CSE 474 Machine Learning Project 3

University at Buffalo

Kevin Yiu-Wah Cheung (kcheung8)

UB Peronal Number: 50148100

Nov 15, 2018

Contents

1 Introduction

1.1 Description

This project is to implement machine learning methods for the task of classification. You will first implement an ensemble of four classifiers for a given task. Then the results of the individual classifiers are combined to make a final decision.

The classification task will be that of recognizing a 28x28 grayscale handwritten digit image and identify it as a digit among 0, 1, 2, ... , 9. You are required to train the following four classifiers using MNIST digit images.

1. Logistic regression, which you implement yourself using backpropagtion and tune hyperparameters.
2. A publicly available multilayer perceptron neural network, train it on the MNIST digit images and tune hyperparameters.
3. A publicly available Random Forest package, train it on the MNIST digit images and tune hyperparameters.
4. A publicly available SVM package, train it on the MNIST digit images and tune hyperparameters.

1.2 Tasks

1. MNIST trained models on two different test sets: the test set from MNIST and a test set from the USPS data set. Do your results support the "No Free Lunch Theorem"?

2. Observe the confusion matrix of each classifier and describe the relative strengths/weaknesses of each classifier. Which classifier has the overall best performance?

3. Combine the results of the individual classifiers using a classifier combination method such as majority voting. Is the overall combined performance better than that of any individual classifier?

1.3 Datasets

1.3.1 MNIST Data

For both training and testing of our classifiers, we will use the MNIST dataset. The MNIST database is a large database of handwritten digits that is commonly used for training various image processing systems.The database is also widely used for training and testing in the field of machine learning.

The database contains 60,000 training images and 10,000 testing images. The dataset could be downloaded from here: `http://yann.lecun.com/exdb/mnist/`

The original black and white (bilevel) images from MNIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. The images were centered in a 28x28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field.

1.3.2 USPS Data

We use USPS handwritten digit as another testing data for this project to test whether your models could be generalize to a new population of data. Examples of each of the digits are given below. The dataset will be available on UBLearns.

Each digit has 2000 samples available for testing. These are segmented images scanned at a resolution of 100ppi and cropped. Resize or fill the images to 28x28 like MNIST digits and feed this into your trained model and compare the result on USPS data and MNIST test data.

1.4 Plan of Work

1. **Download Data:** Download the MNIST dataset from the Internet and download the USPS data set from ublearns.

2. **Image Processing:** Preprocessing images by normalizing to fit in a 20x20 pixel box while preserving aspect ratio. Images were centered in a 28x28 pixels.

3. **Train Models:** Train models by Logistic Regression, Neural Network, Support Vector Machine and Random Forest.

4. **Tune Hyper-Parameters:** Validate the classification performance of models on validation sets. Tune hyper-parameters till it gives best performance.

5. **Evaluate on Test Sets:** Test the trained models on both MNIST test set and USPS data.

1.5 Evaluation

1. Evaluate each solution on the test set using classification accuracy, $Acc = \frac{Ncorrect}{N}$, where $Ncorrect$ is the number of corrected classified data samples, and N is the total number of samples of the validation set. Under the 1-of-K coding scheme, each data sample will be assigned a class label as $C = argmaxy_i$, where y $\in R^K$ is the output probability distribution over classes.

2. Construct a confusion matrix for each classifier and observe the relative strengths and weaknesses.

3. Evaluate the performance of the ensemble classifier

2 Data Preprocessing

2.1 Normalization

Normal all images into range from 0 to 1.

2.2 Resize

Resize images to 20x20 pixels.
Add 0 values around images make them 28x28.

2.3 Center of mass

It calculates the center of mass of the values of an array at labels.
Calculate shift x and shift y.

2.4 warpAffine

It will shift the image by multiplying specific transformation matrix, where the shift x and shift y are calculated before.

