Machine Learning CSE574

David Jegan Abishek Dominique
Person no 50290785
davidjeg@buffalo.edu

Abstract

The report provides a detailed insight on how combination of Classifier algorithms can be used to recognize an handwritten digit image.

1 Introduction

1.1 Handwritten digit image detection

This report addresses how we could recognize an handwritten digit image using four classifiers, namely Logistic regression, Neural networks, Support vector machines, Random forest classification. The algorithm detects the digit as 0,1,...9 based on its training. So the input is an 28*28 grayscale handwritten image, the model is a multi-class classifier and the output is a scalar and discrete value in the range (0,9).

1.2 Approach

There are three stages in this problem

1. Data Preparation
2. Classification Algorithms
   (a) Multiclass Logistic regression
   (b) Deep Learning Neural network
   (c) Convolutional neural network
   (d) Random Forest Classification
   (e) Support Vector Machines
   (f) Classifier Combination
3. Conclusion

2 Data Preparation

There are two sources of handwritten digit data for this project. One is the standard MNIST dataset used for training and testing, and the latter is the USPS data used only for testing. All the images contain 28*28 features extracted from each pixel.

MNIST dataset : We consider 70,000 handwritten samples for the MNIST dataset where each digit is from range 0 to 9. We divide the image set into 80,10 and 10 percents of actual data called training, validation and testing result set respectively, in such a way that the sets do not overlap.

USPS dataset : We consider the USPS dataset consisting of 19999 handwritten digit image samples from digits 0 to 9. There are around 200 images under each category of (0,9). The previously built model is compared with the USPS datasets to validate the accuracy of our model.

From the image, the feature set is formed with dimensions being (number of samples * 784). The target is extracted from the input files on the same lines using the python script and has the dimensions (number of samples * 1) with value in the range (0,9).

3 Classification Algorithms

3.1 Multiclass Logistic Regression

Definition

In this approach, we build a multiclass logistic regression model, which can predict the digits classes as (0,1,...,9) by finding and updating the optimal weights. The updation is iteratively done by finding the softmax and gradiant values at each iteration.

Terminologies

1. **Genesis equation**

$$y = softmax(a)$$

where ϕ is the input set , W is the weight, y is the output (target) and softmax denotes the formula,

$$softmax(a) = \frac{e^{a_j}}{\sum_{j=1}^{10} e^{a_j}}$$

and $a = w^T\phi + b$

2. **Loss function** The loss function used in the multiclass logistic regression is Cross entropy. It is calculated using the following formula,

$$L = -\sum_{i=1}^{N} y.loga + (1-y).log(1-a)$$

Where a denotes $(w^T * x)$, and the backward travel from L to w_i is done by using partial differential of L with respect to w_i . Thus $\Delta w_i = x_i(a-y)$, where a is the predicted target and y is the actual target.

3. **Bias** We add a bias term to the feature set as an added noise. This is added to the feature matrix and the size is denoted by the number of sample records.

4. **Epoch and Blocksize** The number of iterations done on the training set is called epoch. Batchsize is the number of sample taken for processing by the model at a time.

Epoch	MNIST Accuracy	USPS Accuracy
50	72%	28%
100	76%	29%
300	79%	32%

5. **Regularization** Regularization term avoid overfitting, and decreases the complexity of the model.

Lambda	MNIST Accuracy	USPS Accuracy
0.1	76%	30%
2	79%	32%

6. **One hot encoding** The target is scalar and discrete, and this needs to be converted into a binary array of class 10. So we use one hot encoding to convert it into an array.

7. **Softmax** The softmax function is for multi class classification, and it determines the probabilities for each class score.

Steps

- For logistic regression, we convert the target scalar into an 10 dimensional array.
- We identify the optimal weight by iteratively calculating the loss and trying to minimizing the loss.
- The results are then validated and tweaked using the validation data.
- Then the prediction is determined for the MNIST and USPS test data, and the respective classes are predicted
- This result is then compared with the actual target values and the confusion matrix with classification report is generated

Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Predicted class

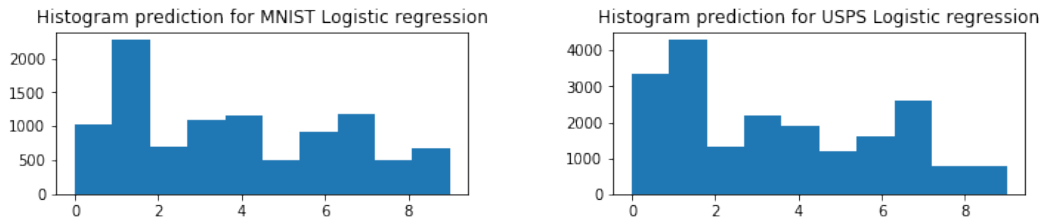

Figure 1: MNIST (left) and USPS (right) Prediction for Logistic Regression

Confusion Matrix

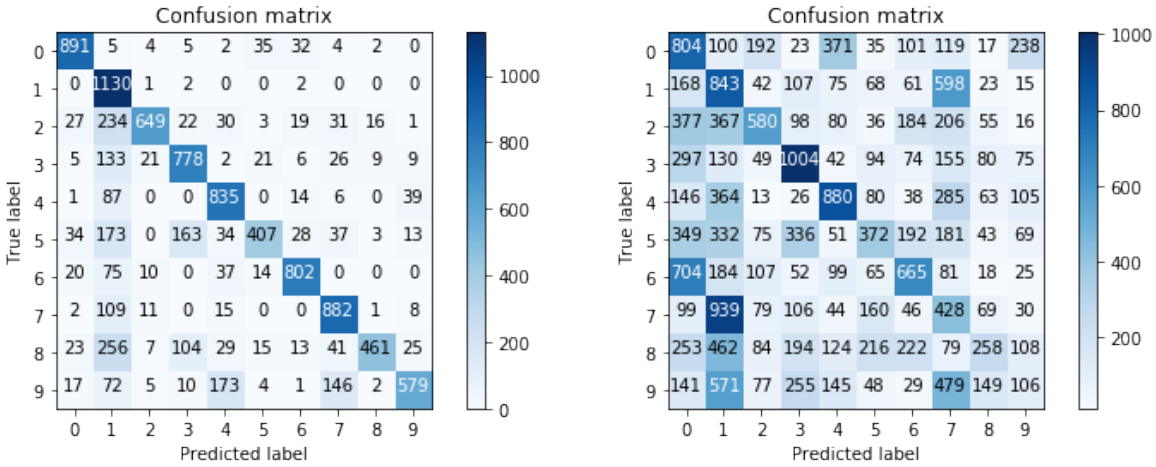

Figure 2: MNIST (left) and USPS (right) Confusion Matrix for Logistic Regression

Classification Report

	precision	recall	f1-score	support
0	0.87	0.91	0.89	980
1	0.50	1.00	0.66	1135
2	0.92	0.63	0.75	1032
3	0.72	0.77	0.74	1010
4	0.72	0.85	0.78	982
5	0.82	0.46	0.59	892
6	0.87	0.84	0.86	958
7	0.75	0.86	0.80	1028
8	0.93	0.47	0.63	974
9	0.86	0.57	0.69	1009
avg / total	0.79	0.74	0.74	10000

	precision	recall	f1-score	support
0	0.24	0.40	0.30	2000
1	0.20	0.42	0.27	2000
2	0.45	0.29	0.35	1999
3	0.46	0.50	0.48	2000
4	0.46	0.44	0.45	2000
5	0.32	0.19	0.23	2000
6	0.41	0.33	0.37	2000
7	0.16	0.21	0.19	2000
8	0.33	0.13	0.19	2000
9	0.13	0.05	0.08	2000
avg / total	0.32	0.30	0.29	19999

Figure 3: MNIST (left) and USPS (right) Classification Report for Logistic Regression

The Logistic regression algorithm has predicted MNIST data with an accuracy of 79%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 32%. In addition we observe that the class 1 and 3 has been predicted the highest in USPS data.

3.2 Deep Learning Neural Network

Definition

Deep Learning Neural networks is a multi-layer based neural network prediction model. A network of entities named Neurons, are initialized and are responsible for running mathematical equations. There are layers of these neurons, and are connected to the next level via weights.

Terminologies

1. **Activation** The ML model consists of a network of neurons. Each neuron is linked with the next level via weights. The output for a neuron is called activation. Based on this output we decide an

algorithm which is sensitive to input and has faster computation time. Sigmoid is used here as an activation function, and it is used to normalize(filter).

2. **Layers of NN** There are three types of layers, ie., Input, Hidden and Output layers. All three layers, input and Output are single layered and it directly maps to the inputs and outputs respectively, whereas the hidden layer is multi-layered and it leverages the sigmoid method for the input layer and softmax converts it into a context to be used by the output layer.

Layers	MNIST Accuracy	USPS Accuracy
2	94%	42%
3	95%	42%

3. **Weights** The weights connecting the neurons help to find the output. These values are updated via a method called Back-propagation. It can be initialized randomly or using a trained weight.

4. **Cost Function** This method helps to increase the accuracy of the model. The value must be low in order to mitigate errors. We use categorical_cross_entropy cost function in this model.

5. **Epoch and Blocksize** The number of iterations done on the training set is called epoch. Batchsize is the number of sample taken for processing by the model at a time.

Epoch	MNIST Accuracy	USPS Accuracy
50	92%	37%
100	93%	40%
200	95%	42%

Batch size	MNIST Accuracy	USPS Accuracy
512	92%	39%
256	94%	40%
64	95%	42%

6. **One hot encoding** The target is scalar and discrete, and this needs to be converted into a binary array of class 10. So we use one hot encoding to convert it into an array.

Steps

- For DNN we add the input, hidden and output layer, optimizer, loss and the activation function. We convert the target into a binary array of class 10 using one hot encoding.
- We use the model implementation in Sklearn. The initialization has the deep learning layer, dense and compile methods initialized.
- The fit() is used for training the model, and this model is tested with the validation data and tweaked accordingly
- Then the prediction is determined for the MNIST and USPS test data, and the respective classes are predicted
- This result is then compared with the actual target values and the confusion matrix with classification report is generated

Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Predicted class

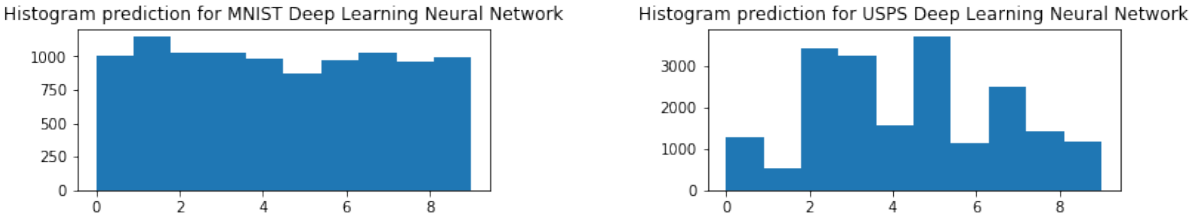

Figure 4: MNIST (left) and USPS (right) Prediction for DNN

Confusion Matrix

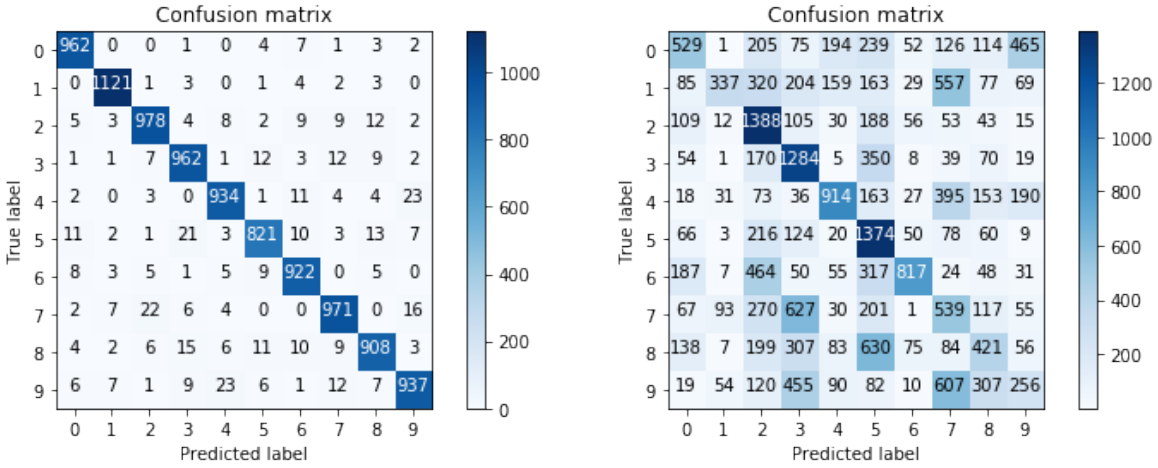

Figure 5: MNIST (left) and USPS (right) Confusion Matrix for DNN

Classification Report

	precision	recall	f1-score	support
0	0.96	0.98	0.97	980
1	0.98	0.99	0.98	1135
2	0.96	0.95	0.95	1032
3	0.94	0.95	0.95	1010
4	0.95	0.95	0.95	982
5	0.95	0.92	0.93	892
6	0.94	0.96	0.95	958
7	0.95	0.94	0.95	1028
8	0.94	0.93	0.94	974
9	0.94	0.93	0.94	1009
avg / total	0.95	0.95	0.95	10000

	precision	recall	f1-score	support
0	0.42	0.26	0.32	2000
1	0.62	0.17	0.26	2000
2	0.41	0.69	0.51	1999
3	0.39	0.64	0.49	2000
4	0.58	0.46	0.51	2000
5	0.37	0.69	0.48	2000
6	0.73	0.41	0.52	2000
7	0.22	0.27	0.24	2000
8	0.30	0.21	0.25	2000
9	0.22	0.13	0.16	2000
avg / total	0.42	0.39	0.38	19999

Figure 6: MNIST (left) and USPS (right) Classification Report for DNN

The DNN algorithm has predicted MNIST data with an accuracy of 95%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 42%. In addition we observe that the class 2 and 3 has been predicted the highest in USPS data.

Strengths/weaknesses

- DNN has better performance with reduce computational complexity and can work well with very large datasets
- However, model requires very large dataset to train and is computationally expensive.

3.3 CNN

Definition

Convolutional Neural networks is a class of feed forward neural networks. They process dataset as tensors, where each tensor is a matrix of numbers. The CNN algorithm slices the input dataset features, to create a map of feature occurrence space. The image considered is grayscale, so the input shape is (28,28,1), where 1 denotes the depth (grayscale - absence of rgb)

Terminologies

1. **Convolution layer** : The convolution layer extracts the features from the input dataset. It preserves the relationship between pixels by learning the features from the small subsets of input data.
2. **Filter** : Analogous to kernels in SVM, the filter is responsible for finding some pattern in a group of features. It determines the dimensionality of the output filter in the convolution

Filter	MNIST Accuracy	USPS Accuracy
32	96%	43%
64	96%	45%

3. **Kernel_size** : The kernel size determines the length of the convolution window. A large kernel size denotes that the key features are overlooked and a low kernel size has low signifiance for the key features. So we choose an optimal kernal size for best fit.

Kernal size	MNIST Accuracy	USPS Accuracy
2	93%	43%
3	96%	45%

4. **Dense layer** : The dense layer count determines the layer of neurons in the neural network. Every neuron is densely connected with all neurons from the previous layer. Activation functions in the Neural network like Relu introduce non-linearity in the CNN model.

Layers	MNIST Accuracy	USPS Accuracy
1	95%	43%
2	96%	45%

5. **One hot encoding** The target is scalar and discrete, and this needs to be converted into a binary array of class 10. So we use one hot encoding to convert it into an array.
6. **Pooling** : If images are too large, we use pooling to reduce the parameters, via methods like subsampling or downsampling. The dimensionality is reduced but important features are retained.

