Project 3: Handwritten Digit Image Classification

Rohan Gupta
Department of Computer Science
University at Buffalo, SUNY
rgupta24@buffalo.edu
50290793

1 Overview

In this project we will apply machine learning to solve the classification task of identifying a 28x28 grayscale handwritten digit. We will implement four classification algorithms for this task: **logistic regression**, **neural networks**, **support vector machine** and **random forest**. We will also implement an ensemble of these four classifiers with an aim to get an improvement over results obtained with individual classifiers.

2 Datasets and Data Preprocessing

We have the two following datasets for this task:

2.1 MNIST Dataset

The MNIST dataset is a large dataset which consists 28x28 pixel images of handwritten digits from 0 to 9. It is one of the most popular datasets. It comprises of 60,000 training images and 10,000 testing images. For preprocessing, we will first flatten this image, which will give a 1-D vector of 784 pixel values. We will use these 784 pixel values as features for our classifiers. We will also partition this dataset as 50,000 training data points, 10,000 validation data points and 10,000 testing data points. The data and labels are provided as tuples. Finally, we will slice these tuples to separate data points and labels. We will both train and test our model with this dataset.

2.2 USPS Dataset

The USPS dataset also consists of images of handwritten digits. The dataset comprises of 20,000 images in total (2,000 for each digit). We will first resize these images to 28x28 pixels to make them similar to the MNIST dataset. We will use this dataset to test the model trained with MNIST dataset. This will test how our model generalizes to a new population of data.

3 Solutions (Model Training Approaches)

3.1 Multi-class Logistic Regression / Softmax Regression

Multi-class Logistic regression or Softmax regression is the appropriate regression analysis to conduct when the dependent variable y is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

The **Genesis Equation for Logistic Regression** has the following form:

$$\hat{y} = \sigma(w^T x) = \sigma(w_1 x_1 + w_2 x_2 + ... + w_n x_n + b) \tag{1}$$

where $w = (w_0, w_1, w_2 ... w_{n-1})$ is the weight vector which will be computed using the training samples, $x = (x_0, x_1, x_2 ... x_{n-1})^T$ is the feature vector and b is the bias term.

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}} \tag{2}$$

Here σ is the softmax function. The softmax function returns a same dimension vector with all positive values, where each valus is in the range (0,1), and they add up to 1.

The **Loss/Error function for Logistic Regression is Cross Entropy** which has the following form:

$$L = -(y * log(\hat{y}) + (1 - y) * log(1 - \hat{y})) \tag{3}$$

3.2 Neural Network

Neural Network is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, processes information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve a specific problem. Neural Networks, like people, learn by examples. In a neural network, we have a series of one or more layers with their associated weights.

The **Genesis Equation for Neural Network** has the following form:

$$\hat{y} = f(w, x) \tag{4}$$

The **Loss/Error function we used for Neural Network is Softmax Cross Entropy with Logits**.

3.3 Support Vector Machine

Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. It finds out a line or a hyper-plane in multidimensional space that separate outs the classes. It is very easy to use and solve variety of problems with little tuning. It is a supervised machine learning algorithm which can be used for both classification or regression challenges. SVM generally takes a long time to converge.

The **Genesis Equation for Support Vector Machine** has the following form:

$$h_\theta x = 1 \qquad \theta_T X >= 1 \tag{5}$$

$$h_\theta x = 0 \qquad \theta^T X <= -1 \tag{6}$$

The **The Cost Function for Support Vector Machine** has the following form:

$$cost = C[\sum_{i=1}^{m} cost_1 + cost_0 + \sum_{j=1}^{n} Q_j^2] \tag{7}$$

3.4 Random Forest

Random Forest Classifier is an ensemble algorithm which computes the results based on the results of multiple decision trees. A Decision trees represents the process of decision-making, It creates a flow chart like structure, where each node signifies a condition. The branches represent the condition's outcome and the leaf nodes represent the class labels.

In the Random Forest ensemble algorithm, we divide the dataset into N non-overlapping parts and create N different decision trees. To predict a target variable for classification, the mode is calculated of the results of all individual decision trees.

The algorithm used to build a decision tree is **Iterative Dichotomiser 3 (ID3). We use the Entropy and the Information Gain to make decisions.** The equations are as follows:

Entropy:

$$S = -\Sigma p_i \log_2 p_i \tag{8}$$

Information Gain:

$$IG = [\, E_0 - \Sigma \frac{N_i}{N}] S_i \tag{9}$$

4 Results

4.1 Results for Logistic Regression:

				MNIST Data			**USPS Data**
				Training	Validation	Testing	Testing
	Epoch	η	Mini-Batch Size	Accuracy	Accuracy	Accuracy	Accuracy
1.	1500	0.01	-	86.65	88.13	88.05	34.15
2.	1000	0.05	-	89.02	90.28	90.10	35.69
3.	1500	0.05	-	89.68	90.79	90.59	36.17
4.	1500	0.05	256	93.54	92.9	92.56	32.57

With a learning rate (η) of 0.01 and number of epochs as 1500, we find that the training accuracy comes out as 86.65%. The validation accuracy is 88.13%, which is close to our training accuracy, so we know that the training of the model is going in the right direction. But, the training accuracy increases by a very small percentage in the last 500 epochs.

