. Introduction

The goal of this assignment is to build and evaluate a digit classification system using the MNIST dataset, which consists of 60,000 training images and 10,000 testing images of handwritten digits (0–9). Five different machine learning models were trained and compared: K-Nearest Neighbours (KNN), Naive Bayes, Decision Tree, Multi-Layer Perceptron (MLP), and a Convolutional Neural Network (CNN).

2. Dataset Exploration

The dataset was loaded and explored to understand its structure.

- **Training Set Size:** 50,000 samples
- **Test Set Size:** 10,000 samples
- **Image Dimensions:** 28 x 28 pixels (grayscale)

Sample Images:

Below are examples of the handwritten digits from the dataset:

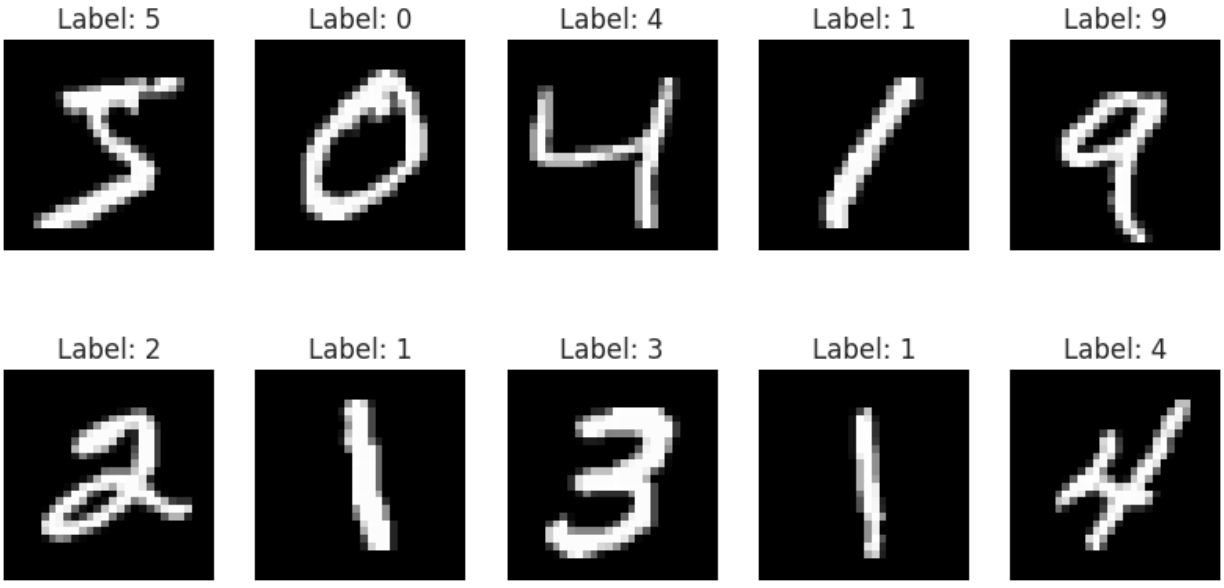

Figure 1: Sample MNIST images showing various handwritten styles.

3. Data Preprocessing

To prepare the data for training, the following steps were taken:

1. **Normalization:** Pixel values were scaled from the range 0–255 to 0–1 to improve model convergence.
2. **Reshaping:**
   - For standard models (KNN, Naive Bayes, Decision Tree, MLP), images were flattened into 1D vectors of size 784.
   - For the CNN, images were reshaped into 3D tensors of shape (28, 28, 1).

4. Model Evaluation & Results

4.1. K-Nearest Neighbours (KNN)

- **Hyperparameters:** k=3
- **Performance:** KNN performs well but is computationally expensive during prediction.

```
Accuracy: 0.9681
```