3 Logistic Regression

3.1 Description

Softmax Regression is a generalization of logistic regression that we can use for multi-class classification. In this task, I have implemented softmax regression with mini-batch gradient descent.

Model:

$a = \texttt{softmax}(z)$
`a` is a `(10 x Mini-Batch Size)` vector.
`z` is a `(10 x Mini-Batch Size)` vector.

$z = w^T x + b$
`w` is a `(n x Mini-Batch Size)` vector.
`x` is a `(n x m)`. n is the number of features. m is number of data.
`b` is just a bias term.

Softmax Function:
$P(y=j|z^i) = \frac{e^{z^{(i)}}}{\Sigma_{j=0}^{k} e^{z_k^{(i)}}}$

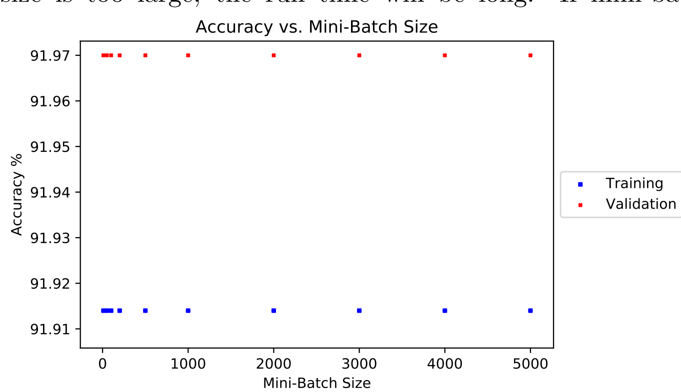
Loss Function:
$E(x) = -\Sigma_{i=1}^{m} \hat{y}_i ln y_i$

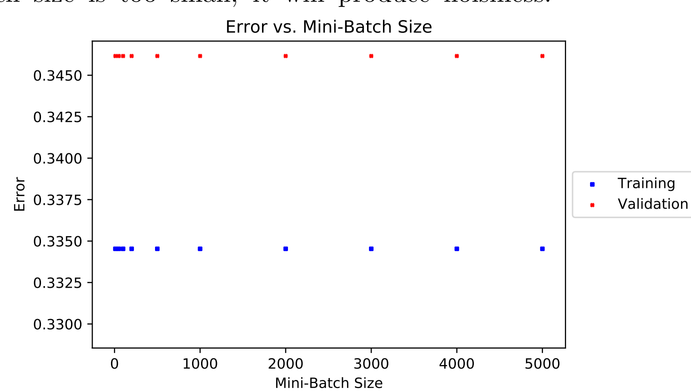
Gradient Descent:
$\frac{\partial J}{\partial w} = (\hat{y} - y)x$
$\frac{\partial J}{\partial b} = (\hat{y} - y)$
$w^{t+1} = w^t - \eta \bigtriangledown E(x)$

3.2 Hyper-parameters

3.2.1 Mini-Batch Size

Mini-Batch size is one of the most important parameters because it affects the run time. If mini-batch size is too large, the run time will be long. If mini-batch size is too small, it will produce noisiness.