Hence, we increase the learning rate to 0.05 for the next training cycle. With η = 0.05 and at 1000 epochs, we get a better training accuracy of 89.02% and a validation accuracy of 90.10%. We train the model for more 500 epochs as both the training and validation accuracies are still on a rise.

With η = 0.05 and at 1500 epoch, we get a training accuracy of 89.68% and a validation accuracy of 90.79%, slightly better than the previous values. We will halt our training and finally, we get a testing accuracy on the MNIST data as 90.59% and a testing accuracy on the USPS data as 36.17%.

By implementing **mini-batch gradient descent**, and taking the mini-batch size of 256, we get a better testing accuracy with MNIST of 92.56%.

For MNIST Data:

```
Best Testing Accuracy: 92.56%

Confusion Matrix:
[[ 960    0    0    4    1    3    7    3    2    0]
 [    0 1115    3    3    0    1    3    1    9    0]
 [    5   12  917   19    7    3   12    9   46    2]
 [    4    0   17  929    1   20    3   11   19    6]
 [    1    1    6    4  919    0    7    5    8   31]
 [   10    2    5   39    7  768   11    9   35    6]
 [    8    3    6    3    7   17  911    2    1    0]
 [    2    7   20    7    6    1    0  952    4   29]
 [    6    9    6   20    8   24   11   10  869   11]
 [    8    7    1   10   24    7    0   24   12  916]]

Average Precision Score (micro-averaged over all classes): 0.86417536
```

For USPS Data:

```
Best Testing Accuracy: 36.17%

Confusion Matrix:
[[ 535    1   91  134  160  228   55  213  113  470]
 [ 101  396   26  187  393   71   36  460  274   56]
 [ 131   41 1114  184   56  205   92   39  101   36]
 [  43    8  302  863   15  601    6   77   65   20]
 [  46   10   51   53  785   80   26  524  246  179]
 [ 115   11  111  234   70 1145   70   92  119   33]
 [ 132    8  589   98   86  386  645   18   24   14]
 [ 122   40   68  584   69  147    9  611  280   70]
 [ 210   11  150  433  100  618   74  125  233   46]
 [  22   14   68  482  110   55    6  720  335  188]]

Average Precision Score (micro-averaged over all classes): 0.17354704577070948
```

4.2 Results for Neural Network :

					MNIST Data			**USPS Data**
	Epoch	η	Mini-Batch Size	Number of Neurons	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing Accuracy
1.	400	0.01	256	256	98.5	97.53	97.43	47.73
2.	400	0.025	256	512	98.84	98.01	97.93	49.98

```
For MNIST:
Best Testing Accuracy: 97.93%

Confusion Matrix:
[[ 968    0    1    1    1    1    3    1    2    2]
 [    0 1122    2    2    0    2    2    1    4    0]
 [    4    2 1009    2    3    0    2    7    3    0]
 [    0    0    4  990    0    3    0    3    4    6]
 [    3    0    2    0  962    0    3    1    0   11]
 [    3    1    0    6    1  867    7    1    3    3]
 [    5    3    1    1    2    3  941    0    2    0]
 [    0    4    8    2    0    0    0 1008    1    5]
 [    3    0    2    6    3    4    2    3  947    4]
 [    3    4    0    5    8    1    1    6    2  979]]

Average Precision Score (micro-averaged over all classes): 0.9610984899999999
```

```
For USPS:
Best Testing Accuracy: 49.98%

Confusion Matrix:
[[ 541    1   99   51  204   66   40   68  147  783]
 [  74  716  121   92  219   48   49  587   38   56]
 [  84   24 1501  118   32   70   51   58   48   13]
 [  29   10  114 1476   12  261    3   29   48   18]
 [   8   27   41   14 1126   41    9  366  287   81]
 [  51    1   49  121   19 1420   28   61  210   40]
 [ 132   28  271   55   83  132 1208   13   58   20]
 [  42  100  197  428   34   49    6  941  178   25]
 [  90    8  169  303  115  338   73   99  765   40]
 [   8   51   80  277  142   33    2  655  450  302]]

Average Precision Score (micro-averaged over all classes): 0.299842522752669
```