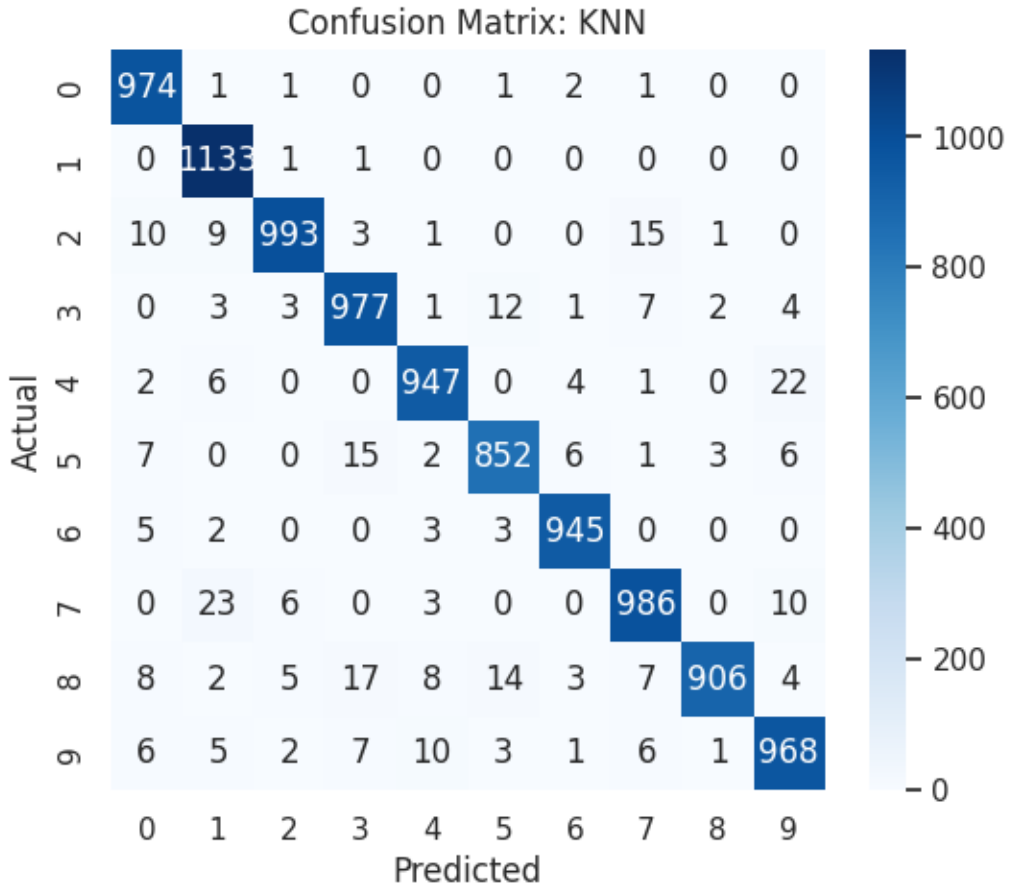

Figure 2: Confusion Matrix for KNN Classifier.

4.2. Naive Bayes

- **Model Type:** Gaussian Naive Bayes
- **Performance:** This model is very fast but assumes independence between pixels, leading to lower accuracy compared to complex models.

```
Accuracy: 0.5544
```

