Handwritten Digit Recognition

Report submitted in fulfillment of the requirements
for the Exploratory Project of

Second Year B.Tech.

by

Nikita Singh

20075060

Gargi Gupta

20074011

Under the guidance of

Dr. Hari Prabhat Gupta

Department of Computer Science and Engineering

INDIAN INSTITUTE OF TECHNOLOGY (BHU) VARANASI
Varanasi 221005, India
May 2022

Dedicated to:

My parents, professors, exploratory project convener, mentor and everyone who helped and motivated me in the successful completion of this project.

Declaration

I certify that

1. The work contained in this report is original and has been done by myself and the general supervision of my supervisor.
2. The work has not been submitted for any project.
3. Whenever I have used materials (data, theoretical analysis, results) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.
4. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Place: IIT (BHU) Varanasi
Date: 28th April, 2022

Nikita Singh and Gargi Gupta
B.Tech. and IDD Student Respectively
Department of Computer Science and Engineering,
Indian Institute of Technology (BHU) Varanasi,
Varanasi, INDIA 221005.

<u>Certificate</u>

*This is to certify that the work contained in this report entitled "**Handwritten Digit Recognition**" being submitted by **Nikita Singh and Gargi Gupta** (**Roll No. 20075060 and 20074011**), carried out in the Department of Computer Science and Engineering, Indian Institute of Technology (BHU) Varanasi, is a bona fide work of our supervision.*

Dr. Hari Prabhat Gupta
Department of Computer Science and Engineering,
Indian Institute of Technology (BHU) Varanasi,
Varanasi, INDIA 221005.

Place: IIT (BHU) Varanasi
Date: 28th April, 2022

Acknowledgments

I would like to express my sincere gratitude to my parents for supporting me in every aspect of this project, my professors for their constant guidance, my exploratory project convener, Dr. Hari Prabhat Gupta, for providing me with such an informative project which has helped me a lot in gaining knowledge about machine learning and everyone who has motivated in completion of this project.

Place: IIT (BHU) Varanasi	**Nikita Singh: 20075060**
Date: 28th April, 2022	**Gargi Gupta: 20074011**

Abstract

This report examines the performance of numerous classifier techniques on a database containing certain features with numerical values along with their corresponding digit values. We have used criteria like raw accuracy, precision, recall, f1-score, training time and execution time for comparing these classifiers. [1]

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc, and classify them into 10 predefined classes (0-9).[2]

1.2 Motivation of the Research Work

The subject of "Handwritten digit recognition" is of great concern and has many applications in various fields like zip code recognition.
Handwritten Digit Recognition poses many challenges as different people have different writing styles and it is not Optical Character Recognition.[2]

1.3 Organisation of the Report

We have implemented four classifier models from the scikit-learn library namely, K Nearest Neighbours(KNN), Polynomial SVM(support Vector Machine), Decision-tree and Multi Layer Perceptron.
Further, we have compared the performance of each of these classifiers on our test data set using various evaluation criteria like accuracy, training time, execution time, etcetera.

Chapter 2

Data

2.1 Dataset

The dataset contains digit-based Multivariate Time Series (MTS) data recorded with a sensor-equipped marker pen. The writer's hand movements are recorded using a 9-axis IMU sensor attached to the marker pen while writing a digit on a wall-mounted whiteboard. Three sensors make up the IMU: a 3-axis accelerometer, a 3-axis gyroscope, and a 3-axis magnetometer. These sensors are linked to a NodeMCU, which uses Wi-Fi to send data to a MQTT server. Sensory data is obtained at a sampling rate of 75 Hz from 20 volunteers. A writing activity is designed which consists of 5 rows where each row has 10 digits in ascending order [i.e., 0, 1, $\cdots$, 9]. . As a result, a single writing activity generates an MTS of 50 digits (i.e., each digit 5 times), which is later segmented to have a separate MTS for each of 50 digits. Each volunteer completes the writing exercise 10 times, resulting in a total of 50x10x20 (=10000) tagged cases in the dataset.[3]

