Performance Analysis of Handwritten Digits Recognition on Five Classification Algorithms

Sudarshanan Muthukumar, G Udaykiron Chowdary, Vandana Vishnuraj, Vignesh Raj T.S.R and Vineet Kumar Gupta

Department of Information Technology
SSN College of Engineering
Kalavakkam, Chennai, India

{sudarshanan2010803, gudaykiron2010902, vandhana2010750, vigneshraj2010388 & vineetkumar2010502}@ssn.edu.in

***Abstract* – This paper aims to analyze the performance variances of 5 Classification algorithms across Machine Learning and namely, k-Nearest Neighbours, Support Vector Machine, Random Forest Classifier and Multi-Layer Perceptron on the Handwritten Digits dataset. Analysis is done based on the accuracy each model provides, defined as the percentage of correct classifications done by the model on the test subset/validation subset of the dataset.**

Index Terms – Handwritten Digits Recognition, Exploratory Data Analysis, k-Nearest Neighbours, Support Vector Machine, Random Forest Classifier and Multi-Layer Perceptron

I. HANDWRITTEN DIGITS – AN OVERVIEW

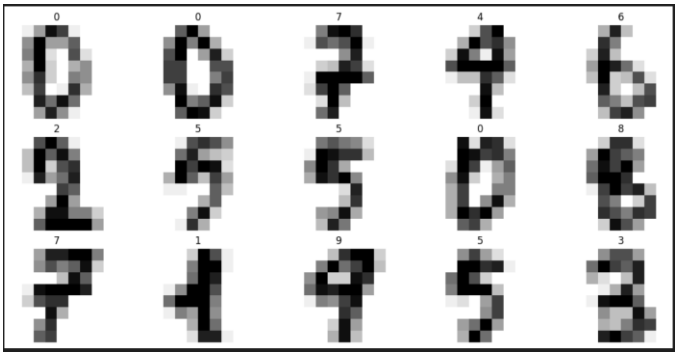
Handwritten Digits dataset contains a collection of 8x8 pixel images of handwritten digits (0-9). It consists of a total of 5620 samples, where each sample is a grayscale image. Each image is represented as an array of 64 numerical features, where each feature represents the intensity of a pixel. The pixel intensities range from 0 to 16, with 0 indicating white and 16 indicating black. 4,496 records are used to train and 1,124 records to test the model. The records contain pixel values of each image along with the label.

A. Content

Each image is 8 pixels in height and 8 pixels in width, for a total of 64 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 16. The training and test data sets have 65 columns. The first column consists of the class labels, and represents the value of digit. The rest of the columns contain the pixel-values of the associated image.

1) Pixel Location: To locate a pixel on the image, suppose that we have decomposed x as x = i * 8 + j, where i and j are integers between 0 and 8. The pixel is located on row i and column j of a 8 x 8 matrix.

2) Labels: Each training and test example is assigned to one of the following labels: 0 – represents digit 0, 1 – represents digit 1, 2 – represents digit 2, 3 – represents digit 3, 4 – represents digit 4, etc. up to, 9 – representing digit 9



In our dataset, image data has been converted to Comma Separated Value (CSV) data, with individual CSV columns corresponding to a pixel intensity value.

II. EXPLORATORY DATA ANALYSIS

A. An Overview

EDA is primarily used to see what data can reveal beyond the formal modelling or hypothesis testing task and provides a better understanding of dataset variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate. Originally developed by American mathematician John Tukey in the 1970s, EDA techniques continue to be a widely used method in the data discovery process today.

.

B. EDA performed for Handwritten Digits

Types of Exploratory Data Analysis that are performed in this paper are varied:

1) *Checking the dimensions of an element in the dataset:* We check the size of each image in the dataset and the number of images present in the training/testing set. By doing this, we ensure data completeness, data quality, and the structure of data. Doing this is essential, since we may need to transform/reshape the data.
2) *Description of the Dataset:* Various mathematical parameters like count, mean, std, min, first quartile, second quartile, third quartile and max are used to understand the dataset.

3) *Handle missing values:*
We need to keep track of the missing values in the dataset so as to either remove or replace those values. Here, there are no missing values.

4) *Visualize pixel value distribution:*
Pixel is the smallest element of an image. Each pixel correspond to any one value. In a 4-bit grayscale image, the value of the pixel between 0 and 16. The value of a pixel at any point correspond to the intensity of the light photons striking at that point. We find out the pixel value that has the biggest frequency of occurrence in the dataset images. First 15 images are displayed here.