4.3 Results for Support Vector Machine :

			MNIST Data		USPS Data
			Validation	Testing	Testing
	kernel	gamma	Accuracy	Accuracy	Accuracy
1.	linear	auto	94.23	93.89	32.72
2.	rbf	1	18.24	17.59	10.00
3.	rbf	auto	94.48	94.79	40.28

```
For MNIST:
MNIST Testing Accuracy: 94.35
Confusion Matrix:
[[ 967    0    1    0    0    5    4    1    2    0]
 [    0 1120    2    3    0    1    3    1    5    0]
 [    9    1  962    7   10    1   13   11   16    2]
 [    1    1   14  950    1   17    1   10   11    4]
 [    1    1    7    0  937    0    7    2    2   25]
 [    7    4    5   33    7  808   11    2   10    5]
 [   10    3    4    1    5   10  924    0    1    0]
 [    2   13   22    5    7    1    0  954    4   20]
 [    4    6    6   14    8   24   10    8  891    3]
 [   10    6    0   12   33    5    1   14    6  922]]

Average Precision Score (micro-averaged over all classes): 0.8958422500000001

For USPS:
USPS Testing Accuracy: 40.28
Confusion Matrix:
[[ 594    3  358   18  287  218   69   33    6  414]
 [   59  601   83  131  311  213   55  510   20   17]
 [ 138   31 1344   61   55  194   67   73   22   14]
 [   81    4  151 1131   16  494    5   73   27   18]
 [   15   76   72   13 1212  233   17  188   68  106]
 [   84   21  147  109   25 1471   61   50   22   10]
 [ 188    9  431   22  121  411  795    5    8   10]
 [   49  239  447  248   59  420   15  457   43   23]
 [   75   28  190  184   94 1012   97   41  249   30]
 [   28  191  206  233  236  163   12  502  226  203]]

Average Precision Score (micro-averaged over all classes): 0.22201733817877456
```

4.4 Results for Random Forest :

		MNIST Data		USPS Data
		Validation	Testing	Testing
	estimators	Accuracy	Accuracy	Accuracy
1.	10	95.13	94.59	32.19
2.	100	97.26	96.86	40.88
3.	1000	97.39	97.06	42.23

```
For MNIST:
MNIST Testing Accuracy: 97.06

Confusion Matrix:
[[ 970    0    0    0    0    1    3    1    4    1]
 [    0 1121    3    3    0    2    3    0    2    1]
 [    6    0  998    7    3    0    4    8    6    0]
 [    0    0    7  978    0    4    0    9    7    5]
 [    1    0    1    0  954    0    5    0    2   19]
```

```
 [   2    0    0    8    3  863    6    3    5    2]
 [   6    3    0    0    2    4  940    0    3    0]
 [   1    3   19    1    1    0    0  989    2   12]
 [   4    0    5    8    3    5    4    3  930   12]
 [   6    5    3    9    9    4    1    4    5  963]]

Average Precision Score (micro-averaged over all classes): 0.94500436

For USPS:
USPS Testing Accuracy: 42.16

Confusion Matrix:
[[ 611   16  259   67  448  142   59  129    1  268]
 [   7  722   28   98   15  126   35  967    1    1]
 [  68   39 1260   69   54  182   10  312    3    2]
 [  33   10   65 1327   51  295    1  200    2   16]
 [  10  227   33   21 1099  159   13  393   25   20]
 [  74   37   70   64   17 1591    9  131    3    4]
 [ 295   56  201   23   91  347  850  123    3   11]
 [  32  340  377  231   29  247   32  702    3    7]
 [  42   45  139  199   97 1135   65  103  165   10]
 [  15  312  196  267  235  125   12  655   77  106]]

Average Precision Score (micro-averaged over all classes): 0.23563939435033654
```

5 Answers to the Questions asked

5.1

Highest accuracy with MNIST test set: 97.93%
Highest accuracy with USPS test set: 49.98%

We get good results with the MNIST test set, but not so good results with the USPS test set. We observe a big difference between these two accuracies.

As our model was trained only with the MNIST training data, getting a low accuracy with USPS data could signify that our model doesn't generalize well to a new population of data.

The "No Free Lunch" theorem simply states that there is no one model that works best for every problem. In other words, a model which works well for some particular data might not necessarily work well for some other dataset. A model is a simplified representation of reality, which disregards the unnecessary details and focusses on the important aspects of the data that we train it with. This implies that a model which describes a certain situation well, might fail in another situation.

We observe that our results support "No Free Lunch" theorem.

5.2

From the **Confusion matrix and the Average Precision Score (AP)** provided in the results section, we can compare the classifiers based on their Precision and Recall.

We observe that Neural Networks gives the best overall results.

5.3

Accuracy obtained with Majority Vote Ensemble (Hard Vote) are:
For MNIST Data: 96.34%
For USPS Data: 44.18%

From these results we infer that our ensemble performs better than Logistic Regression and SVM, but not better than Neural Network and Random Forest.

References

[1] https://www.statisticssolutions.com/what-is-logistic-regression/

[2] https://www.doc.ic.ac.uk/ nd/surprise_96/journal/vol4/cs11/report.html

[3] https://medium.com/@rakendd/building-decision-trees-and-its-math-711862eea1c0

[4] https://chemicalstatistician.wordpress.com/2014/01/24/machine-learning-lesson-of-the-day-the-no-free-lunch-theorem/