2.2 Data Pre-processing

In order to improve the performance of classifiers on our dataset, we have standardized or normalized our data according to the algorithm used.
Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

Max-Min Normalization

$$x_{\mathrm{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Figure 2.1 Normalization

In standardization, the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

Standardisation (Z-score Normalization)

$$x_{\mathrm{stand}} = \frac{x - \mathrm{mean}(x)}{\text{standard deviation }(x)}$$

Figure 2.2 Standardization

Chapter 3

Classifier Models

3.1 K-Nearest Neighbours (KNN)

KNN tries to predict the correct class for the test data by calculating the distance between the test data and all the training points. Then select the K number of points which is closet to the test data. Hamming distance can be used as a distance metric default metric for KNN classification.

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

The KNN algorithm calculates the probability of the test data belonging to the classes of 'K' training data and class holds the highest probability will be selected.[4]
KNN is known as a lazy algorithm - because it just stores the data during the training time and doesn't perform any calculations. It doesn't build a model until a query is performed on the dataset. Also, KNN is considered a non-parametric method because it doesn't make any assumptions about the underlying data distribution[5]

In our implementation, K=2 gives the best accuracy for our dataset.

3.2 Polynomial-SVM

3.2.1 Basic Structure

The polynomial kernel is a kernel function commonly used with support vector machines(supervised learning models), that represents the similarity of vectors (training samples) in a feature space over polynomials of the original variables, allowing learning of non-linear models.[6] We have used sklearn to implement polynomial svm model in our given dataset.

Intuitively, the polynomial kernel looks not only at the given features of input samples to determine their similarity, but also combinations of these. When the input features are binary-valued, then the features correspond to logical conjunctions of input features.[6] For

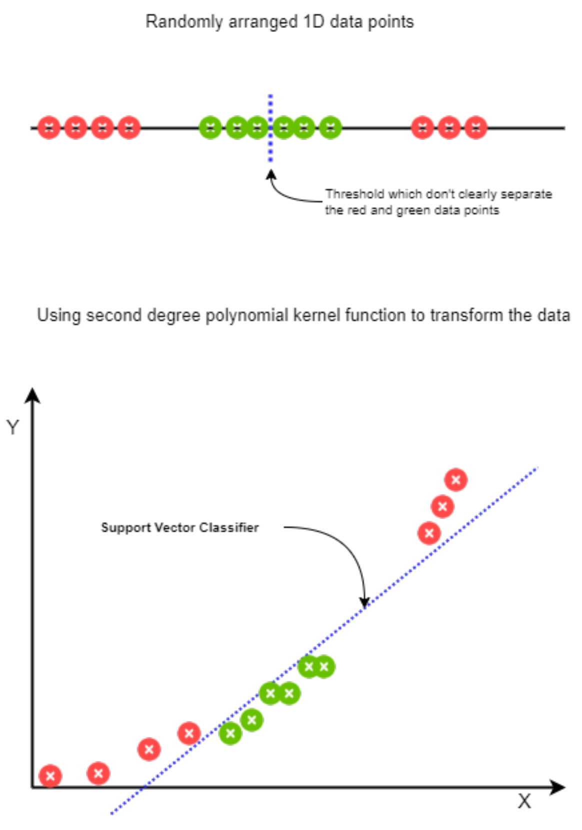

Randomly arranged 1D data points
Threshold which don't clearly separate the red and green data points
Using second degree polynomial kernel function to transform the data
Y
Support Vector Classifier
X

Figure 3.1 Polynomial-SVM

d-degree polynomial svm, our function is defined as:

$$K(x, y) = (x^{\mathsf{T}} y + c)^d$$

x,y being vectors in input space; c, a free parameter trading off the influence of higher versus lower order terms (when c = 0, the kernel is homogeneous).

