

MNIST Classification using KNN Algorithm

Name- Priyanshu Kumar

Roll- 2511MC06

Brief about MNIST Dataset

The MNIST dataset (Modified National Institute of Standards and Technology) is one of the most widely used benchmark datasets in machine learning. It consists of **70,000 grayscale images** of handwritten digits from **0 to 9**, each of size **28 × 28 pixels**. Out of these, 60,000 images are used for training and 10,000 for testing. Since the digits vary in writing styles, the dataset provides a good challenge for algorithms to learn robust classification patterns.

Methodology

k-Nearest Neighbours (k-NN)

The k-Nearest neighbours (k-NN) algorithm is a **supervised learning method** that classifies a sample based on the majority class of its nearest neighbours. It works as follows:

1. Choose a value of **k** (number of neighbours).
2. For a given test sample, compute its distance (e.g., Euclidean distance) from all training samples.
3. Select the **k closest samples**.
4. Assign the label that is most frequent among these neighbours.

k-NN is simple, non-parametric, and often effective for image classification tasks like MNIST, where digit shapes have visual similarities. However, it can be computationally expensive for large datasets since it must calculate distances to many points.


Preprocessing Steps

Before applying the k-NN algorithm, preprocessing is necessary to improve performance:

1. **Normalization:** The raw pixel values range from 0 to 255. To ensure fair distance calculations, pixel values are scaled to the range **0–1** by dividing by 255.
2. **Flattening:** Each 28 × 28 image is reshaped into a **784-dimensional vector** so that it can be processed by k-NN.
3. **Train-Test Split:** The dataset is divided into a **training set** (used to fit the model) and a **test set** (used to evaluate the model's performance). A typical split is 80% training and 20% testing.

MNIST Classification using KNN Algorithm

Code snippet

```
[3]  import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split

# 1. Load MNIST dataset from TFDS
print("Loading MNIST dataset from TFDS...")
ds = tfds.load("mnist", split="train+test", as_supervised=True)

# Convert dataset to numpy arrays
X = []
y = []
for image, label in tfds.as_numpy(ds):
    X.append(image.reshape(-1)) # Flatten 28x28 → 784
    y.append(label)
|
X = np.array(X)
y = np.array(y)

print("Dataset shape:", X.shape, y.shape)


# 2. Normalize pixel values (0-255 → 0-1)
X = X / 255.0

# 3. Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

```
# 4. Train KNN model
print("Training KNN...")
knn = KNeighborsClassifier(n_neighbors=3, algorithm='auto', n_jobs=-1)
knn.fit(X_train, y_train)

# 5. Make predictions
y_pred = knn.predict(X_test)

# 6. Evaluate model
accuracy = accuracy_score(y_test, y_pred)
print("KNN Accuracy on MNIST:", accuracy)
```

```
 Loading MNIST dataset from TFDS...
Dataset shape: (70000, 784) (70000,)
Training KNN...
KNN Accuracy on MNIST: 0.975
```

MNIST Classification using KNN Algorithm