ML-MAJOR-JULY

Machine learning Project

MNIST-Digit Recognition

Table of contents:

1. Introduction
2. Problem Statement
3. Dataset Used
4. Tools used
5. Algorithm used
6. About the algorithm
7. How does the algorithm work
8. Steps involved
    - Importing Libraries
    - Data Collection
    - Data Analysis and Visualization
    - Plotting
    - Train & Test Split
    - Visualizing Samples
    - K-Nearest Neighbors(K-NN)
    - Accuracy
9. Conclusion

List of figures:

Figure1.1- sample digits used for training the classifier

Figure1.2- formula of Euclidean distance

Figure1.3-Heatmap

Figure1.4- Countplot for visualizing the number of class and counts

Figure1.5- Visualizing samples

Figure1.6- Visualizing samples

Figure1.7- Visualizing samples

Introduction:

The aim of this project is to implement a classification algorithm to recognize handwritten digits (0-9). It has been shown in pattern recognition that no single classifier performs the best for all pattern classification problems consistently. Hence, the scope of the project also included the elementary study the different classifiers and combination methods, and evaluate the caveats around their performance in this particular problem of handwritten digit recognition. This report presents our implementation of the classification algorithm called K-Nearest Neighbor (K-NN) to recognize the numeral digits.

Problem Statement:

The problem statement is to classify handwritten digits. The goal is to take an image of a handwritten digit from the MNIST dataset and identify it using any classification algorithms.

Dataset Used:

The MNIST database (Modified National Institute of Standards and Technology Database) is a large database of handwritten digit that is commonly used for

training various image processing systems. The database is also widely used for training and testing in the field of machine learning. The dataset chosen here contains 42000 handwritten digit samples.

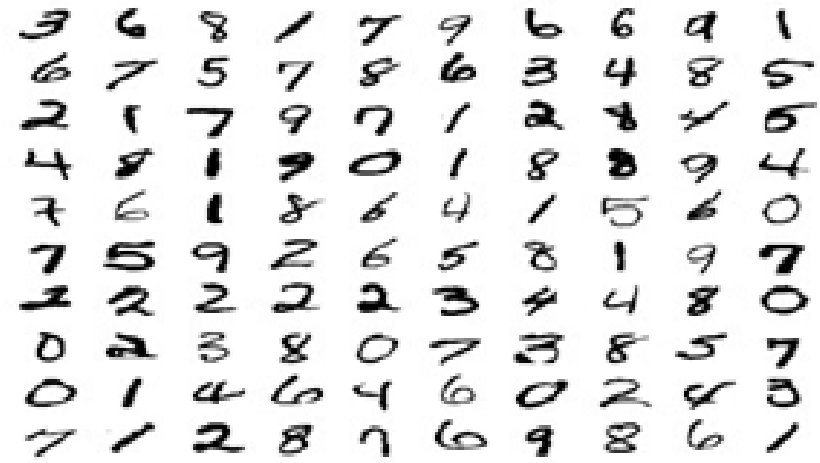

Figure 1.1-sample digits used for training the classifier

Tools Used:

Pandas

Numpy

Matplotlib.pyplot

Seaborn

from sklearn.model_selection import Train_test_split

from sklean.model_selection import confusion_matrix

Algorithm Used:

The algorithm used is K-Nearest Neighbor (K-NN), It is a simple classification algorithm that is very effective.

About K-NN algorithm:

K-Nearest Neighbor is one of the simplest classification algorithm which is very effective in nature that stores all the available cases and classifies the new data or case based on a similarity measure. As the (K-Nearest Neighbor) suggests it considers “k” Nearest Neighbors to predict the class or continuous value for the datapoint. K in K-NN stands for number of nearest neighbors. K-NN algorithm is also known as the “Lazy-Learner algorithm”.

How does K-NN work:

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**. Euclidean distance is the distance between two points.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Figure 1.2-formula of Euclidean distance

- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.

- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Steps Involved:

1. Importing the libraries: In this step we import all the necessary libraries in order to work with the problem. Some of the libraries are Pandas, Numpy, etc.

2. Data collection: In this step we fetch the dataset into the notebook.

3. Data analysis and Visualization: In this step we use different pre-defined functions which help us to analyse and visualize the data in a more simple and understanding manner. We used df.head(), df.info(), df.describe(), isnull().

4. Plotting: In this step we used different plotting functions to plot the images for the better understanding of the data.