3.2.2 Application

In our implementation, we have taken degree d=3, so that the polynomial kernel computes the 3D relation between each pair of observations which are used to find the support vector classifier.

3.3 Decision Tree Classifier

3.3.1 Basic Structure

Decision tree learning or induction of decision trees is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (a tree-like predictive model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.[7]

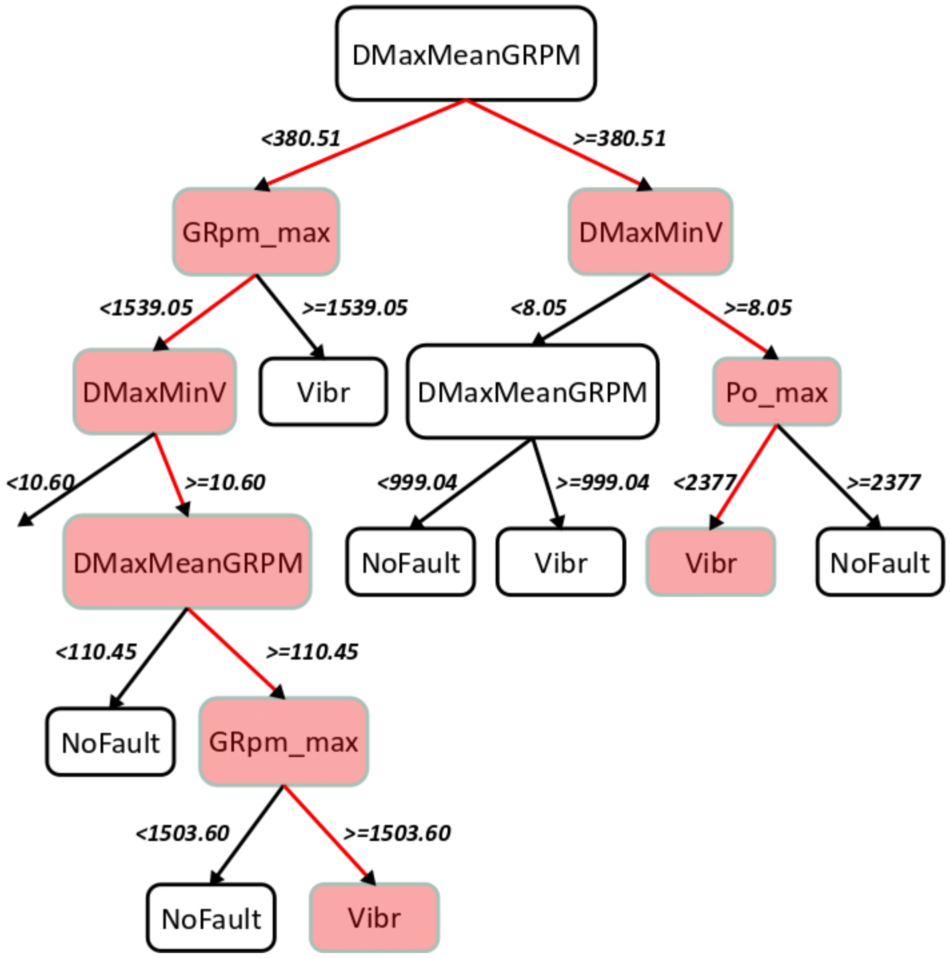
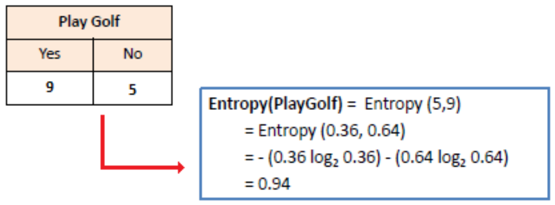


Figure 3.2 An example of a decision tree classifier

3.3.2 Application

In our implementation, we have given the criterion as entropy, which determines how a decision tree chooses to split data. If entropy is high, we are very uncertain about the random picked point of interest.

