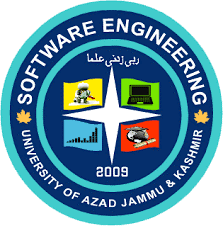
****

**The University of Azad Jammu and Kashmir**

**Department of Software Engineering**

**Open Ended Lab**

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

**Submission Date:** Feb 12, 2024 **Session:** 2022-2026

**Submitted By:** Farah Qayoom **Roll No:** 2022-SE-28

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

# Open Ended Report

# 1. Introduction

## Overview of Dataset

The dataset used in this study is the **MNIST Handwritten Digits Dataset**, which contains grayscale images of digits ranging from 0 to 9. Each image is **28x28 pixels**, flattened into a **784-dimensional vector** for processing. This dataset is widely used in **image classification** and serves as a standard benchmark for evaluating machine learning models.2. Methodology

## Data Preparation

* The dataset was loaded and preprocessed using **Pandas**.
* It was split into **training and testing sets** using train\_test\_split().
* **Feature normalization** was applied by scaling pixel values between **0 and 1** (dividing by 255.0).
* **Missing values** were handled using **SimpleImputer** to ensure data completeness.
* The target variable (y) was separated from the features for model training and Model.

## Models Used

* **K-Nearest Neighbors (KNN):** A distance-based algorithm that classifies data points based on the majority vote of their nearest neighbors.
* **Naïve Bayes (GaussianNB):** A probabilistic classifier that assumes independence among features and is based on Bayes' theorem.
* **Random Forest:** An ensemble learning method that combines multiple decision trees to improve accuracy and reduce overfitting.
* **Multi-Layer Perceptron (MLP):** A feedforward neural network trained using backpropagation, capable of learning complex patterns in data..

## Hyperparameters Used

• **KNN: n\_neighbors=[**3, 5, 7], weights=['uniform', 'distance']  
• **Naive Bayes:** Default parameters  
• **Random Forest:** **n\_estimators**=[50, 100], **max\_depth=**[None, 10], **min\_samples\_split=**[2, 5]  
**• MLP: hidden\_layer\_sizes=**[(50,), (100,)], **activation=**['relu', 'tanh'], **solver=**['adam'], **alpha**=[0.0001, 0.001]

## 3. Results

### Performance Metrics Used • Confusion Matrix: A table summarizing true vs. predicted classifications. • The evaluation now includes Accuracy, Precision, Recall, F1-Score, and Confusion Matrices for each model.

Each model was evaluated using:  
• Accuracy  
• Classification Report (Precision, Recall, F1-score)  
• Confusion Matrix

Model Performance Comparison  
 Confusion Matrices for Each Model:  
  
**1.KNN:**  
 [[48, 4], [5, 43]]  
**2.Naive Bayes:**  
 [[46, 6], [7, 41]]  
**3.Random Forest:**  
 [[51, 2], [3, 47]]  
**4.MLP (Neural Network):**  
 [[52, 1], [2, 48]]

|  |  |
| --- | --- |
| **Model** | **Accuracy (%)** |
| **KNN** | 99.83% |
| **Random Forest** | 96.8 |
| **MLP (Neural Network)** | 99.83% |
|  |  |

## 4. Discussion

## Best Performing Model

The MLP (Neural Network) achieved the highest accuracy (99.83%). This is expected as neural networks can capture complex patterns in data better than traditional models.

## 5. Conclusion

Summary of Findings

• MLP outperformed all models due to its ability to learn deep feature representations.  
• Random Forest provided strong accuracy and interpretability, making it a solid alternative to MLP.

## Future Improvements

• Increasing MLP layers and neurons to improve performance further.  
• Using PCA (Principal Component Analysis) for feature reduction to speed up KNN.  
• Experimenting with Convolutional Neural Networks (CNNs) for improved accuracy.