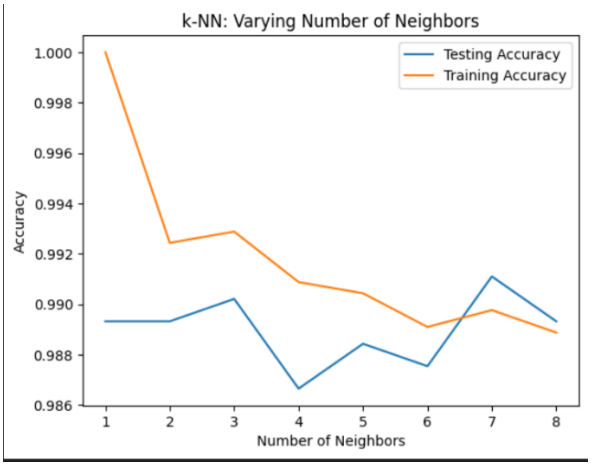
5) *Plot the confusion matrix:*
A confusion matrix is a matrix that summarizes the performance of a machine learning model on a set of test data. It is often used to measure the performance of classification algorithms. For our multi-classification, the matrix shape will be equal to the number of classes i.e., for 10 classes it will be 10X10.

C. *Methodology*

1. K-Nearest Neighbours

The k-NN algorithm operates on the principle that similar instances or data points tend to have similar labels or values. In classification tasks, it assigns a class label to an unlabelled data point based on the class labels of its k nearest neighbors.

To calculate the optimal k value, find the training and testing accuracy for k values from 1 to 9. Based on these values, optimal values is selected as 7 from the below image.



Accuracy and precession exceed 99 %. Even recall score and F1-score exceed 99%.

Recall = TP / (TP + FN)

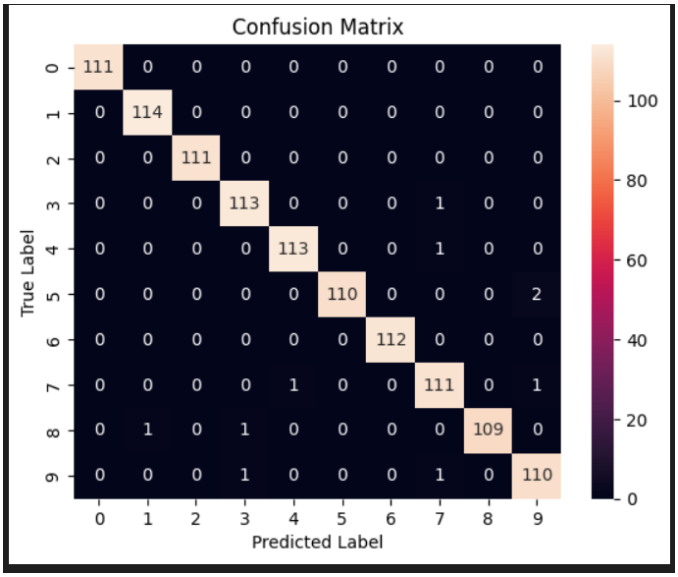
TP (True Positives), FP(False Positives)

The F1 score is calculated using the following formula:

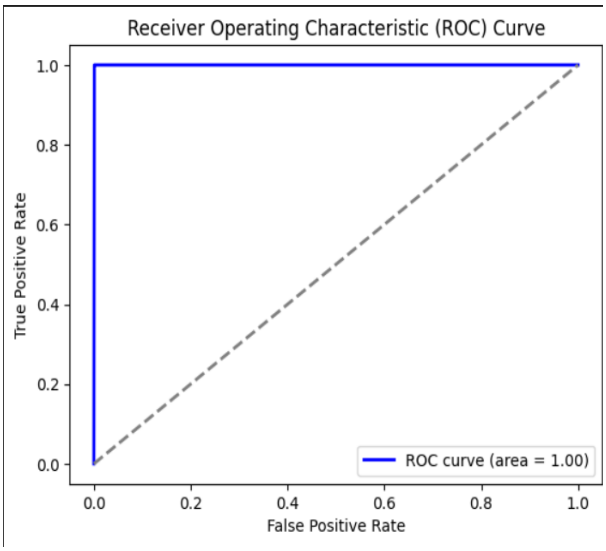
F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

Accuracy measures the overall correctness of the model's predictions. Precision provides insights into the model's ability to avoid false positive errors. A high precision indicates that the model has a low rate of incorrectly classifying negative instances as positive. Recall, also known as sensitivity or true positive rate, is a metric used to evaluate the performance. It indicates the model's ability to capture the positive instances and avoid false negatives. The F1 score is especially useful in scenarios where both precision and recall are important and need to be balanced. This provides a balance between these two metrics and is especially useful when the dataset is imbalanced.