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$



Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
$= \text{Entropy } (0.36, 0.64)$
$= -(0.36 \log_2 0.36) - (0.64 \log_2 0.64)$
$= 0.94$

For information gain, we subtract entropy of parent node with the combined entropy of its child nodes.

3.4 Multilayer Perceptron (Neural Network)

3.4.1 Basic Structure

Multi layer perceptron (MLP) consists of three types of layers—the input layer, output layer and hidden layer. The input layer receives the input signal to be processed. The required task such as prediction and classification is performed by the output layer. An arbitrary number of hidden layers that are placed in between the input and output layer are the true computational engine of the MLP. [8]

3.4.2 Working

In an MLP, the data flows in the forward direction from input to output layer (feedforward neural network). The connections between the layers are assigned weights. The weight of a connection specifies its importance. The values of all the other neurons are calculated through a mathematical function involving the weights and values of the layer before it.

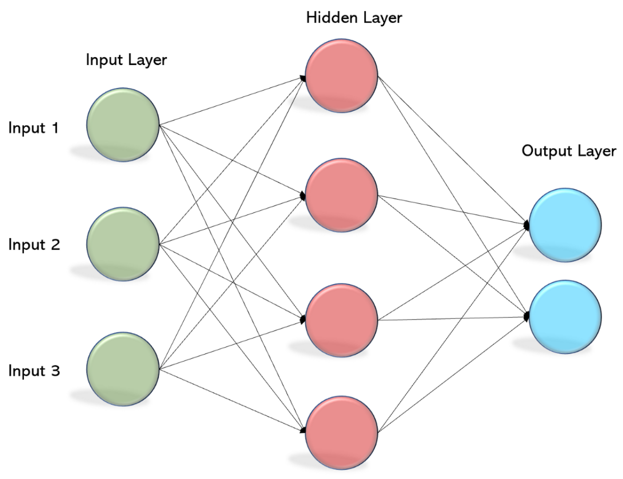


Figure 3.3 Multi-layer Perceptron Structure

The neurons in the MLP are trained with the back propagation learning algorithm. The weights are initialized with some random values which are used to predict an output. Then, the difference between the actual and predicted output (called error) is calculated and sent back through the network to readjust the weights such that the error is minimized. In this way, the weights are optimized using the outputs as inputs (hence called backpropagation).[9]

3.4.3 Applications

MLPs are designed to approximate any continuous function and can solve problems which are not linearly separable. The major use cases of MLP are pattern classification, recognition, prediction and approximation.[8]

In our implementation, we have used 3 hidden layers with the respective number of hidden units as 150, 100 and 50; and the sigmoid (logistic) function serves as the activation function. The Number of nodes in the input and output layers depend on the number of attributes and output classes in the dataset respectively, which are both 9 in our dataset. 500 iterations were performed to reduce the loss to 0.30