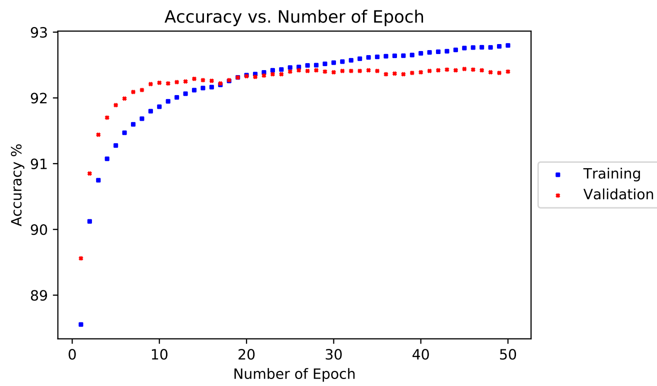


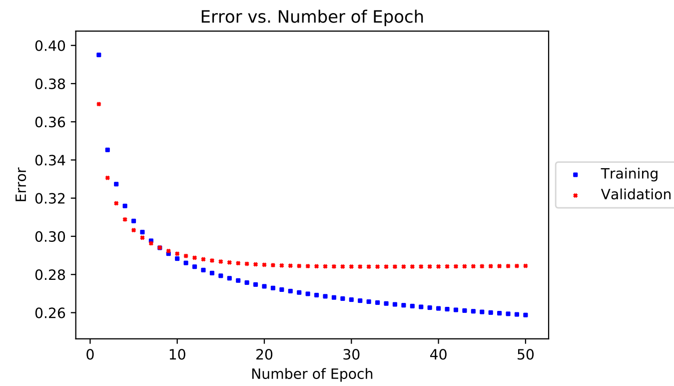


According to graphs above, there is no much difference in min-batch size as long as data has been iterated many times. But due to efficiency, 1000 will be chosen as mini-batch size.

3.2.2 Number of Epoch

In our training, one iteration of all data might be not enough. Hence, I have done several different number of epochs to retrieve the accuracy and error.

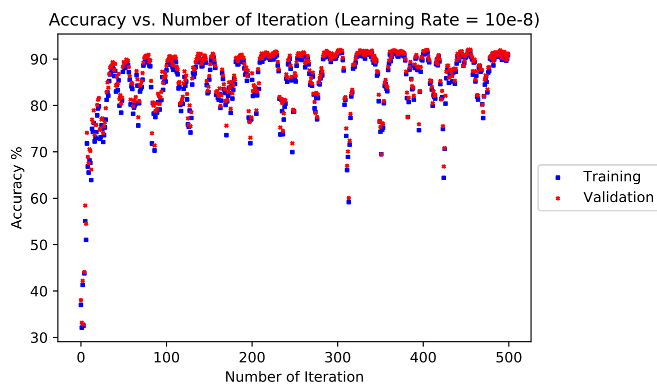


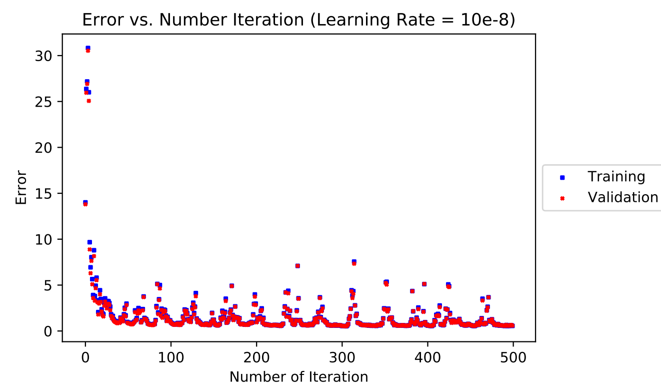


From the above graphs, it is clearly that after 18 epochs, the model starts overfitting. Hence, 20 will be chosen as number of epochs.

