CSC-202L Artificial Intelligence D

Project Report

Submitted To:

Ma'am Shanfa Irum

Submitted By:

M.Abdullah Feroz 2023-SE-03

Mehboob Alam 2023-SE-07

Haniya Batool 2023-SE-29

University of Engineering and Technology New Campus, Lahore

Handwritten Digit Recognition System

1. Introduction

This report details the development and performance analysis of a Handwritten Digit Recognition System using multiple machine learning algorithms. The system classifies digits (0–9) from the Optical Recognition of Handwritten Digits dataset and includes a web-based frontend that allows users to draw digits and receive real-time predictions.

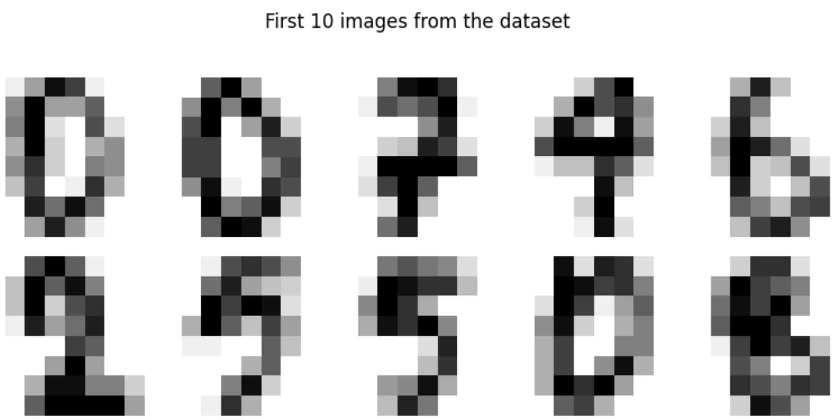
2. Dataset Preparation

Dataset Overview

- **Name:** Optical Recognition of Handwritten Digits (UCI ML Repository)
- **Instances:** 5,620
- **Features:** 64 (8×8 grayscale pixel images; pixel values range from 0–16)
- **Target Classes:** Digits from 0 to 9

Data Visualization

The figure below displays sample digit images from the dataset (first 10 samples):



Train-Test Split

- **Training Set:** 80% (4,496 samples)
- **Testing Set:** 20% (1,124 samples)

3. Preprocessing

Feature Scaling

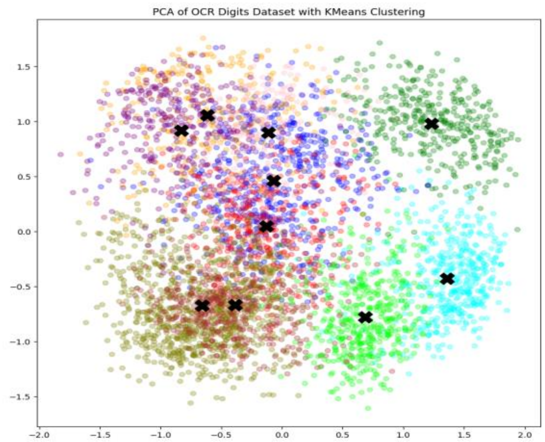
To improve model performance, the following scalers were tested:

- **MinMaxScaler:** Normalized pixel values to [0, 1]

4. Model Training & Evaluation

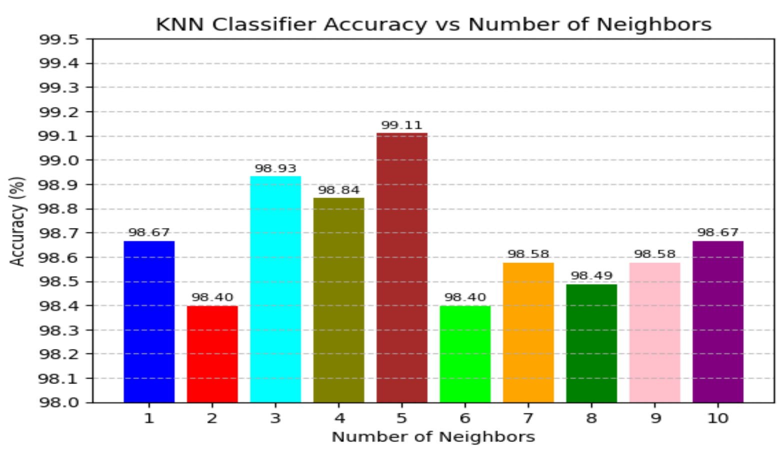
(A) K-Means Clustering (Unsupervised)

- **Objective:** Group digits into 10 clusters without labels
- **Metric Used:** Normalized Mutual Information (NMI)
- **NMI Score:** 71.32%
- **Visualization:**



(B) K-Nearest Neighbors (KNN)

- **Best Hyper parameter:** n_neighbors = 5
- **Accuracy:** 99.11%
- **Classification Report:**
-
    - **Precision:** 0.99
    - **Recall:** 0.99
    - **F1-Score:** 0.99