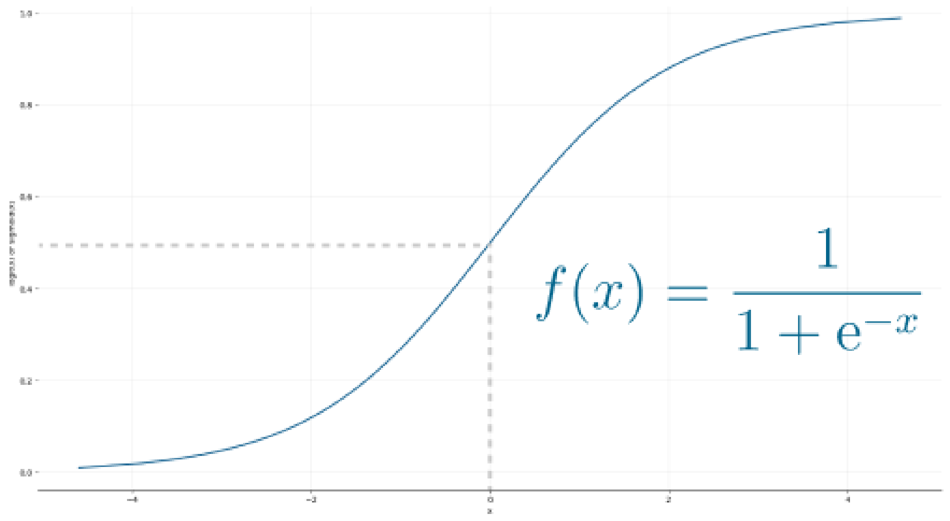


Figure 3.4 Sigmoid Function

Chapter 4

Performance Evaluation

4.1 Cross-validation (K-Fold)

Cross validation is used to estimate the performance of a machine learning algorithm with less variance than a single train-test set split. It splits the dataset into k-parts (folds). The algorithm is trained on k-1 folds and tested on the one remaining fold. This process is repeated so that each fold becomes the test set at least once. After running cross validation, we get k different performance scores which can be summarized using mean and standard deviation. It is more accurate because the algorithm is trained and evaluated multiple times on different data.[10] We have used k=5 for training models on our dataset.