7. **Flatten** : We flatten our matrix into vector and introduce non-linearity in the dataset. This is a fully connected network and is used to create a complex model. This has a softmax activation function to output the prediction.

Steps

- CNN is a class of feed forward neural network. Thus we specify the convolutional layer, kernal size, filters and the activation function. We convert the target into a binary array of size 10 using one hot encoding.

- We use the model implementation in Sklearn. The initialization has the convolutional layer, flatten, dense and compile methods initialized.

- The fit() is used for training the model, and this model is tested with the validation data and tweaked accordingly

- Then the prediction is determined for the MNIST and USPS test data, and the respective classes are predicted

- This result is then compared with the actual target values and the confusion matrix with classification report is generated

Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Predicted class

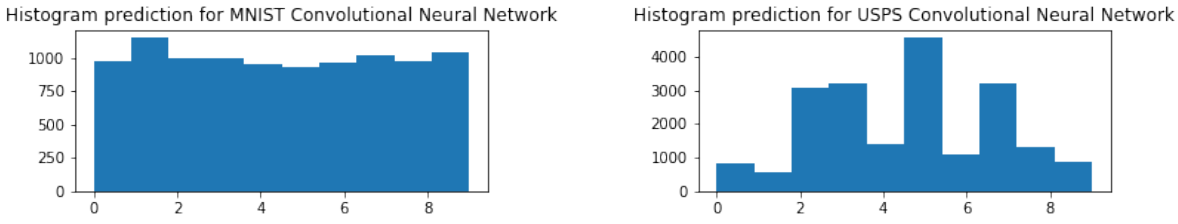

Figure 7: MNIST (left) and USPS (right) Prediction for CNN

Confusion Matrix

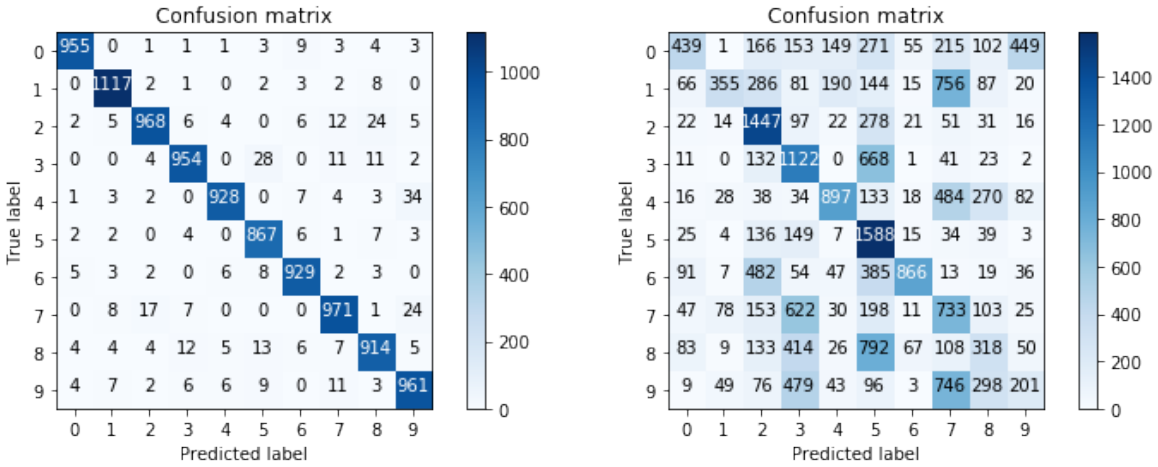

Figure 8: MNIST (left) and USPS (right) Confusion Matrix for CNN

Classification Report

	precision	recall	f1-score	support
0	0.98	0.97	0.98	980
1	0.97	0.98	0.98	1135
2	0.97	0.94	0.95	1032
3	0.96	0.94	0.95	1010
4	0.98	0.95	0.96	982
5	0.93	0.97	0.95	892
6	0.96	0.97	0.97	958
7	0.95	0.94	0.95	1028
8	0.93	0.94	0.94	974
9	0.93	0.95	0.94	1009
avg / total	0.96	0.96	0.96	10000

	precision	recall	f1-score	support
0	0.54	0.22	0.31	2000
1	0.65	0.18	0.28	2000
2	0.47	0.72	0.57	1999
3	0.35	0.56	0.43	2000
4	0.64	0.45	0.53	2000
5	0.35	0.79	0.48	2000
6	0.81	0.43	0.56	2000
7	0.23	0.37	0.28	2000
8	0.25	0.16	0.19	2000
9	0.23	0.10	0.14	2000
avg / total	0.45	0.40	0.38	19999

Figure 9: MNIST (left) and USPS (right) Classification Report for CNN

The CNN algorithm has predicted MNIST data with an accuracy of 96%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 45%. In addition we observe that the class 2 and 5 has been predicted the highest in USPS data.

Strengths/weaknesses

- CNN tends to share parameters across different functions. CNN focuses on relevant features and is efficient than a fully connected neural network. Also, a small change in the input changes the output (equivariant)
- However, the computation cost is high, and large training data is required.

3.4 SVM

Definition

Support Vector Machine is a supervised algorithm used for classification. The data points are plotted

in a n-dimensional space, where the particular coordinate is decided by the features. A hyperplane is formed with helps to segregate the data points into classes.

Terminologies

1. **Kernal** : Kernals are functions which seperate non seperable problems into seperable problems by finding a right hyperplane seperating the datapoints. The various kernal types are linear, radial and poly, where each type determines the type of hyperplane the function generates and the type of equation involved to determine the hyperplane.

