# Pen-Based Recognition of Handwritten Digits Data Set

Hema Bahirwani Navneet Sinha Vatsala Singh
Computer Science
Rochester Institute of Technology
{hgb1348, nxs9384, vs2080}@rit.edu

## ABSTRACT

In this paper, we built a system that efficiently recognizes digits written on a digital tablet and classifies them into their respective classes, from 0-9. We implemented and evaluated the performance of four classification algorithms namely Multinomial Logistic Regressor, Decision Tree Classifier, Random Forest and a Multi-Layer Perpectron Classifier. We started off with a Multinomial Logistic Regressor as a method of classification, and then moved on to a more complex algorithm like Decision Tree. We then increased the complexity notch a level more and saw how the previous algorithms performed versus a Random Forest and Multi-Layer Perceptron. Finally, we used Stacked Generalization algorithm to combine three of the models to make a final prediction. Based on our observations, Multi-Layer Perceptron Classifier made the most accurate predictions giving an accuracy of 98.02%. In this report, we summarize the performance analysis of four models and talk about a combined classifier using Stacked generalization.

## 1. INTRODUCTION

Handwriting Recognition is a foundation task towards bridging the gap between human style of writing and computers. Handwriting Recognition is an advanced version of optical character recognition, as handwriting has it's own characteristic. The biggest challenge in this task is that every individual has his or her own variation of writing when they write with a free hand. We needs models robust enough to tackle that hurdle to the fullest.
This project intends to build a classification model to recognize the handwritten digits (0-9). The data is obtained from UCI Machine Learning repository. It is a digital database, collected from a tablet where 44 writers used a digital pen to provide samples of the digits. There are 16 attributes for the data collected. Data preparation section provides the details on the steps taken to prepare the dataset to be used by models. We intend to use three different machine learning algorithms, compare the accuracies, and finally choose the best algorithm for the system. As there are ten classes to be classified, we kick off with a simple algorithm like multinomial logistic regression, then move on to more non-linear classifiers like Decision tree, Random Forest Classifier and Multi-Layer Perceptron. We also implemented an ensemble classification algorithm called Stacked Generalization which combines multiple classifiers together to make a final prediction. Using the results from the aforementioned algorithms, we summarized and compared their performance.
In section 2, we talk about the project objective explained from a business perspective. In section 3 we talk about where we got the data from and how we understand it, and then focus on how we could use it for our model. Section 4 talks about the ethics bound with using data for training, evaluation and testing purposes. In section 5, we make use of the decisions we took on how to structure the data, and then do our cleaning and preparation. In section 6, we discuss the models that we plan to use for our paper, and how we intend on to them. Section 7 holds the discussion about our evaluation on the previously stated models along with the results and analysis. Section 8 entails what we understood from the result, and how what we achieved can be utilized. In section 9 we give a final overview of what we achieved in the paper, and also talk about the open ends which can further be worked upon.

## 2. BUSINESS UNDERSTANDING

The problem we are trying to tackle here involves classifying handwritten digits on a digital board. We will use different models with incremental algorithmic complexities to achieve the desired result. Once the results are obtained, we perform analysis of the tested methodologies. We predict that the the complexity of our algorithms will be directly proportional to the accuracy they achieve.

## 3. DATA UNDERSTANDING

The digit dataset is created by collecting 250 samples from 44 writers. Samples by 30 writers are used for training, cross-validation and writer dependent testing, and the digits written by the other 14 are used for writer independent testing[4]. The data was collected using WACOM PL-100V pressure sensitive tablet with an integrated LCD display and a stylus. X and Y co-ordinates and pressure level values were collected every 100 milliseconds. Handwriting samples were then collected by an Intel 486 based PC which was connected to the tablets. Digits were written inside the 500*500 pixel boxes. [4] considers only x and y co-ordinates of the digits for classification. As shown in Tabel 1, the dataset contains 7494 test data points, 3498 training data points and 16 attributes, plus 1 class attribute. The dataset was thoroughly studied to find any discrepancies such as missing values, or null values or any other outliers. Training dataset was then analyzed to understand the distribution of data corresponding to each class label. The distribution of training data is shown in Table 2.

| Training | Test | Attributes | Class Attribute | Missing Values |
|---|---|---|---|---|
| 7494 | 3498 | 16 | 1 | 0 |

**Table 1: Dataset Distribution**

| Class label | Instances |
|---|---|
| 0 | 780 |
| 1 | 779 |
| 2 | 780 |
| 3 | 719 |
| 4 | 780 |
| 5 | 720 |
| 6 | 720 |
| 7 | 778 |
| 8 | 719 |
| 9 | 719 |

**Table 2: Training Data Per Class Instance Distribution**

## 4. ETHICAL ISSUES IN DATA ANALYTICS

With generation of dataset, we need to ethically obtain and use data. We need to make sure that the data that we use does not use private information without consent. If the data contains sensitive information like SSN, phone number, signature, etc., publishing that data publicly could cause severe repercussions like security breaches and can lead to strict violations of personal data.