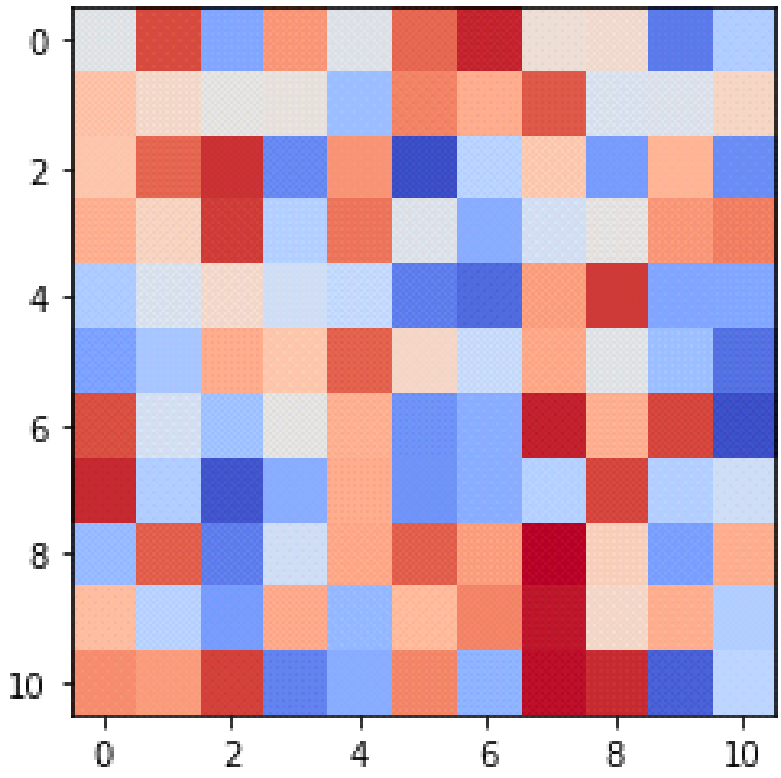

Figure 1.3- Heatmap

We also made a plot for visualizing the number of class and counts in the dataset.