These above scores affirm the fact that knn is very much suited for the optical digits recognition.



The heatmap plot helps to highlight patterns and trends in the confusion matrix, making it easier to interpret the model's performance. It allows you to visually identify areas where the model is performing well (high counts of true positives and true negatives) and areas where it may be making errors (high counts of false positives and false negatives).



ROC with k-NN allows you to evaluate the model's performance in terms of its ability to differentiate between positive and negative instances at different classification thresholds.

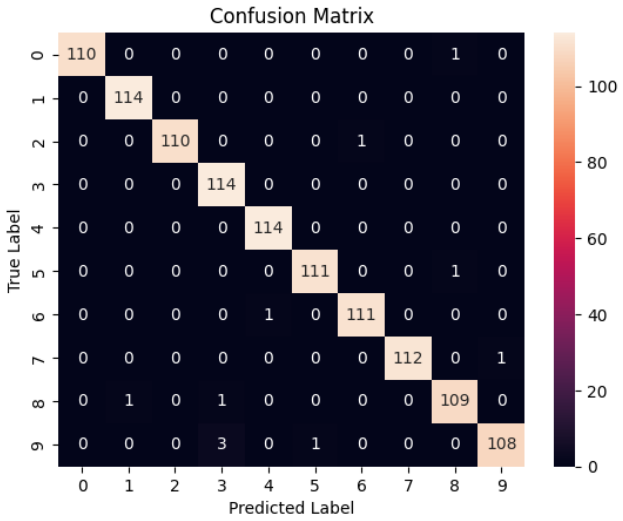
2. Support Vector Machine (SVM)

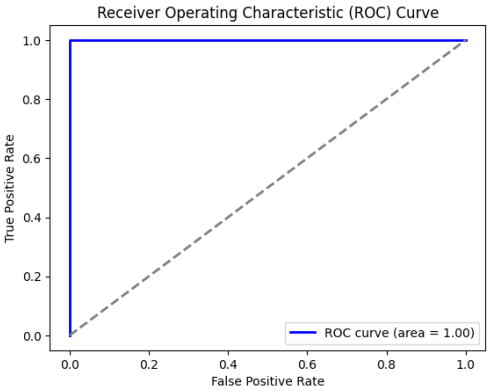
Support Vector Machine is an algorithm which is used in machine learning for classification. It is regularly used as the classification techniques due to its efficiency when compared with the other algorithms. This technique plots a hyperplane for every attribute as a coordinate that is present in the dataset. Classification is performed by identifying the hyperplane that divides one class with the other class.

It is a machine learning technique for classification techniques which as a group of weak prediction models that are like that of decision trees.

The score of the dataset using SVM is about 0.991, which indicates that it is a very efficient algorithm for classification.

We have run the SVM algorithm on our dataset with various kernel functions like linear, radial base, sigmoid and poly, based on the measure of accuracy, we conclude that the poly kernel function of SVM gives us the best accuracy score of about 0.99.
Below is the confusion matrix and the ROC curve generated using SVC algorithm. The confusion matrix gives the inference that 1,113 samples are correctly predicted.





