Handwritten Digit Recognition - A comparison between efficiency of different classifiers on Handwritten Digits

Tejal Bagade
txb170830@utdallas.edu

Pranita Hatte
prh170230@utdallas.edu

Yatin Muniswamy Prakash
yxm171830@utdallas.edu

Abstract

The aim of this project is to compare the accuracy of different classifiers on handwritten digits dataset. The project includes four models, namely Decision Tree, Support Vector Machine, Deep Learning and Neural Networks to classify the trained data (Handwritten Digit Dataset) and use them to predict the handwritten digits present in test dataset. The report provides a comparative study of the results. This project is implemented using the scikit-learn package on Python.

Keywords: Decision Tree, Support Vector Machine, Deep Learning and Neural Networks.

1. Introduction

Machine learning involves the creation of a model that can automatically learn and improve from experience. The process of learning begins with observations or data, such as examples, direct experience, or instruction, to look for patterns in data and make better decisions in the future based on the examples that we provide.

In this project we perform a thorough processing and analysis of a dataset using Machine Learning techniques. We have chosen Digit Recognizer from the Kaggle as our target and MNIST dataset for the purpose of training/testing the models. The accuracy of the models can be evaluated by uploading the output on Kaggle.

2. Problem Definition and Algorithm

2.1 Task Definition

The Digit Recognition problem involves the creation of a machine learning model which efficiently identifies a digit on the test dataset, which corresponds to the image of the hand-written digit. To create the model, we train the model with the classification approach using the training dataset.

Here, the training and test datasets are csv files representing the image in the form of pixels. They take the a value in the range from 0 to 155. The image size is 28 X 28 pixels. In this problem, our goal is to correctly identify digits from a dataset of tens of thousands of handwritten images. We’ve experimented with different algorithms to learn first-hand what works well and have compared the techniques based on their accuracy. The image below is a sample of the MNIST dataset.

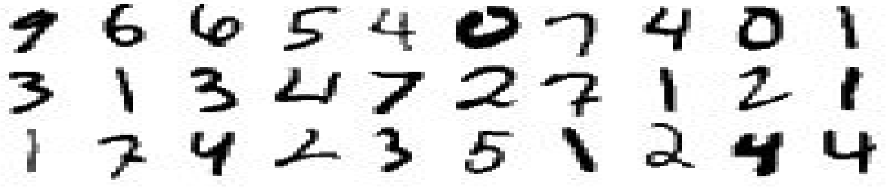


Figure 1: Handwritten digit example

2.2 Algorithm Definition

2.2.1 Artificial Neural Networks

In machine learning, Neural Network is a supervised learning model which is based on a collection of connected units called neurons. Each connection between the neurons has its own weights which varies on each step of learning process.

Here, all the neurons are arranged into layers. There are different layers which perform different tasks on their inputs. The layers are categorized into: input layer, hidden layer and output layer. Neural networks can efficiently classify the non-linear data. It has distributed control that makes it more powerful.

For Digit Recognition, we apply pre-processed data to the Neural Network, provide the initial weights to each neurons of the network and define the number of hidden layers, hidden layer size and the learning rate.

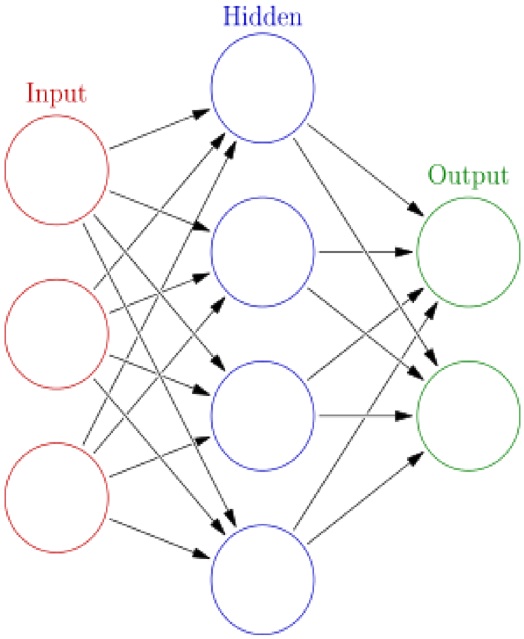


Figure 1: Neural Net

At each iteration to minimize the error we must update the weights of the neurons. To achieve this, we use the Back-Propagation Algorithm, in which we have two phases: forward pass and backward pass. Using this process, we find the weights for which the error will be minimum globally.

2.2.2 Decision Tree

In the Decision tree method, the learning approach is to make a decision-tree from the given training data (observations) to predict the target class of the test data.