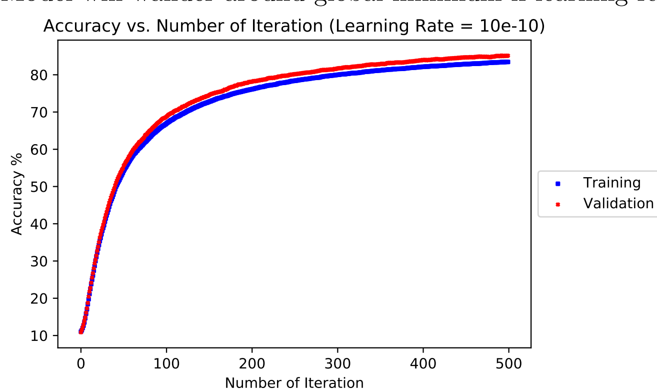
3.2.3 Learning Rate

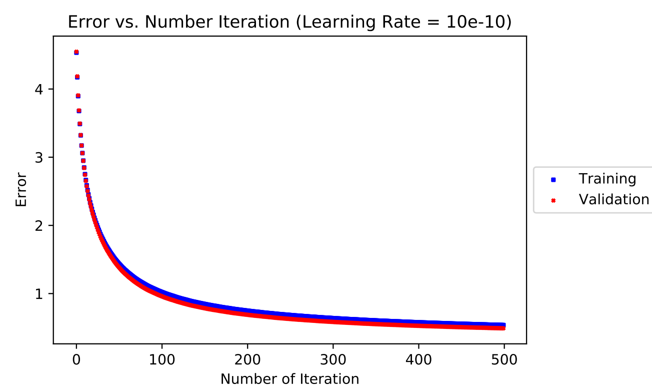
Learning rate is a critical parameter. It will never reach global minimum if it is too large. In contrast, it takes long time to train model if it is too small.



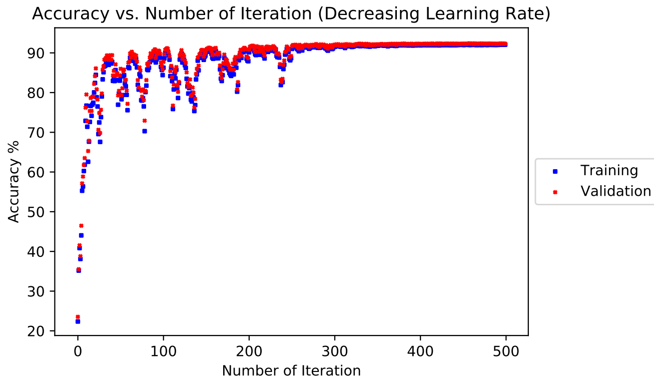


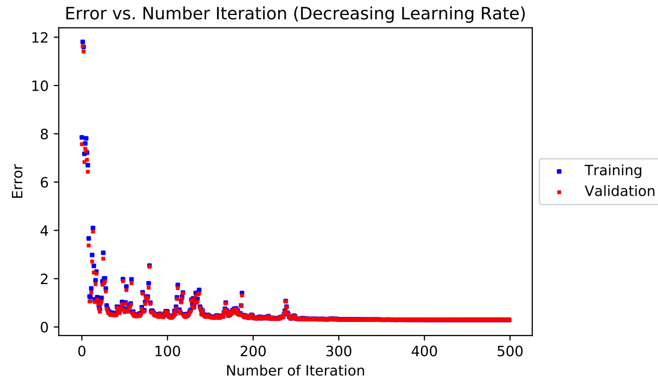
Model will wander around global minimum if learning rate $= 10^{-8}$.





Model will take more epochs to reach the global minimum when the learning rate is too small.





Learning rate = $\frac{1}{\sqrt{10000000 \times epoch^{10} + 20000000}}$ gives the best result. It is converged and reaches the global minimum at the end. Hence, Learning rate = $\frac{1}{\sqrt{10000000 \times epoch^{10} + 20000000}}$ will be chosen.

3.3 Performance Summary

Hyper-Parameters	Optimal Value
Mini-Batch Size	1000
Number of Epochs	20
Learning Rate	$\frac{1}{\sqrt{10000000 \times epoch^{10} + 20000000}}$

Dataset	Error
Training	0.2948
Validation	0.2925
Testing	0.3057
USPS	2.2872

Dataset	Accuracy
Training	91.992 %
Validation	92.22 %
Testing	91.93 %
USPS	32.71 %

`Confusion Matrix for MNIST Test Data`

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	962	0	5	2	0	11	8	2	7	10
	1	0	1111	7	0	2	2	3	9	7	7
	2	1	2	919	19	4	4	6	22	7	1
	3	1	2	19	922	3	40	2	6	25	13
	4	0	0	11	0	907	8	14	5	8	34
	5	2	2	5	26	1	757	16	1	31	8
	6	8	3	11	2	9	16	902	0	8	0
	7	3	2	11	8	5	6	3	949	11	25
	8	3	13	37	23	12	43	4	4	863	7
	9	0	0	7	8	39	5	0	30	7	904

By observing this confusion matrix, the model performs good in MNIST Test Dataset but has plenty of rooms to improve.