When it comes to handwriting digits, we need to be careful that we aren't mining data from sensitive data banks which constitute of an individual's SSN, DOB, PIN and other personal data. The data should have enough variety, including mixture of multiple data banks, so that individuality of a person cannot be learnt from the dataset.

The dataset that we use here is publicly available on the UCI Machine Learning Repository. In the dataset, they collected handwritten digits from 44 writers who were aware that their digit samples will be publicly available and gave their consent to it. So, ethically, there is no challenge in this project.

## 5. DATA PREPARATION

The first step in data preparation is to normalize the data to make it scale and translation invariant. The initial range of 0-500 is changed to 0-100. The data points are then re-sampled using simple linear interpolation between two points. The re-sampled digits are represented as a sequence of T points

$$(x\_t, y\_t)_{t=1}^{T}$$

, regularly spaced in arc length, as opposed to the input sequence, which is regularly spaced in time[4]. The input vector size is 2*T where T=8 is selected for spatial re-sampling. We these plotted these points to visualize data.
For our tests, we also created a validation dataset. We split the training dataset by the ratio of 80:20 with 80% as training and 20% as validation data. The model will be trained on the training data, and then the results will be validated using the validation data. Any changes in the parameters of the model will be done based off the validation dataset so that the model never actually sees the testing data before the final test. This will help prevent bias in the model. The dataset does not contain any missing values, so no cleaning of data was required.

## 6. MODELING

In the given dataset, we have defined testing, validation and training data. Using these datasets, we run the following algorithms and record their performance.

### 6.1 Multinomial Logistic Regression

Multinomial Logistic Regression [3] was introduced for the purpose of classifying data variables with multiple classes. It sits atop logistic regression to target multi-class problems. The data sample could be either binary or ordinal or nominal.

We use Multinomial Logistic Regressor from the scikit-learn library [6]. The processed training data is fed to be learned to classify into the 10 different categories. The model is fit according to the training data, and then is used to classify the test data to obtain results.

### 6.2 Decision Tree

Decision tree [7] is a supervised learning algorithm, used to predict a class for a given data instance. It consists of nodes and branches with each non-leaf node in the tree depicting a test, that is, an if-then rule for a given attribute. Result of these if-then rules are represented by the branches of the tree. And, all the leaf-nodes represent the class labels learned from the training data.

Decision Tree classifier is implemented using python scikit-learn library[6]. The processed training data is then fed to this classifier to learn the model for the pen-based handwritten digits and testing using the testing set.

### 6.3 Random Forest

Random forests [1] are ensemble learning methods which constitute of a multitude of decision trees. For classification, each tree votes for a class and the forest chooses the one with the most votes. The same random forest can also be used for regression. By virtue of implementation, random forests should work better than decision trees as it prevents overfitting. For the final model, we used 100 decision trees to build the forest. We used brute force technique to find the number of trees which gave the best accuracy.

Random Forest classifier is implemented using python scikit-learn library[6]. The processed training data is then fed to this classifier to learn the model for the pen-based handwritten digits and tested using the provided testing dataset.

### 6.4 Multi-Layer Perceptron

Multi-Layer Perceptron(MLP) [2] is a feed forward artificial neural network which falls in the category of supervised learning algorithms.A skeleton of an MLP consists of one input layer, n number of non linear hidden layers and one output layer. Given a set of feature inputs and target outputs, it can create a non linear model to create decision boundaries which can help in classification and regression.