The leaves of the decision tree represent the class labels and branches represents the conjunctions of the features. We focus on the order of the attributes used in the decision tree which gives us more information gain.

At each level of the tree, we find the best split of the data which has least entropy and error. To find this feature, which gives us the best split, we use ID3 algorithm. We apply the ID3 algorithm at each leaf node until the training data is perfectly classified. After that we apply test data to the decision tree model and we try to best fit the data, to classify with minimum error. We can also apply pruning to the decision tree to reduce overfitting and make model more general.

2.2.3 Support Vector Machine

In machine learning, SVM is a kind of supervised learning model with ability to analyse data used for classification and regression analysis. It is generally used for binary classification, but it works efficiently on multi-class classification with non-linear data. It also gives better results for image recognition techniques. SVM uses the kernel trick in which it uses a method to map the inputs into the higher dimensional feature spaces.

In practice, Multi-Class Classification problems (where number of class (n) > 2) generally decompose into a series of binary classification problems. Our aim is to find best split between each class with larger margin between them. In our case we have 10 classes, so we consider one at a time and rest of the 9 classes will be considered as another separate class.

2.2.4 Deep Learning

Deep learning is a method of making model more accurate and more efficient. In Context of neural network, deep learning means to create a network with large number of hidden layers to make it deep and thin which gives the best results.

In deep learning, we first focus on the basic model classifier with large number of training examples at the initial hidden layer (1st hidden layer). After that each layer will use the preceding layers to build a strong classifier.

3 Experimental Evaluation

3.1 Methodology

Pre-processing generally involves implementing the following on the dataset:

a) Checking the dataset for null or missing values.
b) Cleansing the dataset of any wrong values.
c) Standardizing the feature.
d) Converting any nominal (or categorical) variables to numerical form.

We use pandas and NumPy to pre-process the data as these libraries provide various tools to handle the data and to standardize it.

In the training dataset of the Digit Recognition problem, we notice that the data is in already in the numerical form. Hence, no encoding is required. Also, this set does not contain any null or missing values. Thus, we just need to focus on standardizing the features.

We have used following scaling techniques:

1) **Standard Scaling**: This technique standardizes features by removing the mean and scaling to unit variance. Centring and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the transform method.
2) **MinMax Scaling**: This technique transforms features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set, i.e. between zero and one.

Deep Learning (Convolution):

One Hot Vector: It is process of converting class labels value into the one hot vector, in which the output is in vector form.

1-hot vector of value "0": (1 0 0 0 0 0 0 0 0 0)
1-hot vector of value "1": (0 1 0 0 0 0 0 0 0 0)
1-hot vector of value "5": (0 0 0 0 0 1 0 0 0 0)

So, here we converted the class label value into the vector. Here, Shape of the label will become (42000,10). Since we are representing 0 to 9 values of the class label using vector of size 10.

Support Vector Machine:

Support vector machine is linear and non-linear classifier based on the kernel we use.

Here, we are using the principal component analysis with SVM. We have 784 attributes as pixels of image with 28 x 28. So, we have too many values and to reduce the dimensionality of the training dataset, we can use PCA (Principal Component Analysis). When attributes or features are correlated with each other on some scale we can make new dataset with smaller number of features which are very important. Here Scale of correlation of feature does not matter, they can be lightly correlated or highly correlated.

For implementing SVM with PCA, we have to use PCA package which can be imported from scikit-learn and class sklearn.decomposition.

PCA: *class* sklearn.decomposition.**PCA**(*n_components=None*, *copy=True*, *whiten=False*, *svd_solver='auto'*, *tol=0.0*, *iterated_power='auto'*, *random_state=None*)

These are the parameters of the PCA. In which "n_components" is very important which affects the score of classifier maximum.

Neural Network:

The function used for neural network is the MLP Classifier. After applying above techniques we notice that MinMax scaling method provides an output with better accuracy.

In our project we have applied neural network in two ways;

1) **With Standard Scaling**: In this method we first standardize it using the standard scaler logic and apply the neural net function provided by scikit learn.
2) **With MinMax Scaling**: In this method we first apply MinMax scaling and then apply neural net function provided by scikit learn.