Kernal	MNIST Accuracy	USPS Accuracy
Linear	98%	52%
Radial (G = 0.03)	98%	53%

2. **Gamma** : The gamma tries to fit the hyperplane with the training data. The greater the gamma value, the hyperplane fits the training data exactly leading to overfitting. If the gamma value is low, then the farthest points are considered for calculating hyperplane, similarily high gamma value, selects the points near to the probable seperation line.

Gamma	MNIST Accuracy	USPS Accuracy
Linear	96%	52%
Radial (G = 0.03)	98%	53%
Radial (G = 1)	98%	53%

3. **Regularization parameter (C)** : The error term is responsible for smoothing the decision boundary. The value of C decides the margin of hyperplane selected. Small values of C, tends to choose large margin hyperplane, and large values of C returns a small margin hyperplane.

Regularization	MNIST Accuracy	USPS Accuracy
1	96%	51%
2	98%	53%

4. **Margin** : The line separating the closet class points is called a margin and an algorithm tries to find a good margin. If the separation is large for all classes and all points are in their classes without moving on to the next class, then that margin is maximized and the optimal hyperplane.

5. **Hyperplane** : A hyperplane divides the space into seperate parts for classification. It is (n-1) dimensional subplane in an n dimensional space.

Steps

- The data here is not linearly seperable, so we map the samples into another dimensional space using kernals like Radial basis function (RBF) . The dataset is large with high number of features, and therefore this kernal mode is time consuming than linear kernal. Yet radial basis provides us better accuracy.
- We use the SVM implementation in Sklearn. The initialization has the parameters like kernal, C and gamma initialized. The fit() is used for training the model, and this model is tested with the validation data and tweaked accordingly

- Then the prediction is determined for the MNIST and USPS test data, and the respective classes are predicted

- This result is then compared with the actual target values and the confusion matrix with classification report is generated

Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Predicted class

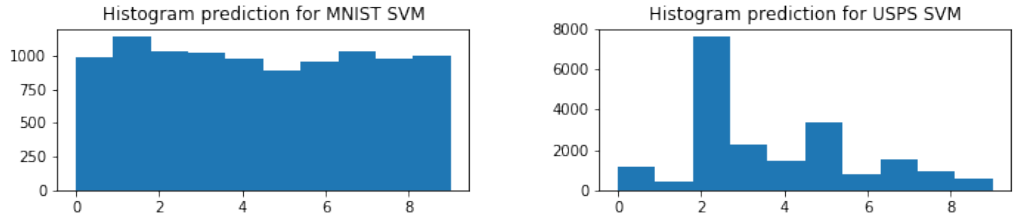

Figure 10: MNIST (left) and USPS (right) Prediction for SVM

Confusion Matrix

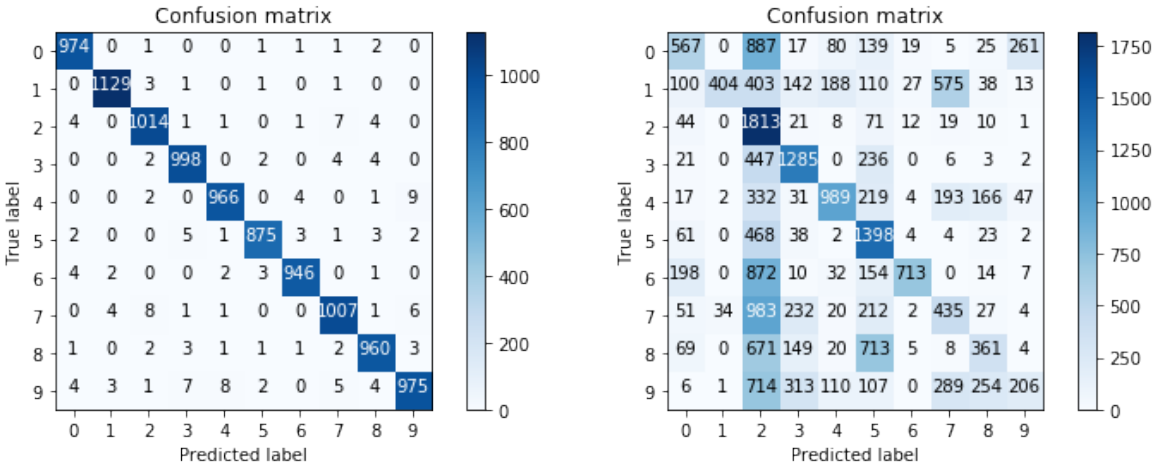

Figure 11: MNIST (left) and USPS (right) Confusion Matrix for SVM

Classification Report

	precision	recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.99	0.99	0.99	1135
2	0.98	0.98	0.98	1032
3	0.98	0.99	0.99	1010
4	0.99	0.98	0.98	982
5	0.99	0.98	0.98	892
6	0.99	0.99	0.99	958
7	0.98	0.98	0.98	1028
8	0.98	0.99	0.98	974
9	0.98	0.97	0.97	1009
avg / total	0.98	0.98	0.98	10000

	precision	recall	f1-score	support
0	0.50	0.28	0.36	2000
1	0.92	0.20	0.33	2000
2	0.24	0.91	0.38	1999
3	0.57	0.64	0.61	2000
4	0.68	0.49	0.57	2000
5	0.42	0.70	0.52	2000
6	0.91	0.36	0.51	2000
7	0.28	0.22	0.25	2000
8	0.39	0.18	0.25	2000
9	0.38	0.10	0.16	2000
avg / total	0.53	0.41	0.39	19999

Figure 12: MNIST (left) and USPS (right) Classification Report for SVM

The SVM algorithm has predicted MNIST data with an accuracy of 98%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 53%. In addition we observe that the class 2 has been predicted the highest in USPS data.