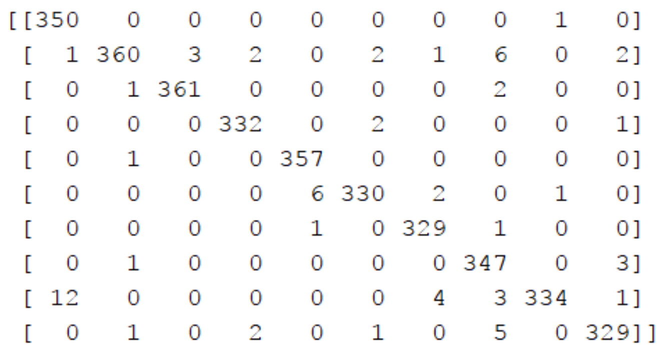
Multi-Layer Perceptron classifier is implemented using python scikit-learn library. In the architecture of the MLP, there are five hidden layers and eighty hidden units in each hidden layer. Additionally there is one input layer and one output layer. The processed training data is then fed to this classifier to learn the model for the pen-based handwritten digit and tested using the provided test set.

### 6.5 Stacked Generalization

After training three different classifiers, this work makes use of Stacked generalization[8] to combine multiple models. We split the extracted features into two disjoint sets. Random forest, Decision tree and Multinomial logistic classifier were trained on the one set and tested on the second set. Using the predictions made by the test set of different base classifiers, a higher level learner, Logistic Regressor, was trained to combine the base classifiers and provide the final prediction.

## 7. EXPERIMENTS, EVALUATION AND COMPARISONS

The four classifiers have been implemented and their performances have been recorded. Additionally we implemented Stacked Generalization as explained in [8] to combine multiple models to predict a final class like ensemble learning. Based on the results, we observe that Multi-Layer Perceptron Classifier provides the best performance for the problem in consideration. Multi-layer perceptron, a neural network approach, is a very popular approach, and known to train well and provide very significant results for 2D inputs even with less amount of data. For this particular problem, multi-layer perceptron gives the best performance with an accuracy rate of about 98.02% which is the highest amongst all the other classifiers we evaluated. The confusion matrix for the same is shown in 1. As we can see, most of the data points are predicted correctly.

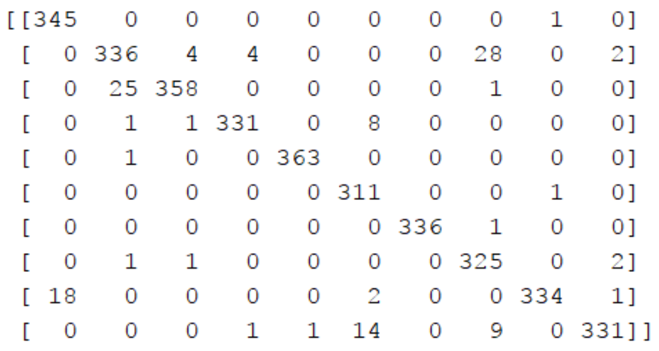


```
[[350   0   0   0   0   0   0   0   1   0]
 [  1 360   3   2   0   2   1   6   0   2]
 [  0   1 361   0   0   0   0   2   0   0]
 [  0   0   0 332   0   2   0   0   0   1]
 [  0   1   0   0 357   0   0   0   0   0]
 [  0   0   0   0   6 330   2   0   1   0]
 [  0   0   0   0   1   0 329   1   0   0]
 [  0   1   0   0   0   0   0 347   0   3]
 [ 12   0   0   0   0   0   4   3 334   1]
 [  0   1   0   2   0   1   0   5   0 329]]
```

**Figure 1: Confusion matrix for Multi-Layer Perceptron**

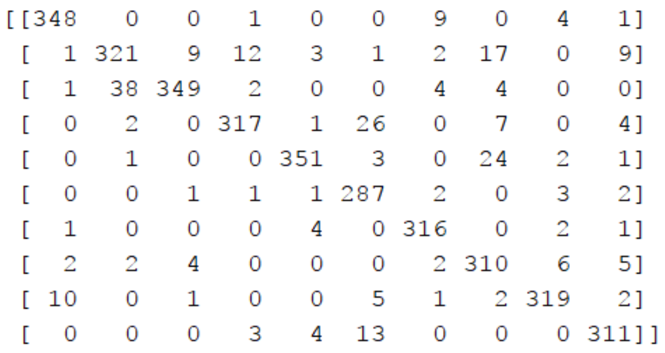
Random Forest categorizes the digits into their respective classes with 96.34% accuracy rate. The confusion matrix generated by the Random Forest Classifier is shown in 2. As we can see in the confusion matrix, all the individual classifiers, for each digit, have similar accuracy which tells us that the classifier is not biased toward any class particular class.