`Confusion Matrix for USPS Dataset`

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	512	139	160	73	58	127	187	200	178	24
	1	1	320	31	4	53	12	9	117	24	74
	2	206	257	1211	240	58	12	9	117	24	74
	3	82	237	178	1117	46	201	89	535	288	451
	4	67	117	24	7	705	27	71	27	80	92
	5	279	84	156	411	133	1073	362	118	670	74
	6	46	21	56	5	41	76	537	10	89	10
	7	198	591	61	65	374	82	41	458	90	592
	8	288	207	103	62	324	112	43	301	340	362
	9	321	27	19	16	208	12	24	82	91	195

By observing this confusion matrix, the model performs very bad in USPS Dataset. The model performs the worst in classifying '9'.

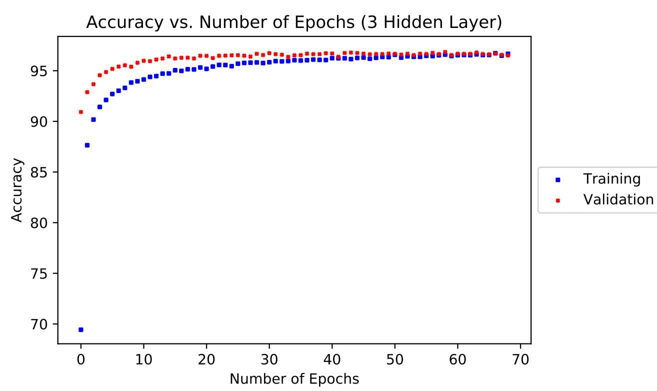
4 Deep Neural Network

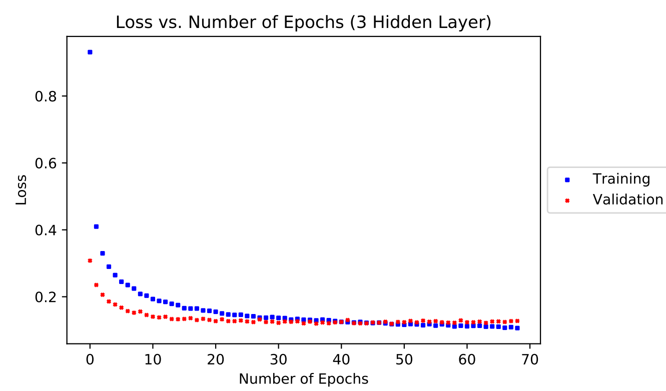
4.1 Description

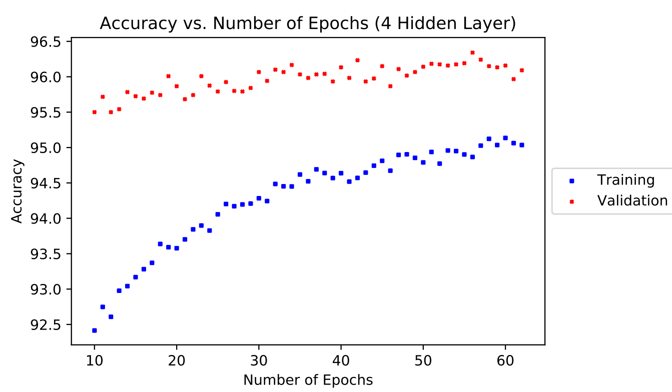
In neural network, it usually has many hidden layers and hidden nodes. First of all, we have to do the forward propagation to calculate the loss. Then, we do backward propagation to calculate the partial derivative of loss with respect to each weights and update them.
In each node, we also apply an activation function, e.g. sigmoid, tanh, relu and so on because the decision boundary might be non-linear. In the past, people usually used sigmoid as the activation function. Nowadays, people tends to use relu, leaky-relu and tanh. relu or leaky relu are more robust and it can shorten the learning time. Because derivate of sigmoid function and tanh is small when the input is large or small, it affect the learning process heavily. relu and leaky-rely can solve the problem.
Even though sigmoids function is not the best choice for the hidden nodes, it is best choice to apply it on output node when the problem is binary classification. If the problem is multi-class classification, softmax will be applied. Hence, in this task for every output nodes, sigmoids will be applied so that it produces a value between 0 and 1, denoting the probability.

