MNIST Digit Classification – Detailed Analysis Report

1. Introduction

This project aims to classify handwritten digits from the MNIST dataset using various machine learning and deep learning models. The objective was to compare their performance, understand data preprocessing challenges, and identify the most suitable model for production deployment.

2. Dataset Description

- **Name**: MNIST (Modified National Institute of Standards and Technology)
- **Type**: Image Classification
- **Input**: 28x28 grayscale images (pixel values from 0 to 255)
- **Output**: Digit labels (0 to 9)

Dataset Split:

Dataset	Samples	Shape
Train Set	60,000	(28, 28)
Test Set	10,000	(28, 28)
Labels	0–9	Integer classes

3. Data Preparation

Steps Taken:

1. **Reshaping**:
    - Images were reshaped:
    - ML models: 28x28 → 784 (flattened)
    - CNN model: reshaped to (28, 28, 1) for 4D input
2. **Normalization**:
    - Pixel values scaled from [0–255] to [0–1] for stability and performance
3. **Label Formatting**:
    - One-hot encoding used for neural networks
    - Integer labels retained for scikit-learn models like SVM, KNN, RF

4. Models Implemented

A. Neural Network (Keras Sequential)

- Layers: Flatten → Dense(128, relu) → Dense(10, softmax)
- Optimizer: Adam

- Loss: Categorical Crossentropy
- **Accuracy**: **97.90%**

B. Support Vector Machine (SVM)

- Kernel: RBF (default)
- Normalized Input
- **Accuracy**: **97.92%**

C. Random Forest Classifier

- Parameters:
    - Default: n_estimators=100
    - Tuned: n_estimators=150, criterion='gini'
- **Accuracy**: **97.08%** after tuning

D. K-Nearest Neighbors (KNN)

- k = 5 (default)
- Normalized input
- **Accuracy**: **96.88%**

E. Logistic Regression

- Solver: saga, multi_class=multinomial
- Normalized input
- **Accuracy**: **92.63%**

5. Model Comparison Summary

Model	Accuracy (%)	Notes
Neural Network	97.90	Fast, accurate, scalable
SVM	97.92	Very accurate, slow on full dataset
Random Forest	97.08	Interpretable, robust without scaling
KNN	96.88	Simple, but slow during prediction
Logistic Regression	92.63	Fast baseline, linear decision boundary

Best Model Recommendation

Recommended Model: - Neural Network (Keras Sequential)

Reasons:

- High accuracy (97.90%)

- Fast inference and GPU-supported training
- Easily scalable and tunable
- Smooth production deployment with TensorFlow/Keras

If Deep Learning is not an option, then next option is to use **Random Forest** (97.08%) — robust, interpretable, no need for normalization or reshaping.

6. Challenges Faced and Solutions

Challenge	Solution	Reason
Input shape mismatches	Used reshape/flatten methods	Fit model input requirements
Large pixel range (0–255)	Normalized to [0–1]	Improve convergence and accuracy
Label format mismatch	One-hot used only for CNN	SVM/KNN require 1D labels
SVM/KNN slow training	Used data subsets during tuning	Reduce computational load
Default hyperparameters	Used GridSearchCV for tuning	Improved accuracy in RF/KNN

7. Conclusion

This project demonstrated that:

- Multiple models can effectively classify handwritten digits with high accuracy.
- Deep learning models like **Neural Networks** perform best overall.
- Classical ML models like **SVM** and **Random Forest** offer competitive accuracy with less complexity in setup.

Neural Network is recommended for deployment due to its speed, accuracy, and compatibility with modern APIs.

8. Future Work

- Try Convolutional Neural Networks (CNN) for even better feature extraction.
- Perform cross-validation for more robust evaluation.
- Deploy the final model using Flask or FastAPI with a simple web interface.