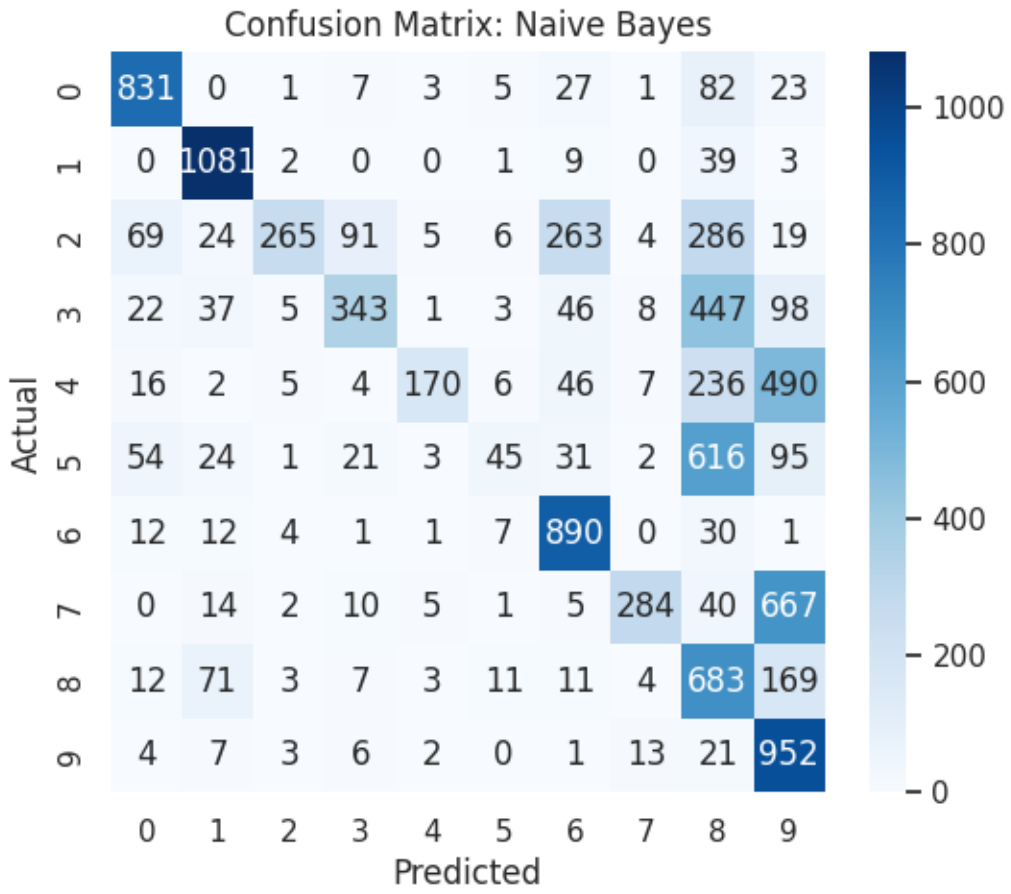

Figure 3: Confusion Matrix for Naive Bayes Classifier.

4.3. Decision Tree Classifier

- **Hyperparameters:** Default settings (random_state=42)
- **Performance:** Decision trees provide interpretability but are prone to overfitting on raw pixel data.

```
Accuracy: 0.8693
```

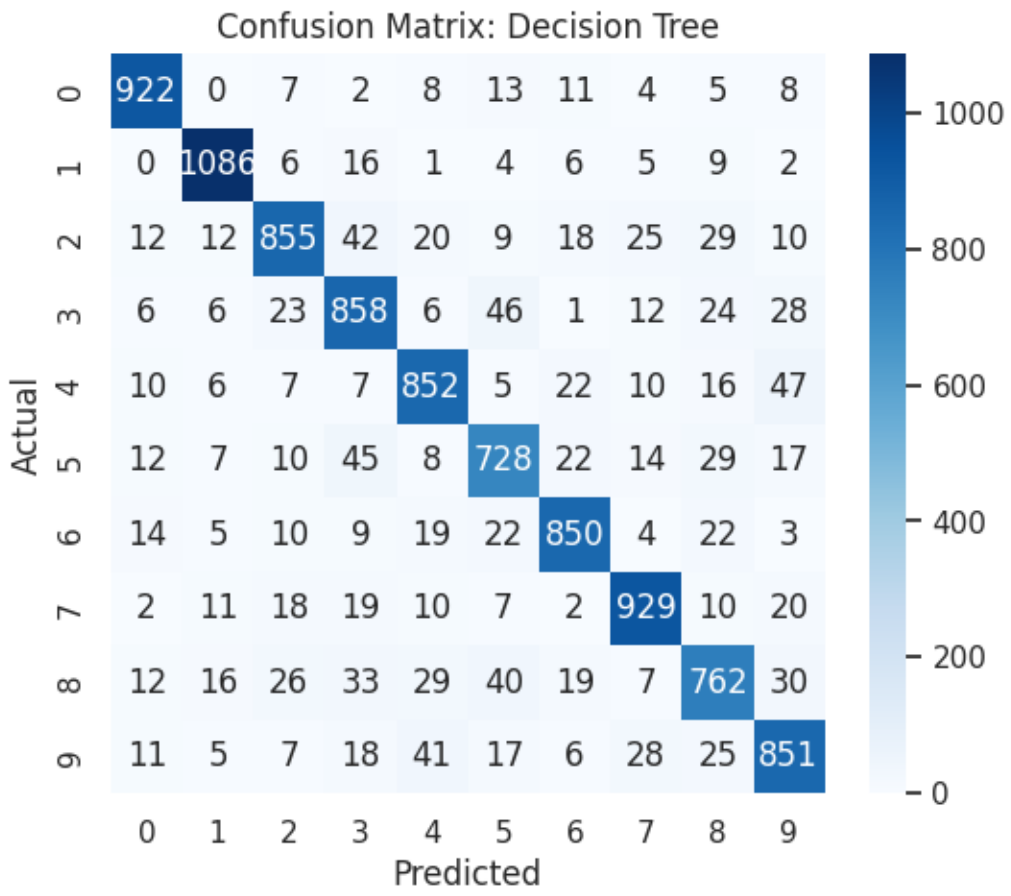

Figure 4: Confusion Matrix for Decision Tree Classifier.