Decision Trees, on the other hand, classifies them with 91.4% accuracy rate. The confusion matrix using this classifier, is shown in 3 Multinomial Logistic Regression, which basically generalizes Logistic Regression for for more than two classes, gives an accuracy of 89.05%. The confusion matrix for the same is shown in 4.



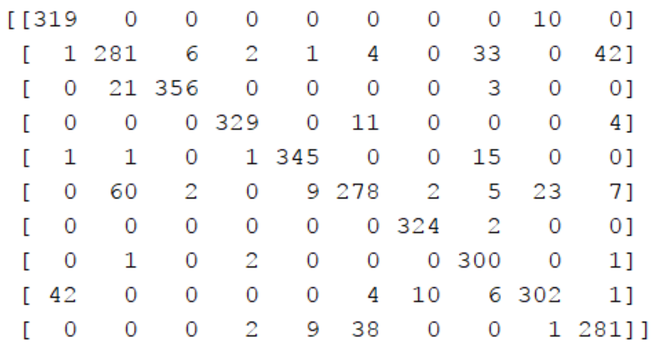
```
[[345   0   0   0   0   0   0   0   1   0]
 [  0 336   4   4   0   0   0  28   0   2]
 [  0  25 358   0   0   0   0   1   0   0]
 [  0   1   1 331   0   8   0   0   0   0]
 [  0   1   0   0 363   0   0   0   0   0]
 [  0   0   0   0   0 311   0   0   1   0]
 [  0   0   0   0   0   0 336   1   0   0]
 [  0   1   1   0   0   0   0 325   0   2]
 [ 18   0   0   0   0   2   0   0 334   1]
 [  0   0   0   1   1  14   0   9   0 331]]
```

**Figure 2: Confusion matrix for Random Forest**



```
[[348   0   0   1   0   0   9   0   4   1]
 [  1 321   9  12   3   1   2  17   0   9]
 [  1  38 349   2   0   0   4   4   0   0]
 [  0   2   0 317   1  26   0   7   0   4]
 [  0   1   0   0 351   3   0  24   2   1]
 [  0   0   1   1   1 287   2   0   3   2]
 [  1   0   0   0   4   0 316   0   2   1]
 [  2   2   4   0   0   0   2 310   6   5]
 [ 10   0   1   0   0   5   1   2 319   2]
 [  0   0   0   3   4  13   0   0   0 311]]
```

**Figure 3: Confusion matrix for Decision Tree**



```
[[319   0   0   0   0   0   0   0  10   0]
 [  1 281   6   2   1   4   0  33   0  42]
 [  0  21 356   0   0   0   0   3   0   0]
 [  0   0   0 329   0  11   0   0   0   4]
 [  1   1   0   1 345   0   0  15   0   0]
 [  0  60   2   0   9 278   2   5  23   7]
 [  0   0   0   0   0   0 324   2   0   0]
 [  0   1   0   2   0   0   0 300   0   1]
 [ 42   0   0   0   0   4  10   6 302   1]
 [  0   0   0   2   9  38   0   0   1 281]]
```

**Figure 4: Confusion matrix for Multinomial Regression**

3 shows a table which compares the losses produced by all the implemented models. As it is shown, MLP gives the lowest loss and performs the best amongst all other classifiers.

Plot shown in 5 compares the errors of the individual ten classifiers (0-9) for all the four models implemented. As we can see in the plot, the errors for almost all the classes for Multi-Layer Perceptron classifier are the least.

Stacked generalization combined the three of the above classifiers and trained a new simple Multinomial Logistic Regressor to make the final. While we hoped that it would increase the accuracy, there was actually a decrease in the accuracy after combining the models to 69%. After doing a lot of literature review on the same, we found out that

|  |  |
|---|---|
| Multi-Layer Perceptron | **0.02** |
| Random Forest | 0.03 |
| Decision Tree | 0.08 |
| Multinomial Logistic Regression | 0.11 |