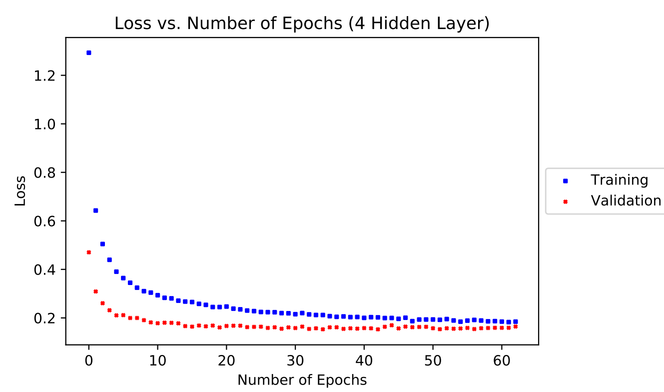
4.2 Hyper-parameters

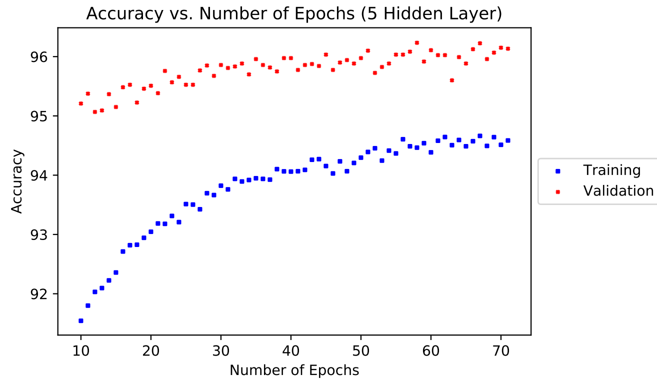
4.2.1 Number of Hidden Layers

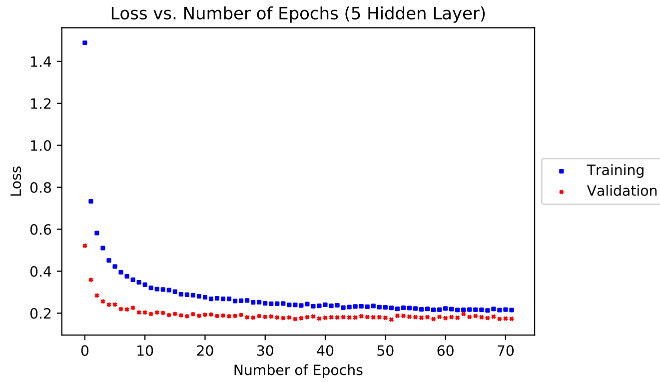






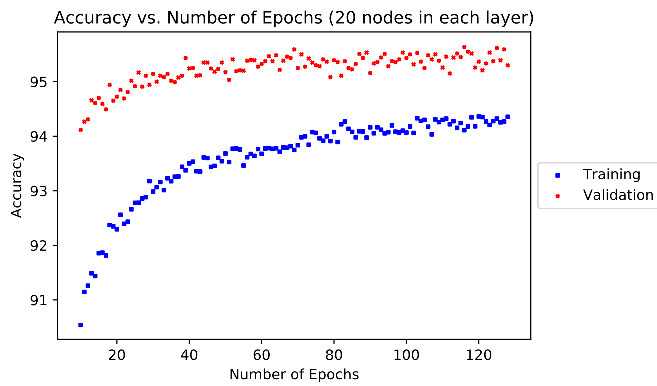


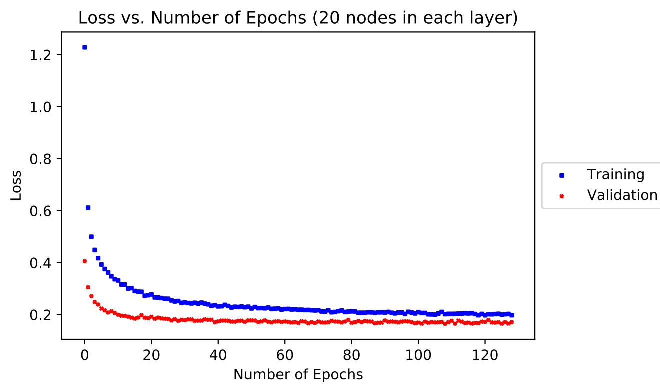


