Project 3 Implementation

Amit Menon
Department of Computer Science
State University of New York at Buffalo, NY
amitanil@buffalo.edu

Abstract

The aim of the project is to implement 4 machine learning models on a multiclass classification problem such as identifying digits MNIST dataset of images. We train and test the models individually on MNIST dataset and then test again on USPS dataset and gather inferences on performance of each of these models. Finally, we implement a combined model of the 4 trained ones and analyze its performance.

1 Data Preprocessing

1.1 MNIST

The data is already extracted in a pickled file. Data is extracted in training, testing and validation sets. The input and output values are separated, and the output labels are one hot encoded into encoded_label_train, encoded_label_test and encoded_label_val set. Both the input and output values are shuffled in unison for each set so that we can have random sampling of the data.

1.2 USPS

From the given code the input and output matrices have been extracted. Since they are ordered class wise, I shuffle the whole data set and then encode the output.

2 Logistic Regression

I implement a stochastic gradient descent optimizer to optimize this logistic softmax regression. I divide the whole data into batches and run the gradient descent for each batch

2.1 Tuning of parameters

Initial settings: Alpha:0.2, Iterations per batch=600,BATCH_SZE=200,LAMDA=0.2

This setting gave an initial accuracy of 84% on the test set. A plot of the error function over the iterations shows that even though the error is reducing, a print of cross entropy error on the console showed that the values reached the global minima and then increased again, indicative of a scope of improvement. And hence I began modifying the hyper parameters.

1. Batch: I realized having kept a batch size less than the number of features would affect the performance, so I increased it to 1000
2. Lamda: Even after increasing the batch size the performance improved marginally. The uneven increase in cross_entropy_error tells that data was under-fitting, hence I reduced it to e^-7 for which the accuracy for mnist improved but error for mnist test and both usps increased, which meant the model wasn't generalizing well. Finally I increased it to e^-5
3. Alpha: Increasing alpha from 0.02 to 0.11, accuracy and error improved gradually until it maxed out, indicating overfitting. We obtained a peak accuracy at alpha = 0.8

4. Iters: as seen below in Fig 1.1 a) the model converges in around 3000 iterations in total. Hence I reduced the batch iterations to 20 and started increasing gradually until it peaked out at 150 iterations for a batch as shown in Figure 1.1 b)

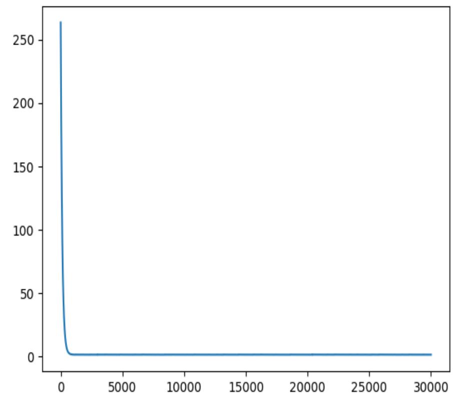

Figure 2.1 a)

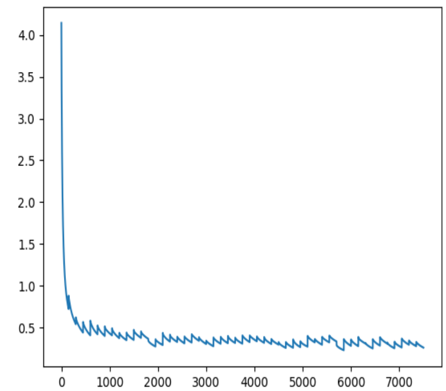

Figure 3.1 b)

Final settings: Alpha:0.08, batch=1000, Iters=150, Lamda=e^-5 giving below results

```
----------------------------------------------Logistic Regression Results-------------------------------
Cross Entropy Loss Train : 0.32892455405163695
Cross Entropy Loss Test : 0.3153285214668487
Cross Entropy Loss Validation : 0.29845130026961314
Cross Entropy Loss USPS : 3.6561779398264527
Accuracy Train : 90.904
Accuracy Test : 91.27
Accuracy Val : 91.38
USPS Accuracy 33.646682334116704
```