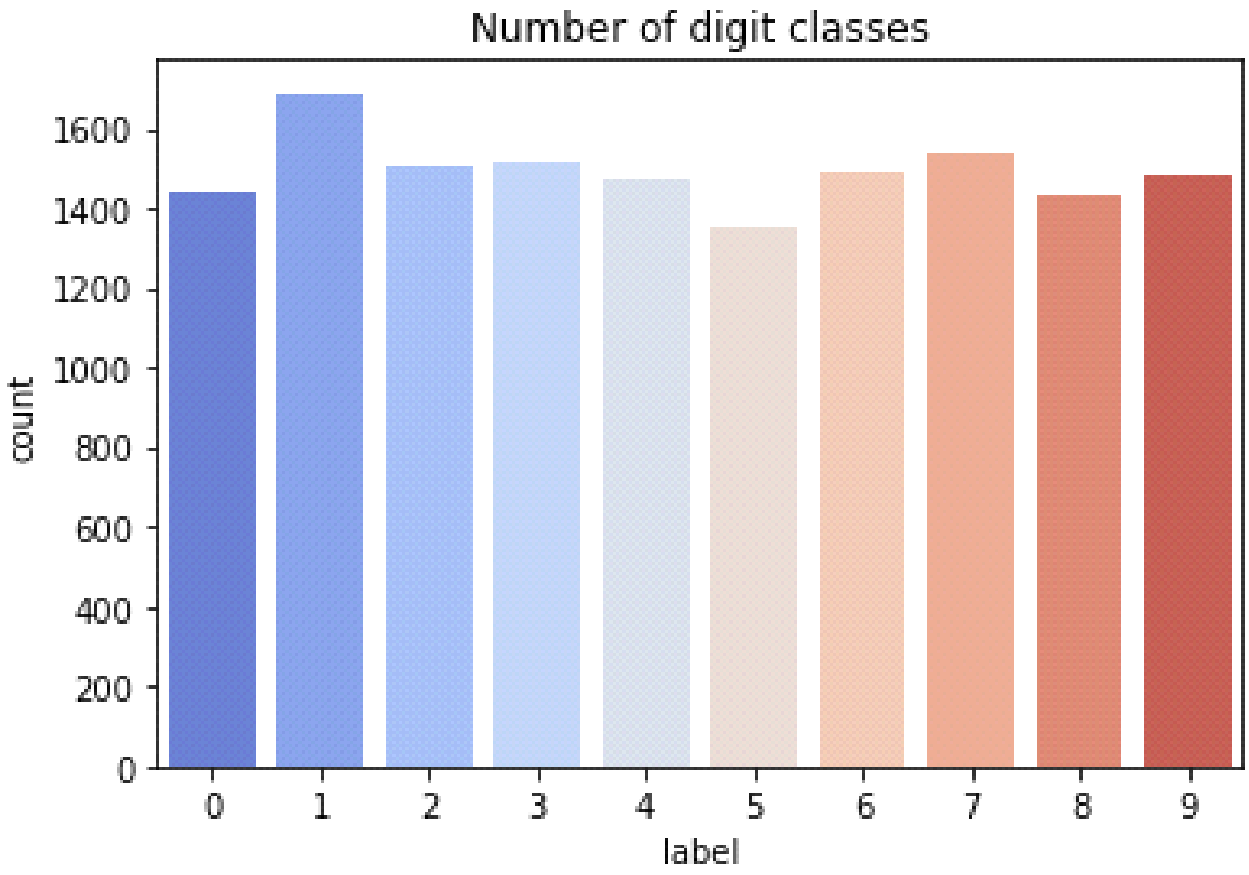

Figure 1.4- Countplot for visualizing the number of class and counts

5. Train and Test Split: In this step we use a procedure called Train-Test Split which is in Scikit-Learn library. So the function "train_test_split()" takes the loaded dataset as input and returns the dataset split into two subsets which are for training data and testing data. With this function we don't have to manually divide the dataset.

6.Visualizing samples: In this step we defined a function which can print samples of the digits from the given dataset.

