JSPM's

RAJARSHI SHAHU COLLEGE OF ENGINEERING

TATHAWADE, PUNE-33

DEPARTMENT OF COMPUTER ENGINEERING

Case Study

on

Handwritten Digit Recognition using SVM and k-NN

– Submitted by –

Name of Student: Adarsh Gangshettiwar

Roll No: CS3274

PRN No: RBTL24CS145

Class and Division: 3rd year B Division

Course: Machine Learning

1. Title

Handwritten Digit Recognition using SVM and k-NN

2. Background/ Introduction

Handwritten digit recognition is a fundamental problem in computer vision and pattern recognition. It has applications in postal mail sorting, bank cheque processing, and digitizing handwritten documents.

Machine Learning algorithms such as Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN) can effectively classify handwritten digits by learning from labeled datasets. These algorithms analyze the pixel patterns of digits and map them to their corresponding numeric values.

In this project, we implement both SVM and k-NN classifiers on the MNIST dataset to compare their accuracy and evaluate which method performs better for digit recognition.

3. Problem Statement

The main challenge is to accurately classify handwritten digits (0–9) using machine learning models. The system should handle variations in handwriting styles and recognize digits efficiently with minimal error.

4. Objectives

- To use labeled handwritten digit data for training ML models.
- To implement SVM and k-NN classifiers for digit recognition.
- To evaluate model performance using accuracy and confusion matrix.
- To predict new handwritten digits with high accuracy.

5. Libraries required

- NumPy – For numerical operations
- Pandas – For data handling (optional)
- Scikit-learn – For machine learning models and evaluation metrics
- Matplotlib – For visualization (optional)

6. Approach/ Methodology

1. **Data Collection:**
   - Use the MNIST dataset or sklearn's built-in digits dataset (load_digits).
2. **Data Preprocessing:**
   - Flatten 2D images into 1D arrays.
   - Normalize pixel values (optional).
3. **Feature & Target Selection:**
   - Features: Pixel values of the digit images
   - Target: Digit label (0–9)
4. **Train-Test Split:**
   - Split the dataset into training and testing sets (80%-20%).
5. **Model Training:**
   - Train **SVM** and **k-NN** classifiers.
6. **Model Evaluation:**
   - Evaluate models using accuracy and confusion matrix.
7. **Prediction:**
   - Predict labels for new digit images.

7. Implementation

```
import numpy as np
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

# Load dataset
digits = load_digits()
X = digits.images.reshape((len(digits.images), -1))
y = digits.target

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

```
# SVM Classifier
svm_model = SVC(kernel='linear')
svm_model.fit(X_train, y_train)
y_pred_svm = svm_model.predict(X_test)

print("SVM Accuracy:", accuracy_score(y_test, y_pred_svm))
print("SVM Confusion Matrix:\n", confusion_matrix(y_test, y_pred_svm))

# k-NN Classifier
knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train, y_train)
y_pred_knn = knn_model.predict(X_test)

print("\nk-NN Accuracy:", accuracy_score(y_test, y_pred_knn))
print("k-NN Confusion Matrix:\n", confusion_matrix(y_test, y_pred_knn))

# Sample Prediction
sample_digit = X_test[0].reshape(1, -1)
print("\nActual Label:", y_test[0])
print("Predicted by SVM:", svm_model.predict(sample_digit)[0])
print("Predicted by k-NN:", knn_model.predict(sample_digit)[0])
```

8.GitHub Link:

https://github.com/AdarshG07/Digit_Recognition_System.git

9.Results

```
SVM Accuracy: 0.9777777777777777

SVM Confusion Matrix:

 [[33  0  0  0  0  0  0  0  0  0]

 [ 0 28  0  0  0  0  0  0  0  0]

 [ 0  0 33  0  0  0  0  0  0  0]

 [ 0  0  0 32  0  1  0  0  0  1]

 [ 0  1  0  0 45  0  0  0  0  0]

 [ 0  0  0  0  0 47  0  0  0  0]
 [ 0  0  0  0  0  0 35  0  0  0]
 [ 0  0  0  0  0  0  0 33  0  1]
 [ 0  0  0  0  0  1  0  0 29  0]
 [ 0  0  0  1  1  0  0  1  0 37]]
k-NN Accuracy: 0.9861111111111112
k-NN Confusion Matrix:
 [[33  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 33  0  0  0  0  0  0  0]
 [ 0  0  0 34  0  0  0  0  0  0]
 [ 0  0  0  0 46  0  0  0  0  0]
 [ 0  0  0  0  0 45  1  0  0  1]
 [ 0  0  0  0  0  0 35  0  0  0]
 [ 0  0  0  0  0  0  0 33  0  1]
 [ 0  0  0  0  0  0  0  0 30  0]
 [ 0  0  0  0  1  1  0  0  0 38]]
Actual Label: 6
Predicted by SVM: 6
Predicted by k-NN: 6
```

10.Conclusion

The Handwritten Digit Recognition project demonstrates how SVM and k-NN classifiers can be applied for pattern recognition tasks.

- SVM is highly accurate and handles high-dimensional data effectively.
- k-NN is simple and performs well for small datasets but may be slower on larger datasets.

This project highlights the importance of choosing appropriate machine learning algorithms for image classification tasks and provides a foundation for more advanced methods like deep learning in digit recognition.