2.2 Confusion Matrix

As seen for the MNIST Data, the classification model performs really well, because all the large numbers reside in the diagonal. Besides most classes have slightly better recall,than precision and both these numbers are high, i.e fscores are high. Since precision for classes 2 3 5 8 9 are lower, it is indicative that it is confused for these classes the most.

```
Confusion MAtrix for Losgistic Softmax Regression MNIST :
[[ 962    0    1    1    1    8    4    1    2    0]
 [   0 1110    4    1    0    4    4    1   11    0]
 [   9    4  937   16   15    2    5   12   27    5]
 [   5    2   34  896    0   31    1   10   20   11]
 [   2    4    8    2  902    0    9    4    5   46]
 [  12    4   12   39    7  760    9    8   33    8]
 [  22    4   14    0   18   19  875    2    4    0]
 [   4    9   32    7    6    1    0  923    2   44]
 [  10    9   16   30    8   34    6   14  842    5]
 [  13    6    1   10   49    8    0   21   11  890]]
Classification Report Mnist
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       980
           1       0.96      0.98      0.97      1135
           2       0.88      0.91      0.90      1032
           3       0.89      0.89      0.89      1010
           4       0.90      0.92      0.91       982
           5       0.88      0.85      0.86       892
           6       0.96      0.91      0.94       958
           7       0.93      0.90      0.91      1028
           8       0.88      0.86      0.87       974
           9       0.88      0.88      0.88      1009

   micro avg       0.91      0.91      0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000
```

```
Confusion MAtrix for Losgistic Softmax Regression USPS :
Classification Report USPS
              precision    recall  f1-score   support

           0       0.27      0.25      0.26      2000
           1       0.43      0.17      0.25      2000
           2       0.38      0.63      0.48      1999
           3       0.32      0.49      0.39      2000
           4       0.55      0.42      0.47      2000
           5       0.29      0.61      0.40      2000
           6       0.55      0.27      0.36      2000
           7       0.20      0.16      0.18      2000
           8       0.18      0.14      0.16      2000
           9       0.22      0.12      0.16      2000

   micro avg       0.33      0.33      0.33     19999
   macro avg       0.34      0.33      0.31     19999
weighted avg       0.34      0.33      0.31     19999

[[ 502    3  214   70   75  389   88   84  283  292]
 [ 178  342  252  242  266  117   16  429  103   55]
 [ 190   20 1259  112   33  207   49   51   49   29]
 [ 127    9  249  979   11  409   16   95   57   48]
 [  81   71   34   88  840  193   57  195  236  205]
 [ 150   12  249  148   23 1215   65   48   63   27]
 [ 282   13  496   62   60  432  536   30   39   50]
 [ 112  167  247  607   40  226   14  322  160  105]
 [ 188   19  163  227   68  810  116   43  278   88]
 [  43  134  115  527  123  153   18  351  288  248]]
```

For USPS data, the performance is a downgrade, as seen from the f1 scores . It performs the best to identify classes 2 4 5 6. But does a poor job of identifying 0 8 and 9. The reason for this maybe that while resizing the usps images, some information (features) may be different for these classes.

2 Neural Networks

1.2 Tuning of hyper parameters

I am using a stochastic gradient descent to correct our model and in that regards learning rate is a factor that determines how quickly or slowly will our gradient descent converge to the minimum. I use the Multiplayer perceptron classifier from the scikit package to implement a neural network. I have kept early_stopping parameter as true, which prevents unnecessary iterations.