Strengths/weaknesses

- SVM has great accuracy for MNIST dataset, as it is effective in high dimensional space. Also the training is done on a subset of points, so it is very memory efficient
- However, the runtime is high for this large dataset, because the time for training is high. When noise is added to this data, the performance degrades

3.5 Random forest

Definition

Random forests is a supervised learning algorithm. A forest comprises of various decision trees and is created with random data samples to create a best possible model for a given dataset. The more the number of decision trees, the more robust the algorithm is. The random forest is an example of bagging, as it performs internal bagging creating an ensemble of decision trees on randomly split data. We then take a vote for predicted result of each decision tree and the most vote is chosen as the final prediction.

Terminologies

1. **N_Estimators** : The n_estimator is the number of trees in the forest that the model builds before taking the maximum voting. The default value is 10, but we increase the trees to 100. The increase in n_estimator would increase the prediction accuracy as well as the computation time.

N Estimators	MNIST Accuracy	USPS Accuracy
15	96%	40%
50	96%	41%
100	97%	43%

2. **Criterion** : Decision tree requires constructing a tree, based on the features. So criterion helps us to find the important feature to improve the quality of dataset split. The default "Gini" (mean

decrease in impurity) determines how much the accuracy decreases when a variable is dropped, this determines the significance of the variable. The lower the Gini index, more pure the node is. Another type of criterion named "entropy" is an computationally intensive, exploratory analysis, suitable for attributes occurring in classes.

Criterion	MNIST Accuracy	USPS Accuracy
Gini (Default)	95%	40%
Entropy	97%	43%

3. **min_sample_split** : Inorder to avoid overfitting of the model, we specify the minimum number of features to be considered while splitting the data for decision tree. With increase of min samples, the model avoids overfitting by regularizing the model.
4. **max_leaf_nodes** :Inorder to avoid drilling down to single features, we set a max leaf node count. This prevents model overfitting.
5. **max_depth** : The max depth option specifies the depth of the decision tree, thus preventing the model overfitting.

Steps

- Random forest is an ensembler of decision trees. Thus we specify the maximum number of trees for one forest. This value is given as 100, so that there are 100 trees involved in each voting. The criterion is chosen as entropy, as it is suitable for attributes occurring in classes.
- We use the from RandomForestClassifier ensemble implementation in Sklearn. The initialization has the parameters like n_estimator and criterion initialized.
- The fit() is used for training the model, and this model is tested with the validation data and tweaked accordingly
- Then the prediction is determined for the MNIST and USPS test data, and the respective classes are predicted
- This result is then compared with the actual target values and the confusion matrix with classification report is generated

Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Predicted class

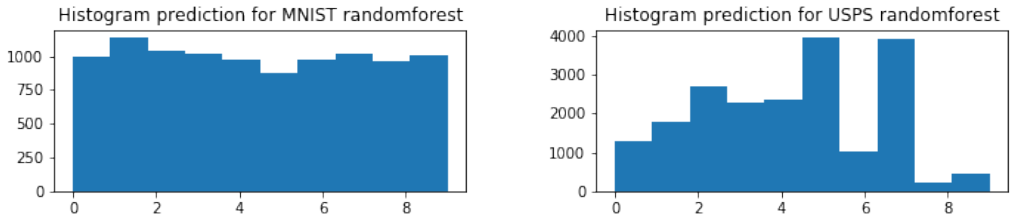

Figure 13: MNIST (left) and USPS (right) Prediction for Random Forest

Confusion Matrix

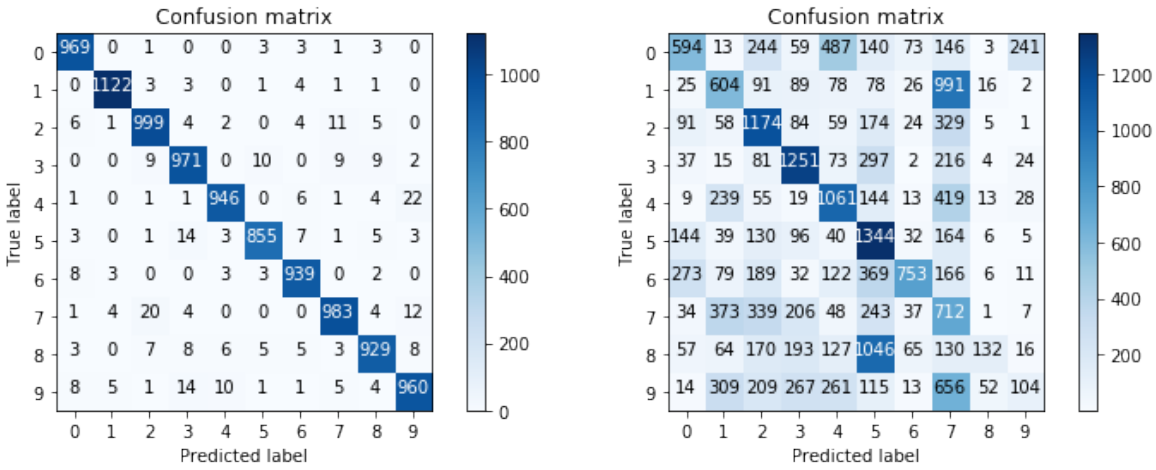

Figure 14: MNIST (left) and USPS (right) Confusion Matrix for Random Forest

Classification Report

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.96	0.97	0.96	1032
3	0.95	0.96	0.96	1010
4	0.98	0.96	0.97	982
5	0.97	0.96	0.97	892
6	0.97	0.98	0.97	958
7	0.97	0.96	0.96	1028
8	0.96	0.95	0.96	974
9	0.95	0.95	0.95	1009
avg / total	0.97	0.97	0.97	10000