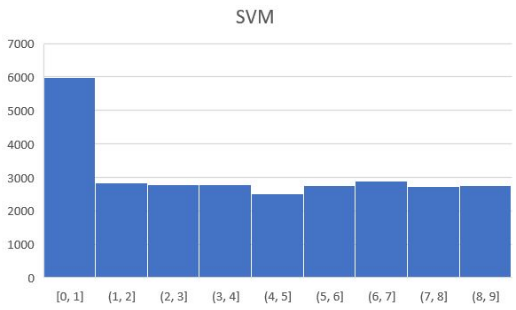
Decision Tree:

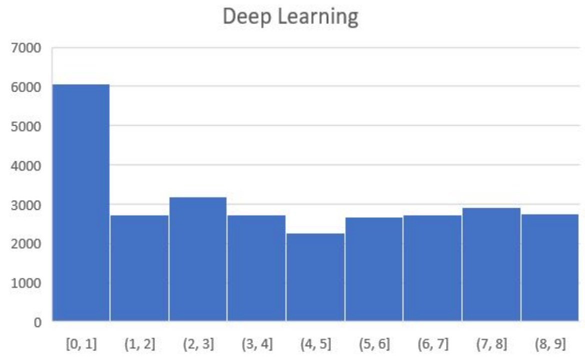
By applying Decision Tree logic to the train dataset, we will obtain a tree whose leaf nodes contain the values of the pixel digits. If we then apply the test dataset onto this tree, we will be able to predict the values of the pixels by searching across the tree. We have used the scikit-kit provided libraries to implement a decision tree classifier in our solution.

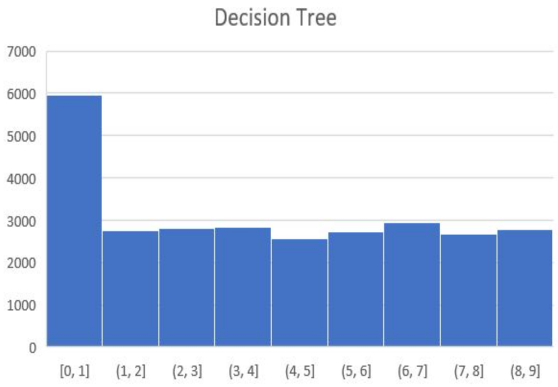
3.2 Results

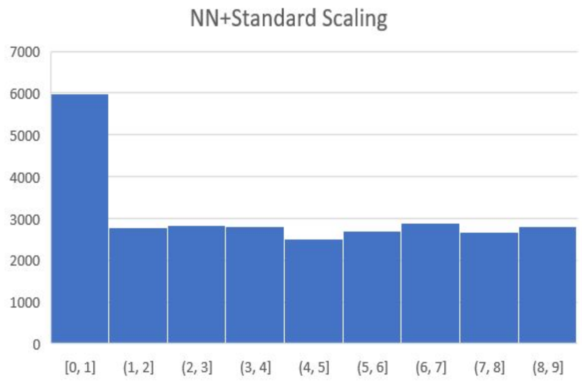
The following accuracies were noted for each classifier on Kaggle.

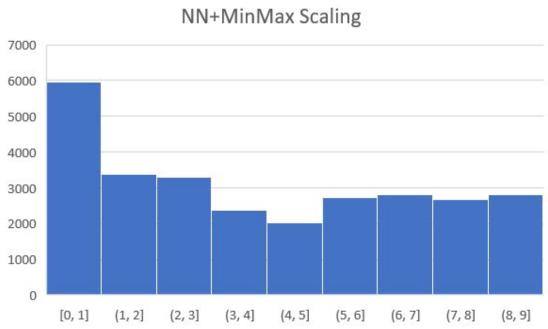
1) Decision tree: 0.85442
2) Neural net with Standard scaling: 0.85400
3) Neural Network with MinMax scaling: 0.90971
4) Deep learning:0.9455
5) SVM: 0.9820











We have also plotted histograms for all above classifiers to show the distribution of their output data.

4. Conclusion

This project introduces working of four different classifiers on the Digit Recognizer dataset and the above accuracy shows their performance. However, the use of above classifiers always depend on the dataset you possess. Starting with the training of data and predicting the digit at the end goes through different classification techniques based on the dimensional view of the data.

Although, the basic working of each classifier is same where each one tries to classify the data, the internal working of classifier differs. This working decides the best classifier for different datasets. Performance metrics like number of training examples, dimensionality of the feature space, the data should be linearly separable or not, feature dependency, overfitting, etc decide the selection of the classifier. As per the accuracy calculated, the highest performance is given by the Support vector machine for handwritten digit recognition.

- Decision tree: 85.44%
- Neural net with Standard scaling: 85.4%
- Neural Network with MinMax scaling: 90.97%
- Deep learning: 94.55%
- SVM: 98.2%

References

http://www.expertsystem.com/machine-learning-definition/
https://www.kaggle.com/c/digit-recognizer
https://en.wikipedia.org/wiki/Machine_learning
https://www.python.org/
https://en.wikipedia.org/wiki/Python_(programming_language)
http://pandas.pydata.org/
http://www.numpy.org/
https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial
http://scikit-learn.org/stable/