Initial settings: Alpha=e^-5, learning_rate=0.02, activation='logistic', hidden_layer_size=15

Learning_rate: Increase in alpha showed that model was converging quickly, till lamda = 0.08 after that the training accuracy improved but testing error for both MNIST and USPS increases

Hidden layer units Number of nodes in the hidden layer is a very impactful factor in improving the model. As each node of a layer is an input to node in the next layer. Performance improved as the factor was increased from 15, 20, 50 ,80 as increase in this parameter computes as many new features in the first step. Further increase caused the error in USPS data set to increase

Activation: 'Relu' performs better on the training set as it eliminates the dying perceptron problem, but logistic generalized better for the USPS data.

```
----------------------------------------Neural Network Results------------------------------------------------
Cross Entropy Loss Train : 0.14918352110799865
Accuracy Train : 98.008
Cross Entropy Loss Test (MNIST) : 0.5002145134429888
Accuracy TEST (MNIST) : 95.23
Cross Entropy Loss Validation (MNIST) : 0.494386364339786
Accuracy Validation (MNIST) : 95.59
Error TEST USPS : 11.63270854049747
Accuracy TEST USPS : 35.24176208810441
```

1.2 Confusion Matrix

This is an indication that the learning rate is too slow. If the learning rate is too small, gradient descent will not converge in given number of iterations.

```
Confusion MAtrix for Neural Networks MNIST test :
[[ 973    0    0    1    0    1    4    1    0    0]
 [  13 1115    3    1    0    1    2    0    0    0]
 [  39    1  980    3    1    0    0    6    2    0]
 [  50    0    6  947    0    2    0    5    0    0]
 [  24    0    6    0  938    0    4    0    0   10]
 [  41    1    0   16    0  824    6    0    2    2]
 [  20    2    3    0    3    5  924    1    0    0]
 [  39    2    7    5    3    0    0  967    0    5]
 [  43    2    3    3    6    7    3    2  904    1]
 [  49    6    0    6   10    3    0    8    1  926]]
Classification Report Mnist
              precision    recall  f1-score   support

           0       0.75      0.99      0.86       980
           1       0.99      0.98      0.98      1135
           2       0.97      0.95      0.96      1032
           3       0.96      0.94      0.95      1010
           4       0.98      0.96      0.97       982
           5       0.98      0.92      0.95       892
           6       0.98      0.96      0.97       958
           7       0.98      0.94      0.96      1028
           8       0.99      0.93      0.96       974
           9       0.98      0.92      0.95      1009

   micro avg       0.95      0.95      0.95     10000
   macro avg       0.96      0.95      0.95     10000
weighted avg       0.96      0.95      0.95     10000
```

```
Confusion MAtrix for Neural Networks USPS test:
[[1200    0   64   46  100  106   38   68   29  349]
 [ 760  241  330   55  146  106    9  302   29   22]
 [ 481    7 1325   62   14   44   19   22   23    2]
 [ 471    1  171 1137    1  173    2   18   17    9]
 [ 460    3   35   14  977  107    4  193  124   83]
 [ 482    2  234   84    9 1047   23   36   75    8]
 [ 703    3  337   27   34  129  712    4   14   37]
 [ 820   64  107  239   19   87    4  565   87    8]
 [ 864    1   96  204   32  296   33   60  395   19]
 [ 531    9   92  293   77   28    2  486  269  213]]
Classification Report USPS
              precision    recall  f1-score   support

           0       0.18      0.60      0.27      2000
           1       0.73      0.12      0.21      2000
           2       0.47      0.66      0.55      1999
           3       0.53      0.57      0.55      2000
           4       0.69      0.49      0.57      2000
           5       0.49      0.52      0.51      2000
           6       0.84      0.36      0.50      2000
           7       0.32      0.28      0.30      2000
           8       0.37      0.20      0.26      2000
           9       0.28      0.11      0.15      2000

   micro avg       0.39      0.39      0.39     19999
   macro avg       0.49      0.39      0.39     19999
weighted avg       0.49      0.39      0.39     19999
```

As seen from the matrix, the F1-scores are high for each class for the MNIST test set which indicates that the model is performing well. Again performance is a downgrade for the USPS data set as it is a complete set of 20000 unseen data. The model correctly classifies middle classes of 3,4,6 better than others.