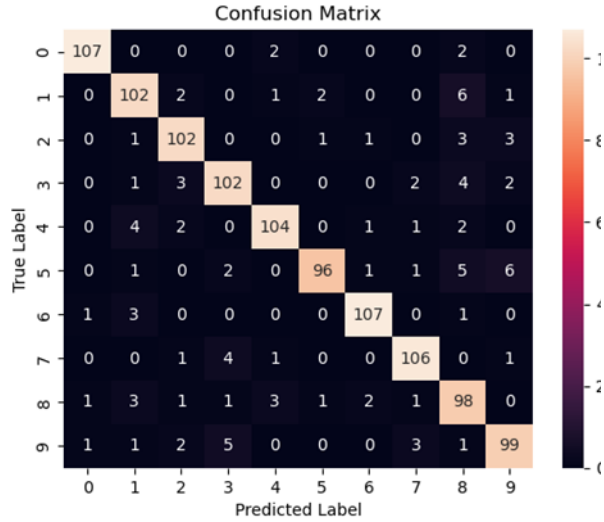
3. Decision tree

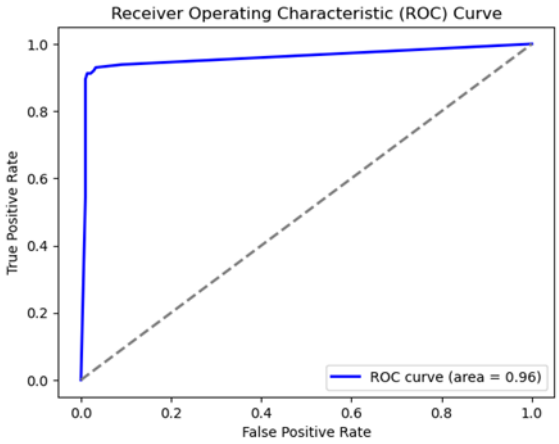
A decision tree is essentially a tree-like model where each internal node represents a test on a feature, each branch represents the outcome of the test, and each leaf node represents a class label or a numerical value. In a classification problem, decision trees partition the feature space into a set of disjoint regions, with each region corresponding to a specific class label. In a regression problem, decision trees estimate the output value by recursively partitioning the feature space and fitting a constant value to each leaf node.

Decision trees are attractive because they are easy to interpret, they can handle both categorical and numerical features. However, they can be prone to overfitting, especially when the trees are deep and complex. To mitigate this, various regularization techniques can be used, such as pruning or setting a maximum depth for the tree.

In the dataset, we find that the max-depth of the decision tree is 11, for giving the best accuracy score.

We have used Gini index values for building the decision tree.
The accuracy score for the decision tree is 0.91. Below is the confusion matrix and ROC curve generated for the decision tree.





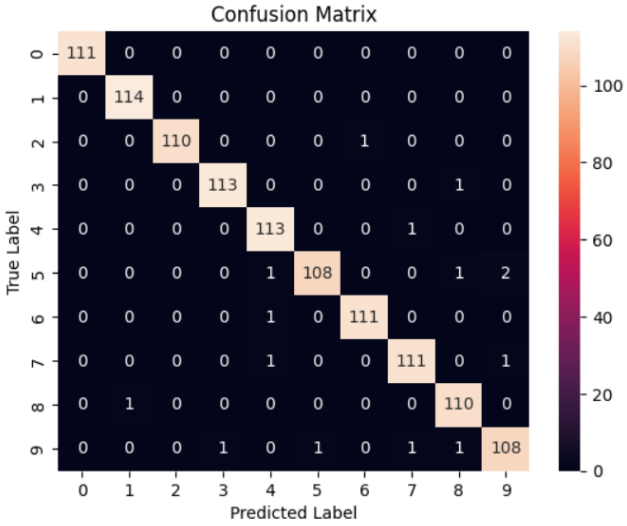
4. *Random Forest*

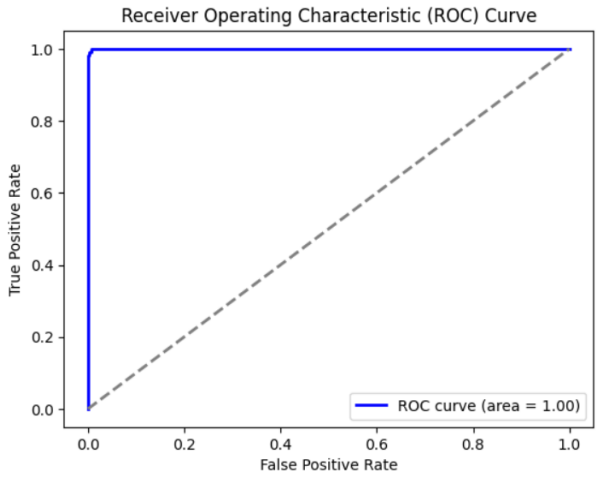
Random Forest is an ensemble learning algorithm that combines multiple decision trees to improve the accuracy and robustness of the classification or regression models. Here is a brief overview of how the algorithm works.

The algorithm starts by randomly selecting a subset of the original data for each tree in the forest.
For each tree, the algorithm builds a decision tree on the selected subset of the data by recursively partitioning the data based on the values of the input features.
At each node of the tree, the algorithm selects a random subset of the input features to use in the partitioning. This helps to prevent overfitting and improve the generalization of the model.

