**Digit Recognizer**

**Project Report by Sruti Paku(sxp131731), Maringanty Krishna Kavya(kxm152630)**

**Introduction and Problem Description:**

The project Digit Recognizer aims is to identify hand written digits. We used Kaggle training data as the input. It contains the pixel values of each hand written image. Our aim is to predict the number on the given hand written number image (pixel values).

**Related Work:**

We pre-processed the input data set to check for NA values. Displayed the input values and the associated labels the images needed rotation. We implemented three classifiers Deep learning, DBN deep net and Random forest.

**Dataset Description:**

The data files train.csv and test.csv contain gray-scale images of hand-drawn digits, from zero through nine.

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 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 255, inclusive.

The training data set, (train.csv), has 785 columns. The first column, called "label", is the digit that was drawn by the user. The rest of the columns contain the pixel-values of the associated image. Each pixel column in the training set has a name like pixelx, where x is an integer between 0 and 783, inclusive [5].

**Pre Processing Technique:**

In the Dataset Each image is represented from 784 pixels in each row. A total of 8400 images are given in the training data. The following preprocessing is performed on the Dataset.

* The NA values are checked- There are no NA values in the dataset.
* The Scaling and Normalization is not performed on the dataset as each column represents a pixel value and change of value might reduce the authenticity of data.
* As the data contains pixel values we first plot the pixel color values to obtain a picture of the digit. The images need rotation so we rotate and plot the images again.

**Proposed Solution and Methods:**

We have taken the approach of Deep learning and Random Forest classifiers. For Deep Learning we have used two packages i.e. H2o and Deep net. In H2o we used the auto encoder method for deep learning and in Deep net package we used deep belief networks.

For Random forest classifier we used the Random Forest package in R.

**Deep Learning**: Deep Learning is a part of broader family of Machine Learning based on Learning Representations. An observation can be represented in many eats such as vector of Intensity values, set of edges, regions of particular shape etc.

There are various deep learning architectures such as Deep Neural Networks, Convolutional Deep Neural Networks, Deep Belief Networks and Recurrent Neural Networks. The Convolutional Deep Neural Networks play a revolutionary role in Image Recognition. In our experiment we have used 2 Layered Neural Network with Rectifier with Dropout as the Activation.

**Random Forest**:

Random forest allows us to use many trees for classification. In random forest we place input vector under each trees in the forest, each tree will give a classification based on the number of overall votes each tree has in the forest. There is no pruning allowed in the random forest. The forest error rate depends on the correlation and strength values.

The correlation is between two trees in the forest, increasing the correlation might increase the error rate. The tree should have lower error rate in order to be classified as strong classifier. It is inversely proportional with the forest error rate.

**Experimental Results and Analysis:**

After multiple iterations varying the parameters of the model We have achieved the best Accuracy of 96.46% in Deep Learning.

**H2o Deep Learning:**

|  |  |  |  |
| --- | --- | --- | --- |
| Training/Test ratio | Hidden Layers | epochs | Accuracy |
| 80/20 | 50 | 15 | 92.52 |
| 80/20 | 100 | 15 | 94.36 |
| 80/20 | 150 | 20 | 95.9 |
| 80/20 | 200 | 20 | 96.14 |
| 80/20 | 250 | 20 | 96.46 |

**Deep net - dbn:**

|  |  |  |  |
| --- | --- | --- | --- |
| Training/Test ratio | Hidden Layers(3) | numepochs | Accuracy |
| 80/20 | 50 | 3 | 96.25 |
| 80/20 | 100 | 3 | 98.16 |
| 80/20 | 150 | 3 | 98.173 |
| 80/20 | 200 | 8 | 98.174 |

**Random Forest:**

|  |  |  |
| --- | --- | --- |
| Training/Test ratio | Number of trees | Accuracy |
| 80/20 | 20 | 94.25 |
| 80/20 | 25 | 95.5 |
| 80/20 | 50 | 95.83 |
| 80/20 | 100 | 95.98 |

**Conclusion:**

The Deep Learning and Random Forest have given a good accuracy for multiple iterations and change in parameters. Deep Learning accuracy was highest with 250 hidden nodes, the accuracy of deep net was highest with 200 hidden nodes with 98.174 % accuracy. The best accuracy was given by deep net dbn classifier 98.174%.

**Contribution of Team Members:**

Pre- Processing- Krishna Kavya Maringanty, Sruti Paku

Deep Learning- Krishna Kavya Maringanty

Random Forest- Sruti Paku.

Deep Net – Sruti Paku

**References:**

[1] Deep Learning

<https://en.wikipedia.org/wiki/Deep_learning>

[2] Convolutional Neural Networks

<https://en.wikipedia.org/wiki/Convolutional_neural_network>

[3] Deep Learning with H2O Demonstration

<http://h2o-release.s3.amazonaws.com/h2o/rel-tibshirani/3/docs-website/h2o-r/h2o_package.pdf>

[4] Deep Learning in R using H2O

<http://didericksen.github.io/deeplearning-r-h2o/>

[5] Data set

<https://www.kaggle.com/c/digit-recognizer/data>

[6] Random Forest

https://en.wikipedia.org/wiki/Random\_forest