3 Support Vector machines

I trained an SVM model using both ‘linear’ and ‘RBF’ kernels and a one vs all scheme. An SVM with linear kernel did exceptionally well on the training and testing with MNIST data set, but fared poorly for USPS data as seen below.

```
---------------------------------------SVM Results---------------------------------------------
Accuracy for Training 96.854
Accuracy for Testing 94.31
Accuracy for Validation 94.45
USPS Accuracy  30.306515325766288
Confusion MAtrix for SVM MNIST test:
```

On the contrary accuracy dropped for the SVM with a Gaussian Kernel but the performance increased by a fair margin for the USPS data set

```
Accuracy for Training 93.994
Accuracy for Testing 94.34
Accuracy for Validation 94.47
USPS Accuracy  38.53692684634232
```

2.1 Confusion Matrix

```
Confusion MAtrix for SVM MNIST test:
[[ 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    2    7    0  936    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]]
Classification Report Mnist
              precision    recall  f1-score   support

           0       0.96      0.99      0.97       980
           1       0.97      0.99      0.98      1135
           2       0.94      0.93      0.94      1032
           3       0.93      0.94      0.93      1010
           4       0.93      0.95      0.94       982
           5       0.93      0.91      0.92       892
           6       0.95      0.96      0.96       958
           7       0.95      0.93      0.94      1028
           8       0.94      0.91      0.93       974
           9       0.94      0.91      0.93      1009

   micro avg       0.94      0.94      0.94     10000
   macro avg       0.94      0.94      0.94     10000
weighted avg       0.94      0.94      0.94     10000
```

```
Confusion MAtrix for SVM USPS test:
[[ 573    2  429   19  285  248   73   44    6  321]
 [ 110  429  285  137  273  180   46  501   22   17]
 [ 128   18 1401   59   39  198   62   57   23   14]
 [  75    3  185 1124   11  483    5   71   27   16]
 [  18   67   91   13 1167  267   22  194   69   92]
 [ 108   17  257  102   25 1366   60   43   16    6]
 [ 197    7  489   24   98  394  748   13    7   23]
 [  50  225  457  265   57  416   15  452   41   22]
 [  73   25  209  193   87 1006   95   41  244   27]
 [  26  166  227  278  213  165    8  500  214  203]]
Classification Report USPS
              precision    recall  f1-score   support

           0       0.42      0.29      0.34      2000
           1       0.45      0.21      0.29      2000
           2       0.35      0.70      0.46      1999
           3       0.51      0.56      0.53      2000
           4       0.52      0.58      0.55      2000
           5       0.29      0.68      0.41      2000
           6       0.66      0.37      0.48      2000
           7       0.24      0.23      0.23      2000
           8       0.36      0.12      0.18      2000
           9       0.27      0.10      0.15      2000

   micro avg       0.39      0.39      0.39     19999
   macro avg       0.41      0.39      0.36     19999
weighted avg       0.41      0.39      0.36     19999
```

4 Random forests

I implement a random forest using the SCIKIT Learns RandomForestClassifier Package.
Initial settings: No of estimators (trees) 10, Criterion for splitting: ‘GINI’
The results for these were as follows:

```
--------------------------------------------Random Forest Results--------------------------------------------
Accuracy for Training 99.908
Accuracy for Testing 94.59
Accuracy for Validation 94.97
USPS Accuracy  31.82159107955398
```

As seen from the image, the random forests produces results quickly and performs well on MNIST data. The reason the testing accuracy is almost a certainty because the model forms a path (Tree) observing each of the data from the set. Now I try to tweak the model to improve its accuracy on the USPS data set.

The split function both perform reasonably same, with gini index giving marginally better accuracy on MNIST and USPS test set.

An increase in the number of estimators (trees) yields an improved accuracy for both the test set. This is because as we increase this number, we divide the training samples in more number of subsets and thus get more number of decision trees or paths to the leaf node. Hence after increasing the no of estimators to 200, best results were observed on the validation and test sets.

Since, random forests are more susceptible to overfitting, I used the max_depth attribute to limit the tree depth so that the model can retain some noise. This however had little impact on the test set accuracy.