4.4. Multi-Layer Perceptron (MLP)

- **Architecture:** Input (784) → Dense(128, ReLU) → Dense(64, ReLU) → Output(10, Softmax)
- **Optimizer:** Adam
- **Performance:** The neural network significantly outperforms simpler statistical models by learning non-linear relationships.

```
Accuracy: 0.9494
```

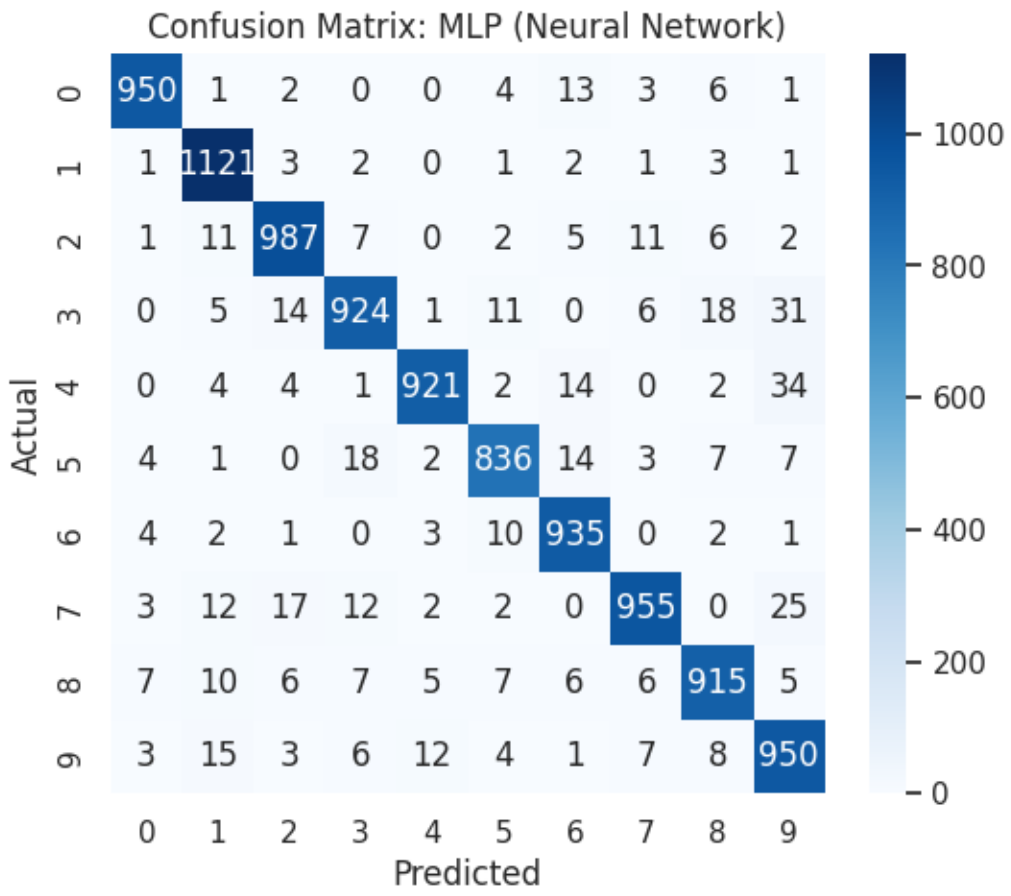

Figure 5: Confusion Matrix for MLP Neural Network.

4.5. Convolutional Neural Network (CNN)

- **Architecture:** Conv2D → MaxPool → Conv2D → MaxPool → Flatten → Dense
- **Performance:** CNN achieved the highest accuracy as it effectively captures spatial hierarchies in image data.

```
Accuracy: 0.9869
```