(C) Naive Bayes

- **Accuracy:** 80.77%
- **Classification Report:**
    - **Precision:** 0.85
    - **Recall:** 0.84
    - **F1-Score:** 0.84

(D) Logistic Regression

- **Hyper parameter:** C = 0.01, max_iter = 1000
- **Accuracy:** 96.71%
- **Classification Report:**
    - **Precision:** 0.97
    - **Recall:** 0.97
    - **F1-Score:** 0.97

(E) Support Vector Classifier (SVC)

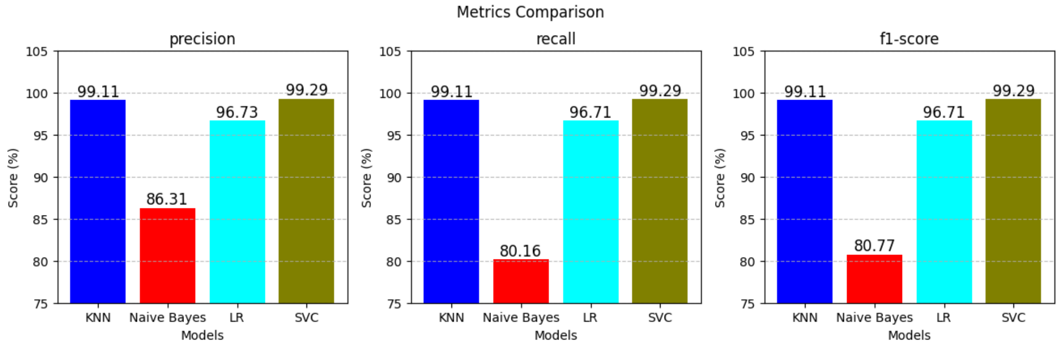
- **Best Kernel:** Polynomial (degree = 4)
- **Accuracy:** 99.29%
- **Classification Report:**
    - **Precision:** 0.99
    - **Recall:** 0.99
    - **F1-Score:** 0.99

5. Model Comparison

F1-Score Comparison Table

Model	F1-Score (%)
K-Means Clustering	71.32
K-Nearest Neighbors (KNN)	99.11
Naive Bayes	80.77
Logistic Regression	96.71
SVC (Poly Kernel)	99.29

Best Model: K-Nearest Neighbors (KNN)
Runner-Up: Support Vector Classifier (SVC, Polynomial Kernel)



6. Why We Used Logistic Regression ?

- **Simple and Interpretable**: Logistic Regression is one of the most basic and easy-to-understand machine learning models. It helps beginners (and teachers!) easily understand how the model is making decisions.
- **Good Baseline**: It's often used as a starting point to compare the performance of more advanced models. If a complex model doesn't perform much better than Logistic Regression, it might not be worth using.
- **Performs Well on Linearly Separable Data**: For problems where the classes (digits) can be separated by a straight line or surface, Logistic Regression performs very well.
- **Fast and Efficient**: Training Logistic Regression is faster and requires less computational power compared to other models like SVC or Random Forests.
- **Regularization**: It allows tuning (with the parameter **C**) to prevent overfitting, making it more flexible and robust.
- **In Our Case**: Logistic Regression achieved a high F1-Score of 96.71%**,** showing it was strong enough even for this image-based digit recognition task.

7. Why We Used Support Vector Classifier (SVC) ?

- **Handles Non-linear Data**: SVC is powerful because it can handle data that is not linearly separable which is often the case in image data like handwritten digits.
- **Kernel Trick**: We used a Polynomial Kernel with degree 4, which helped the model transform the data into higher dimensions to find a better separation boundary between digits.
- **Excellent Accuracy**: SVC achieved a very high F1-Score of 99.29%**,** making it one of the top performers in our project.
- **Works Well on Small to Medium Datasets**: Unlike deep learning models, SVC doesn't need massive datasets to perform well, making it ideal for our dataset of 5,620 samples.
- **Robust to Overfitting (with proper tuning)**: With careful tuning of parameters like C, degree, and kernel type, SVC can avoid overfitting and generalize well.
- **In Summary**: SVC was chosen for its powerful classification capability**,** especially when the data is complex and requires non-linear decision boundaries.

8. Frontend Implementation

A user-friendly web interface was developed to allow users to draw digits and get real-time predictions.

Technologies Used

- **Frontend:** HTML5 Canvas, JavaScript
- **Backend:** Python Flask (served the trained KNN model)

Key Features

1. Draw any digit (0–9) on the canvas.
2. Click the "Predict" button.
3. Instantly receive and view the model's prediction.

9. Conclusion

- The KNN model (n_neighbors = 5) achieved the highest accuracy 99.11%.
- The SVC with polynomial kernel closely followed with 99.29%.
- The system works effectively in both model performance and real-time user interaction.