Below is the best result obtained for this model: (Estimators = 200, criterion=gini)

```
------------------------------------------Random Forest Results------------------------------------------
Accuracy for Training 100.0
Accuracy for Testing 96.72
Accuracy for Validation 97.15
USPS Accuracy  40.07200360018001
```

4.1 Confusion Matrix:

As seen below, this model indentifies class 2,3,4,5,6 better than others as it has high F1 score for that class. As the average f1 scores, recall and precision values lie in below 0.5 region, I conclude the performance is avergage.

```
Confusion MAtrix for Random Forest test:
[[ 708    6  195   48  349  186   99   85    3  321]
 [  52  568  179  114   71  103   24  871   14    4]
 [ 151   33 1239   70   52  193   36  205    8   12]
 [  42    7  110 1176   75  397    6  138    4   45]
 [  19  238   60   34  993  177   28  388   24   39]
 [ 149   25  116   43   17 1517   42   73    9    9]
 [ 397   50  214   25   73  288  836   96    5   16]
 [  75  296  423  250   36  229   41  633    4   13]
 [  85   52  122  224   96  993  111   99  175   43]
 [  30  311  269  345  147  150   14  482   83  169]]
Classification Report USPS
              precision    recall  f1-score   support

           0       0.41      0.35      0.38      2000
           1       0.36      0.28      0.32      2000
           2       0.42      0.62      0.50      1999
           3       0.50      0.59      0.54      2000
           4       0.52      0.50      0.51      2000
           5       0.36      0.76      0.49      2000
           6       0.68      0.42      0.52      2000
           7       0.21      0.32      0.25      2000
           8       0.53      0.09      0.15      2000
           9       0.25      0.08      0.13      2000

   micro avg       0.40      0.40      0.40     19999
   macro avg       0.42      0.40      0.38     19999
weighted avg       0.42      0.40      0.38     19999
```

```
Confusion MAtrix for Random Forest MNIST test:
[[ 973    1    0    0    0    2    1    1    2    0]
 [   0 1121    1    4    1    2    3    1    2    0]
 [   6    1  999    5    2    1    4    9    5    0]
 [   1    0    9  970    0   11    0    9    8    2]
 [   1    0    1    0  954    0    5    0    3   18]
 [   6    2    0   12    2  850    5    3    7    5]
 [   9    3    0    0    5    4  929    0    8    0]
 [   1    6   19    1    1    0    0  990    4    6]
 [   4    1    5    4    5    7    6    2  928   12]
 [   7    5    3   11   12    5    1    3    4  958]]
Classification Report Mnist
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       980
           1       0.98      0.99      0.99      1135
           2       0.96      0.97      0.97      1032
           3       0.96      0.96      0.96      1010
           4       0.97      0.97      0.97       982
           5       0.96      0.95      0.96       892
           6       0.97      0.97      0.97       958
           7       0.97      0.96      0.97      1028
           8       0.96      0.95      0.95       974
           9       0.96      0.95      0.95      1009

   micro avg       0.97      0.97      0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000
```

5 Questions

1. Yes, in all the models before, the model trained on MNIST data doesn't perform very well on USPS data set. This agrees with the No free lunch theorem which says that no single algorithm can perform very well on every data set.
2. To compare the algorithms, I will consider the F1 scores of each of these models for both the tests sets. For both these data sets, the Random Forest Classifier performs the best

which is validated by its accuracy on the test sets of both these sets.

6 Combined Ensemble

I am implementing a combined ensemble using a majority voting scheme. I make use of the outputs from the previous models instead of using the models themselves to avoid unnecessary preprocessing required. In the majority voting scheme, a class gets assigned on the basis of the common opinion of each of each model. This means that if out the 4 predictions for an example if one class gets the majority that class is selected as the prediction. In case of tie, the class with lower number gets assigned.

In my implementation, a marginal improvement can be seen on the test and USPS data as seen below.

```
-----------------------------------Majority Voting Ensemble Results----------------------------------
Accuracy on MNIST 0.9624
Accuracy on USPS 0.3377
```