Once all the trees in the forest have been built, the algorithm uses them to make predictions on new input data. For classification tasks, the algorithm aggregates the predictions of each tree to make a final prediction by majority voting. For regression tasks, the algorithm aggregates the predictions of each tree to make a final prediction by taking the average.

Random Forest is a powerful and widely used algorithm for both classification and regression tasks. It is known for its ability to handle high-dimensional data, noisy data, and data with missing values. It is also relatively fast to train and can be parallelized easily, making it suitable for large-scale applications.





5. *Multilayer Percepton*

Multilayer perceptron (MLP) is a type of artificial neural network that is widely used in machine learning and pattern recognition. It consists of multiple layers of interconnected nodes, where each node performs a simple

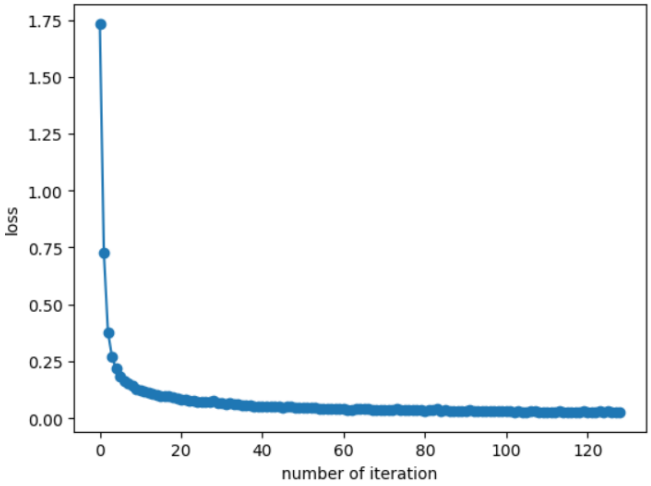
mathematical operation on its inputs and passes the result on to the next layer.

The input layer of an MLP receives the data to be processed, while the output layer produces the network's final output. The layers in between are called hidden layers, and they perform complex computations on the data to extract useful features and patterns.

In a multilayer perceptron, the connections between nodes are weighted, and these weights are adjusted during the training process to optimize the network's performance. This process typically involves minimizing the error between the network's predicted output and the actual output for a given set of training examples.

In this dataset we have used 64 input nodes with one node per pixel in the input images. Then send it to the two hidden layers of 100 and 50 neuron and finally from hidden layer it is sent to output layer with 10 neurons corresponding to our 10 classes of digits from 0 to 9.If we add more hidden layers there can also be risk of overfitting of training data. So Two hidden layers can give more accuracy and prediction.

Next, the data is divided into training and testing sets. The training set is used to train the MLP by adjusting the weights of its connections to minimize the error between the predicted output and the actual outputOnce the MLP has been trained, it can be used to predict the digit represented by a new image. To do this, the image is first preprocessed in the same way as the training data, and then fed into the MLP. The output of the MLP is a vector of probabilities that the input image represents each of the ten possible digits. The digit with the highest probability is then taken as the MLP's prediction for the input image.Overall, MLP is a powerful algorithm that can be used to achieve high accuracy on the opt digits dataset. However, its performance can be affected by factors such as the choice of features and the size and architecture of the MLP itself.



CONCLUSION

In conclusion, we have explored two popular machine learning algorithms, namely Multilayer Perceptron (MLP) and Random Forest, for classification of the optdigits dataset. Both algorithms achieved high accuracy scores on the dataset, with the MLP achieving an accuracy score of around 97% and the Random Forest achieving an accuracy score of around 97.5%.

We also studied the effect of changing the number of hidden layers in the MLP and found that adding more hidden layers did not always improve the performance of the model, indicating the importance of finding the right balance between model complexity and performance.

Overall, the optdigits dataset provides a good benchmark for evaluating the performance of classification algorithms, and our experiments demonstrate the effectiveness of MLP and Random Forest for this task. The results of this analysis can be extended to other classification tasks and datasets with similar characteristics, such as hand-written digit recognition, medical image classification, and more.

REFERENCES

https://scikit-learn.org/stable/datasets/toy_dataset.html#optical-recognition-of-handwritten-digits-dataset

https://medium.com/codex/recognizing-handwritten-digits-with-scikit-learn-90ca6e2471ed

https://towardsdatascience.com/the-best-machine-learning-algorithm-for-handwritten-digits-recognition-2c6089ad8f09

https://www.kaggle.com/code/paraspatidar/optical-recognition-of-handwritten-digits-dataset

https://datahub.io/machine-learning/optdigits