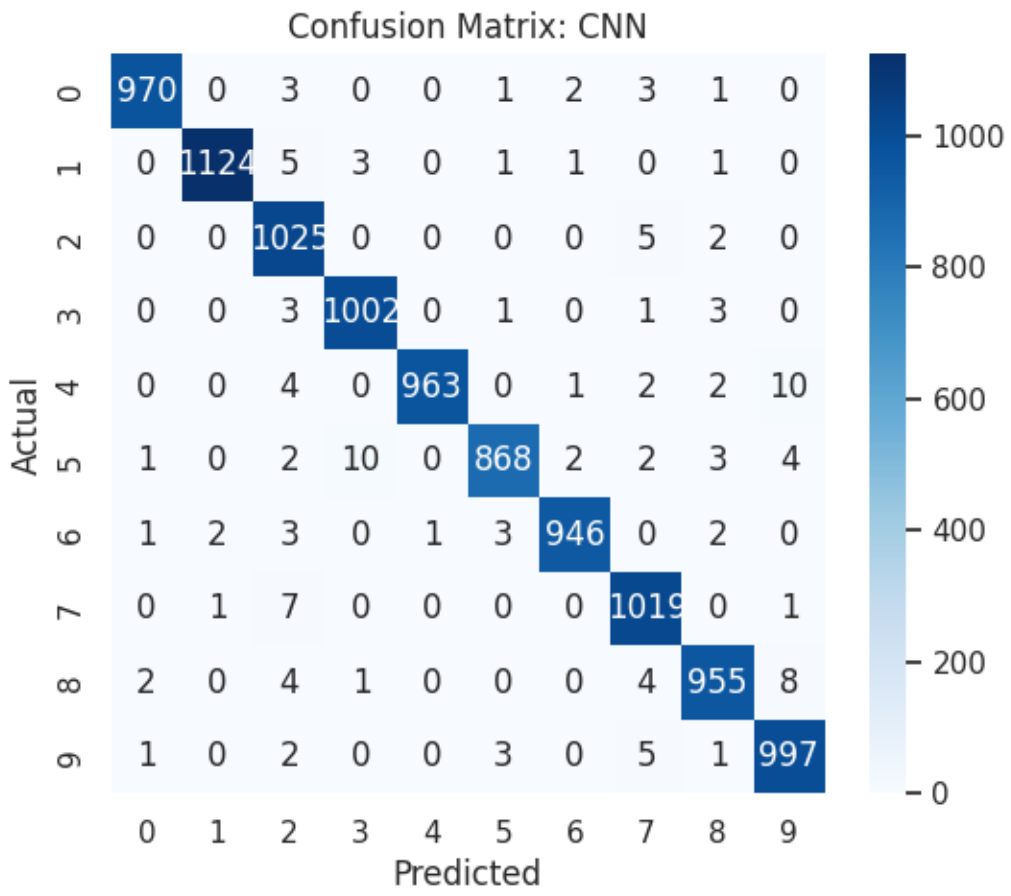

Figure 6: Confusion Matrix for CNN.

5. Final Comparison & Conclusion

The table below summarizes the accuracy scores for all five models evaluated on the test set[21].

Model	**Accuracy Score**
Convolutional Neural Network (CNN)	0.9869
Feed-Forward Neural Network (MLP)	0.9494
K-Nearest Neighbours (KNN)	0.9681

Decision Tree	0.8693
Naive Bayes	0.5544

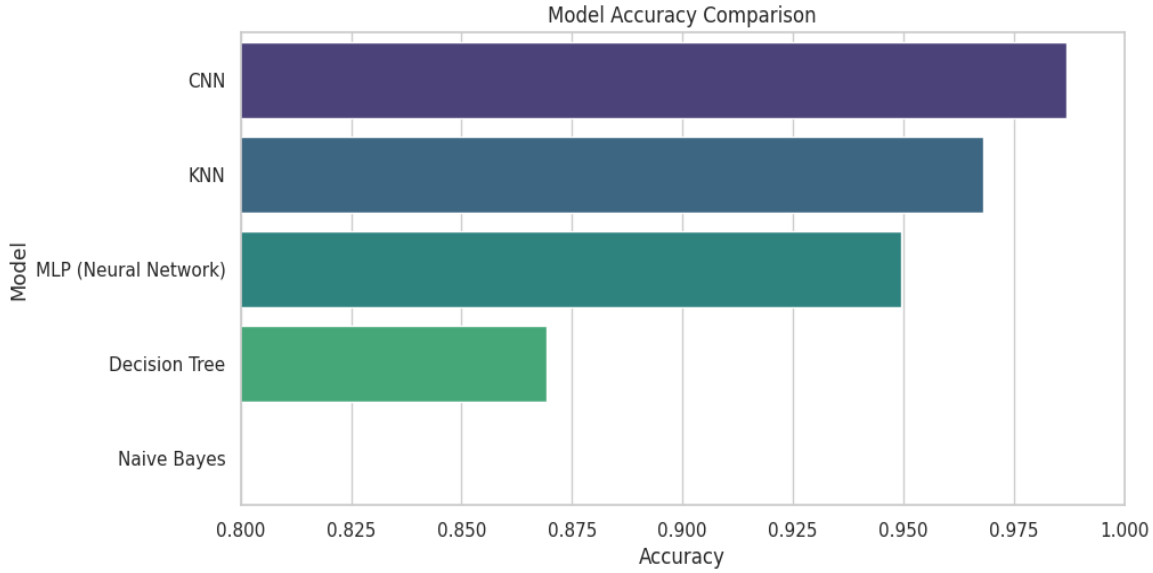

Figure 7: Final accuracy comparison table generated in Python.

Conclusion:

The Convolutional Neural Network (CNN) proved to be the most effective model for this task23. While traditional algorithms like KNN performed reasonably well, they lack the ability to learn spatial features (edges, curves) that the CNN exploits. Naive Bayes performed the poorest due to its assumption that pixel values are independent of each other, which is false for image data.