	precision	recall	f1-score	support
0	0.46	0.30	0.36	2000
1	0.34	0.30	0.32	2000
2	0.44	0.59	0.50	1999
3	0.54	0.63	0.58	2000
4	0.45	0.53	0.49	2000
5	0.34	0.67	0.45	2000
6	0.73	0.38	0.50	2000
7	0.18	0.36	0.24	2000
8	0.55	0.07	0.12	2000
9	0.24	0.05	0.09	2000
avg / total	0.43	0.39	0.36	19999

Figure 15: MNIST (left) and USPS (right) Classification Report for Random Forest

The Randomforest algorithm has predicted MNIST data with an accuracy of 97%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 43%. In addition we observe that the class 3 and 5 has been predicted the highest in USPS data.

Strengths/weaknesses

- Random forest does not have overfitting problem, because of the n_estimators count. Also we can obtain the most important feature for the classifier, returned via the Gini index. The number of trees and the identifying the import features, make the prediction to be highly accurate.
- However, it is computationally intensive and time consuming. It is difficult to interpret the forest, as the choice of dataset is random.

3.6 Classifier Combination

Hard Voting

- Hard voting is done by finding the maximum repetition class of all 5 classifier targets. The combined targets are considered, so that the accuracy is increased.

  Results:

  There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

  Confusion Matrix

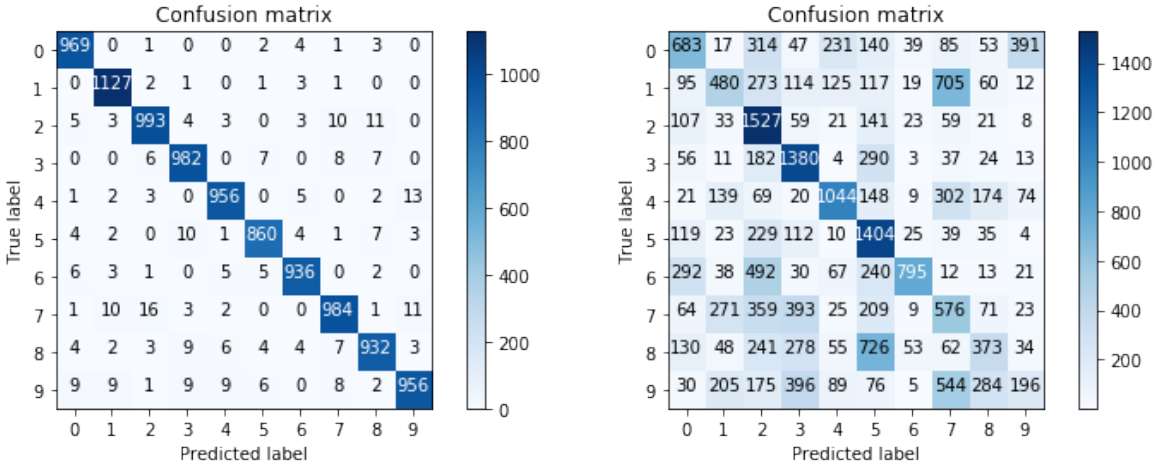

Figure 16: MNIST (left) and USPS (right) Confusion Matrix for Hard Voting

Classification Report

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.97	0.99	0.98	1135
2	0.97	0.96	0.97	1032
3	0.96	0.97	0.97	1010
4	0.97	0.97	0.97	982
5	0.97	0.96	0.97	892
6	0.98	0.98	0.98	958
7	0.96	0.96	0.96	1028
8	0.96	0.96	0.96	974
9	0.97	0.95	0.96	1009
avg / total	0.97	0.97	0.97	10000

	precision	recall	f1-score	support
0	0.43	0.34	0.38	2000
1	0.38	0.24	0.29	2000
2	0.40	0.76	0.52	1999
3	0.49	0.69	0.57	2000
4	0.62	0.52	0.57	2000
5	0.40	0.70	0.51	2000
6	0.81	0.40	0.53	2000
7	0.24	0.29	0.26	2000
8	0.34	0.19	0.24	2000
9	0.25	0.10	0.14	2000
avg / total	0.44	0.42	0.40	19999

Figure 17: MNIST (left) and USPS (right) Classification Report for Hard Voting

The hard voting has predicted MNIST data with an accuracy of 97%, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 44%. In addition we observe that the class 2 has been predicted the highest in USPS data.

Soft Voting

- Soft voting is done by finding the average of the probabilites of all 5 classifier targets, and finding the class of the highest probability term. The combined targets are considered, so that the accuracy is increased compared to individual classifiers. Also compared to hard voting, soft voting has more features for consideration, so the accuracy is higher than hard voting.