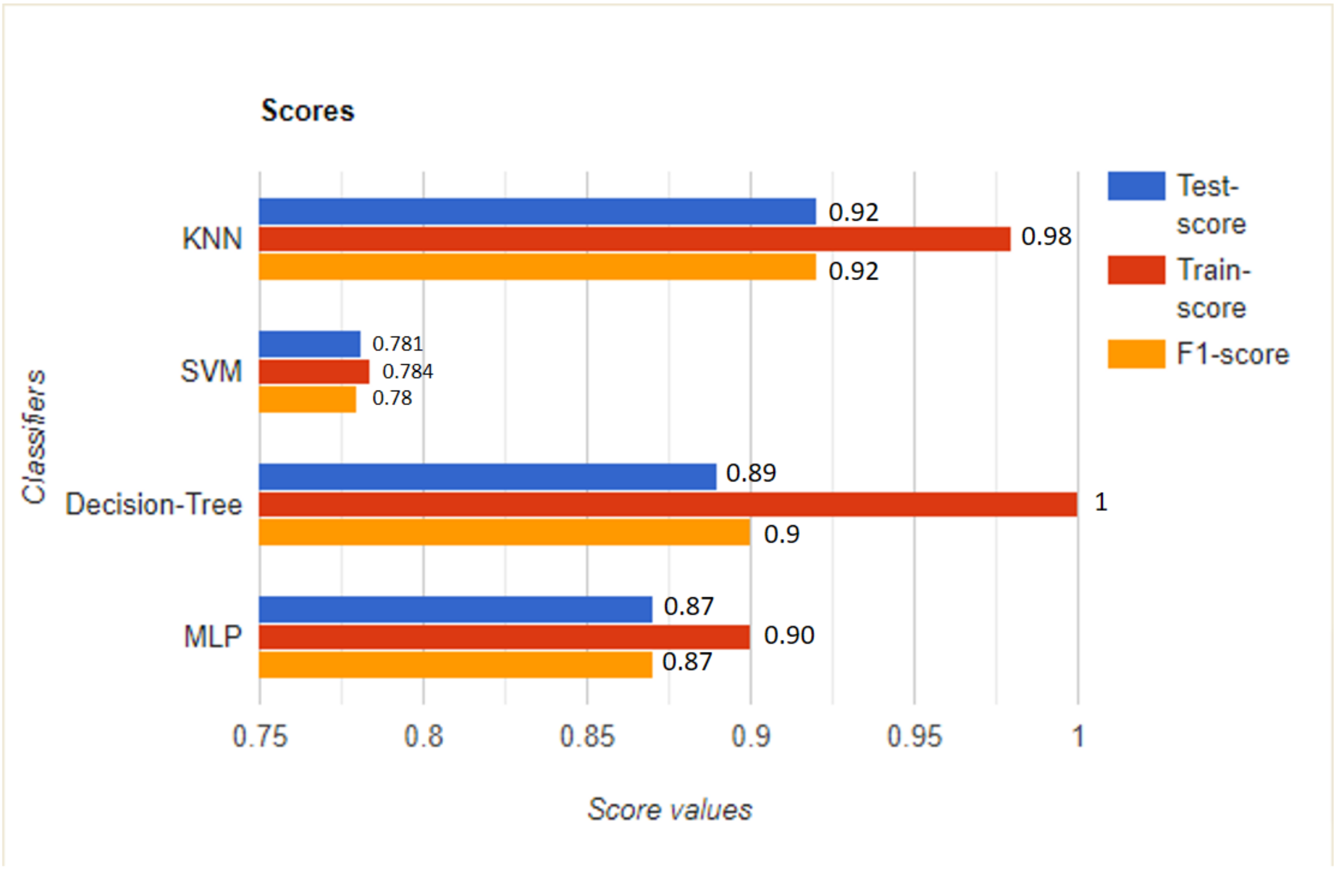
4.2 Performance Metrics

The performance Metrics used during evaluation and comparison of the above discussed classifier models are:

- Accuracy
- F1-score (which encompasses recall and precision)
- Training time
- Test/Execution/Prediction time

Chapter 5

Conclusions and Discussion



As we can see from the above graph, KNN gives the best accuracy on the test set whereas Decision tree gives the best accuracy on training set. Also, KNN has achieved the maximum F1 score.

In terms of Training and Prediction time,

- Multi-layer Perceptron took a training time of 72 minutes and prediction time of 1.58 secs
- Decision tree classifier takes prediction time of 0.05 seconds. Its training time is 10.39 seconds.
- Polynomial SVM took the maximum training time of 188.31 seconds and prediction time of 44.74 seconds.
- KNN gives us the training time 1.08 seconds and prediction time 17.53 seconds.

MODEL	TRAINING TIME	PREDICTION/TEST TIME
KNN	1.08 secs	17.53 secs
SVM	188.31 secs	44.74 secs
Decision Tree	10.39 secs	0.05 secs
MLP	72 mins	1.58 secs

Table 5.1 Comparison of Training and testing times for each model

Therefore, we can observe that:

- KNN gives the best (least) training time and
- Decision Tree gives the best (least) prediction/testing time.

References

[1] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: a case study in handwritten digit recognition," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5)*, vol. 2, 1994, pp. 77–82 vol.2.

[2] S. Pashine, R. Dixit, and R. Kushwah, "Handwritten digit recognition using machine and deep learning algorithms," *arXiv preprint arXiv:2106.12614*, 2021.

[3] A. Gupta, H. P. Gupta, T. Dutta, R. Mishra, and R. Kumar, "Accelerometer, gyroscope, and magnetometer sensors based data for recognizing handwritten digits," 2020. [Online]. Available: https://dx.doi.org/10.21227/52d9-8y30

[4] "K-nearest neighbor. a complete explanation of k-nn — by antony christopher — the startup — medium," https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.

[5] "What is k-nearest neighbor? an ml algorithm to classify data," https://learn.g2.com/k-nearest-neighbor.

[6] "Polynomial kernel," https://en.wikipedia.org/wiki/Polynomial_kernel.

[7] "Decision tree classifier," https://www.sciencedirect.com/topics/computer-science/decision-tree-classifier.

[8] "Mutilayer perceptron- an overview," https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron.

[9] "What is a multi-layered perceptron?" https://www.educative.io/edpresso/what-is-a-multi-layered-perceptron.

[10] "A gentle introduction to k-fold cross-validation," https://machinelearningmastery.com/k-fold-cross-validation/.