Machine Learning, Spring 2023: Project 1

NIKHIL DUGYALA

MNIST DATASET

I have used the NumPy , matplotlib and Pandas packages, and I have used the Python programming language to implement this project.

**Dataset details:** MNIST dataset is a famous dataset of 70,000 small images of handwritten digits. Each image is 28x28 pixels and has a corresponding label indicating the digit it represents.

Shape, arrow

Description automatically generatedChart

Description automatically generated

Plot of first 11 images from each digit category Histogram of the pixel values

We can see the distribution of pixel intensities in the dataset from the above histogram.

I have done the training/test split by choosing the 80% of data for training and 20% of data for testing because it provides a good balance between having enough data to train a model effectively and having enough data to test the model's performance.

**Algorithm Description:**

As we need to preprocess the data. In this case, l have normalized the pixel values by dividing each pixel value by 255. This scales the values between 0 and 1, making it easier for the algorithm to learn.

Distance Metrics: I have used the Euclidean distance metric to compute the distances between the points.

**Algorithm Results:**

I have received the accuracy of 96.69% for the KNN algorithm using Euclidean distance with k=6.

****

The Confusion matrix will be a 10 x 10 matrix (since there are 10 possible digits) and it shows how many times each digit was classified correctly and how many times it was misclassified as another digit. The diagonal of the matrix represents the correct classifications, while off-diagonal entries represent misclassifications.

A picture containing text

Description automatically generated

Chart, line chart

Description automatically generated Chart

Description automatically generated

Accuracy of a single digit 7 as a function of K Accuracy of digit 9 with subsampled data(10%)

**Runtime:**

The runtime of the KNN algorithm depends on the size of the dataset and the value of k chosen.

The actual "wall-clock" time that it took to compute the results is 3524.27 seconds, as the data set is very large it took time to execute.



The runtime has reduced significantly with the subsampled data of 10%.