  Results:

There are digits from the range 0 to 9. The MNIST has around 1000 entries for each digit class which may change as we generate the test dataset randomly, and the USPS dataset has 2000 entries for each digit class.

Confusion Matrix

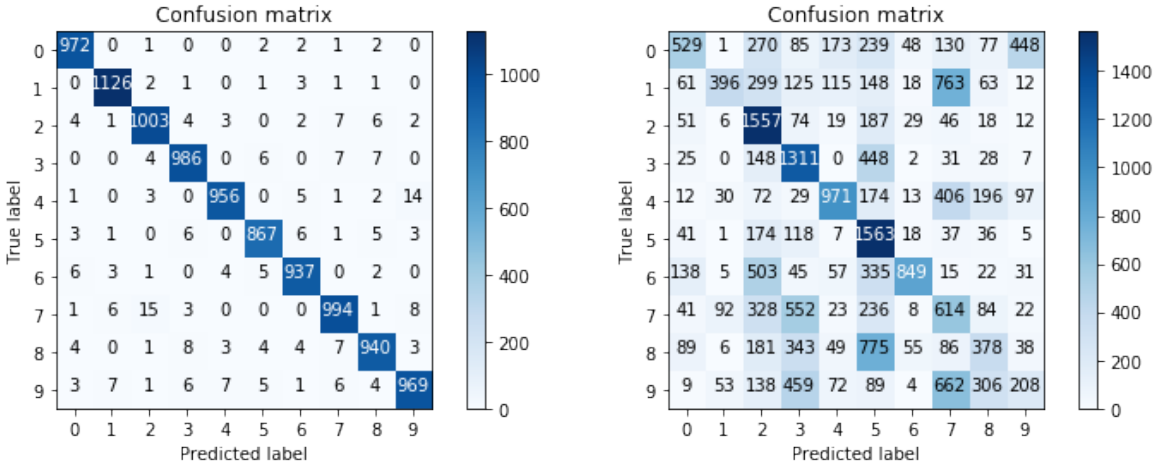

Figure 18: MNIST (left) and USPS (right) Confusion Matrix for Soft Voting

Classification Report

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.98	0.99	0.99	1135
2	0.97	0.97	0.97	1032
3	0.97	0.98	0.97	1010
4	0.98	0.97	0.98	982
5	0.97	0.97	0.97	892
6	0.98	0.98	0.98	958
7	0.97	0.97	0.97	1028
8	0.97	0.97	0.97	974
9	0.97	0.96	0.97	1009
avg / total	0.97	0.97	0.97	10000

	precision	recall	f1-score	support
0	0.53	0.26	0.35	2000
1	0.67	0.20	0.31	2000
2	0.42	0.78	0.55	1999
3	0.42	0.66	0.51	2000
4	0.65	0.49	0.56	2000
5	0.37	0.78	0.50	2000
6	0.81	0.42	0.56	2000
7	0.22	0.31	0.26	2000
8	0.31	0.19	0.24	2000
9	0.24	0.10	0.14	2000
avg / total	0.47	0.42	0.40	19999

Figure 19: MNIST (left) and USPS (right) Classification Report for Soft Voting

The soft voting has predicted MNIST data with an accuracy of 96 %, as shown in the Figure (left). But on the other hand for MNIST data, the accuracy is only 47%, compared with 97 % and 44 % of Hard voting. In addition we observe that the class 2 and 5 has been predicted the highest in USPS data.

- Also, we conclude that classes 2 and 5 have the highest accuracies of all the classes and class 9 has the least predicted accuracy, across all classifiers.

4 Conclusion

The 'No free lunch' theorem states that there is no single model that works best for all problem. The assumptions considered for a great model of one problem, may not be applicable for another problem. So in ML it is quite common, to try various models and find the one that works the best for a particular problem. A great model for a problem, may be based on dataset and inferences in certain conditions derived from

the dataset. This inference may not hold good for another model for the same problem. Thus validation is required to predict accuracy of many models to find the best solution for the problem.

The accuracies observed for the following models:

Classifier	MNIST Accuracy	USPS Accuracy
Logistic Regression	79%	32%
DNN	95%	42%
CNN	96%	45%
SVM	98%	53%
Random Forest	97%	43%
Hard Voting	95%	44%
Soft Voting	97%	47%

Thus our model validates the no free lunch theorem

References

1. https://cedar.buffalo.edu/ srihari/CSE574/Chap3/3.2-GradientDescent.pdf
2. https://cedar.buffalo.edu/ srihari/CSE574/Chap3/3.1-Regression-BasisFns.pdf
3. https://cedar.buffalo.edu/ srihari/CSE574/Chap4/4.3.2-LogisticReg.pdf
4. https://medium.com/@awjuliani/simple-reinforcement-learning-with-tensorflow-part-4-deep-q-networks-and-beyond-8438a3e2b8df
5. https://towardsdatascience.com/understanding-convolutional-neural-networks-221930904a8e
6. https://www.kaggle.com/alanferbach/97-accuracy-using-keras-and-scikit-learn
7. https://towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras-329fbbadc5f5
8. https://www.kaggle.com/moghazy/guide-to-cnns-with-data-augmentation-keras
9. https://keras.io/layers/about-keras-layers/
10. https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python
11. https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd
12. https://chemicalstatistician.wordpress.com/2014/01/24/machine-learning-lesson-of-the-day-the-no-free-lunch-theorem/