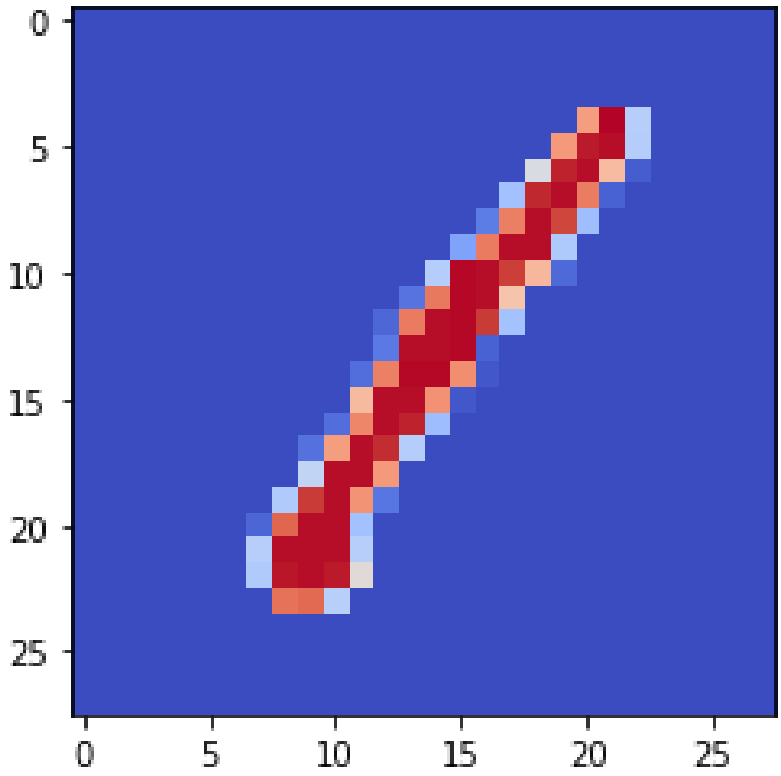

Figure 1.5-Visualizing samples

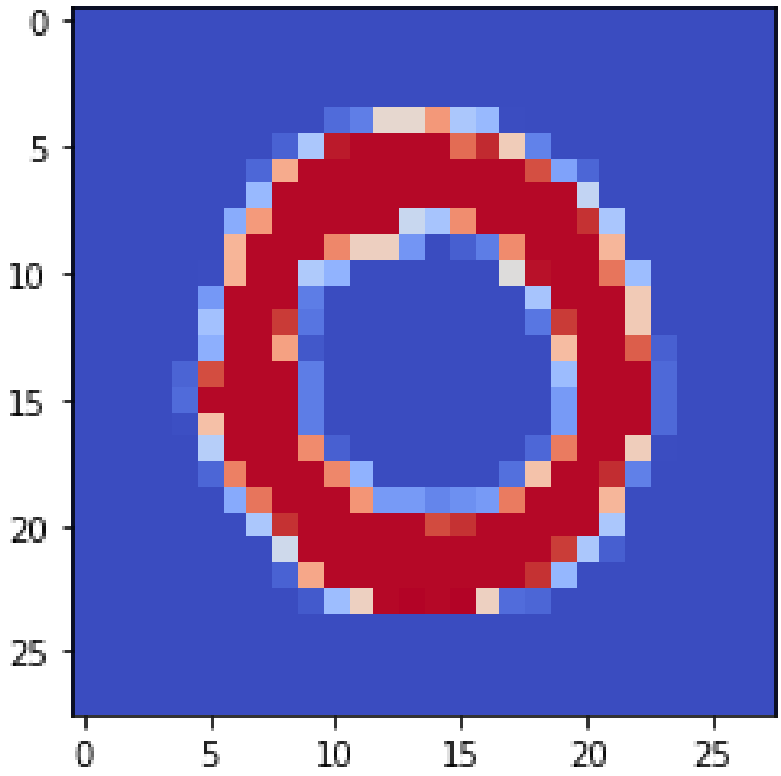

Figure1.6-Visualizing samples

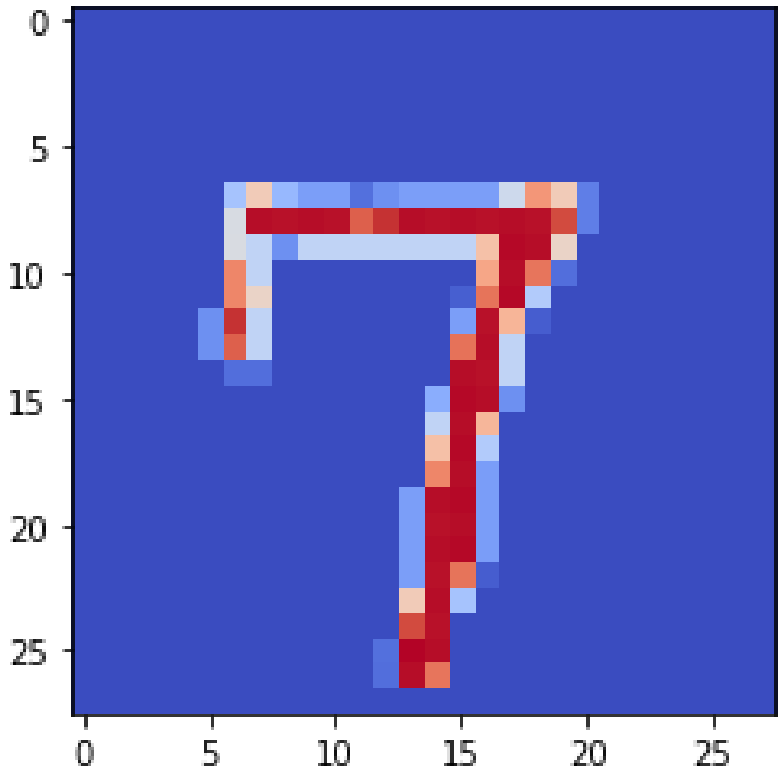

Figure 1.7- Visualizing samples

7.K-Nearest Neighbors (KNN): K-Nearest Neighbor is one of the simplest classification algorithm which is very effective in nature that stores all the available cases and classifies the new data or case based on a similarity measure. “K” in K-NN stands for number of Nearest Neighbors.

8.Accuracy: In this step we calculate the accuracy of the model. In this model we achieved an accuracy of 0.95

Accuracy Score: 0.95

Accuracy Score: It is the ratio of correct prediction to the total number of predictions.

Conclusion:

The proposed project shows the whole process of classification from the data acquisition to the design of a classification system and its evaluation. We learnt about the most simple machine learning classifier—the K-Nearest Neighbor classifier, or simply short for K-NN. The K-NN algorithm classifies unknown data points by comparing the unknown data point to each point in the training set. This comparison is done using a distance function(Euclidean distance). As explained in the text, some changes can be made on the representations of the images and on the distances used to compare two elements in the dataset. This project was implemented and executed by applying K-NN algorithm with accuracy score of 96%. The desired results have been obtained by training the machine with the given Modified National Institute of Standards and Technology database (MNIST).