**Table 3: Loss comparison for different models**

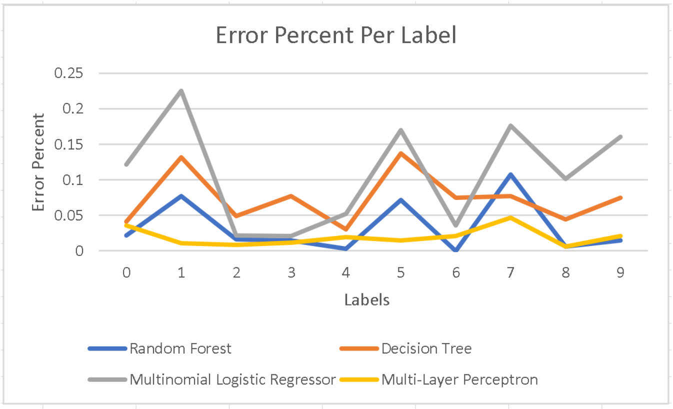


**Figure 5: Plot of error percent for individual classes for all the classifiers**

Stacked Generalization works well when the models are not correlated and provide significantly different results. In our case, all the models are providing significant results and the confusion matrix of all the classifiers shown above are not too different which means that models are highly correlated and can be one of the reasons for the behavior of the explained algorithm.

## 8. DEPLOYMENT

After processing the data and making it suitable for the classification models, we fed them to the implemented models and recorded their performance. The classification models, in consideration, have been implemented using [6] python scikit-learn library. In this work we studied and implemented different model but for the final algorithm, we chose Multi-Layer Perceptron as the classifier to predict the handwritten digits as it provides the best results. The other models and Stacked Generalization can be discarded for deployment as Multi-Layer Perceptron outperforms all the other models for our problem.

## 9. CONCLUSION AND FUTURE WORK

This report focuses on a problem of predicting handwritten digits(0-9). We studied and implemented different machine learning classifiers like Multi-Layer Perceptron, Random Forest, Decision Tree and Multinomial Logistic Regression . We compared the performances of each of the classifiers in which Multi-Layer Perceptron Classifier performs the best with an accuracy of 98.02% and Multinomial Logistic Regression with the least accuracy of 89.05%. We also implemented an algorithm called Stacked Generalization which combines multiple classifiers and trains another classifier on the prediction of it's base classifiers to finally make a final prediction. Although, Stacked Generalization didn't provide expected significant results, it was good learning of a new ensemble classification algorithm other than Random Forest.

Because the dataset is very small, we could not test the models extensively. In future, to improve on the results even further, more data can be collected or generated synthetically to train the models. The MLP architecture we have used consists of 5 hidden layers, to test the model further, a different architecture can be used and evaluated. There are many famous Convolutional Neural Network architectures like VGG-16 [5], which are proven to perform well for 2-D inputs like images as in our problem. Once we have the best performance results, the model can be frozen (memory limits in a mobile phone) and put in a mobile application where users can use a pen to write on the phone, and the model would be able to predict the handwritten digits.

## 10. REFERENCES

[1] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[2] A. Khotanzad and C. Chung. Application of multi-layer perceptron neural networks to vision problems. *Neural Computing & Applications*, 7(3):249–259, Sep 1998.

[3] C. Kwak and A. Clayton-Matthews. Multinomial logistic regression. *Nursing research*, 51 6:404–10, 2002.

[4] M. Lichman. UCI machine learning repository, 2013.

[5] S. Liu and W. Deng. Very deep convolutional neural network based image classification using small training sample size. pages 730–734, Nov 2015.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[7] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, Mar. 1986.

[8] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

## APPENDIX

## A. INDIVIDUAL CONTRIBUTION

1. Hema Bahirwani: Studied the data followed by data cleaning and preprocessing. Studied,implemented and evaluated Decision Tree classifier. Illustrated and demonstrated the classifier in the report.
2. Navneet Sinha: Studied,implemented and evaluated Multinomial Logistic Regressor and Multi-layer perceptron classifier. Illustrated and demonstrated the classifiers in the report.
3. Vatsala Singh: Studied,implemented and evaluated Random Forest Classifier and Stacked Generalization. Illustrated and demonstrated the classifiers in the report.

## B. USER INSTRUCTIONS

In the submitted folder, there are four files:

1. training.csv: This is the cleaned and preprocessed training dataset file.
2. test.csv: This is the cleaned and preprocessed testing dataset file.

3. multipleClassifiers.py: This is a python script which implements multiple classifiers for the the prediction of handwritten digits. Functionality of several methods implemented in the script is provided in the documentation of the code.
4. stacking.py: This is a python script which implements Stacked Generalization. The functionality of all the methods implemented in the script is provided in the documentation of the code.

To run the multiple models, on a terminal, run: "python multipleClassifiers.py" To run the Stacked Generalization algorithm, on a terminal, run: "python stacking.py"