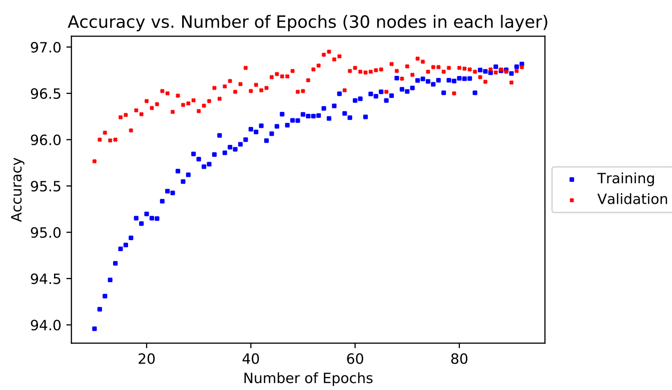


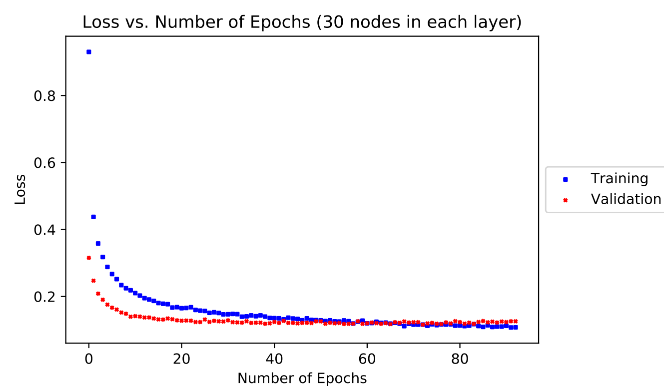
After comparing different number of layers, hidden layers = 5 performs the best. Hence, hidden layers = 5 will be chosen.

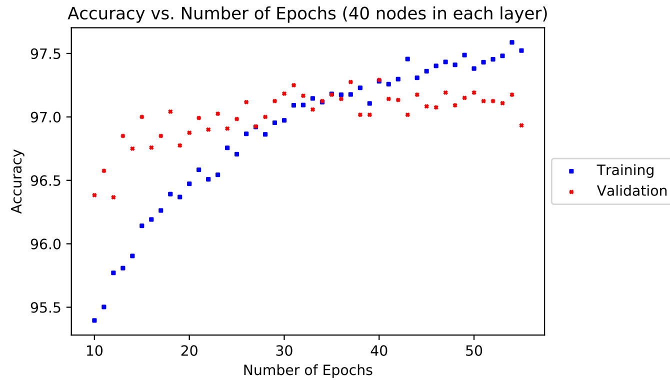
4.2.2 Number of Hidden Nodes

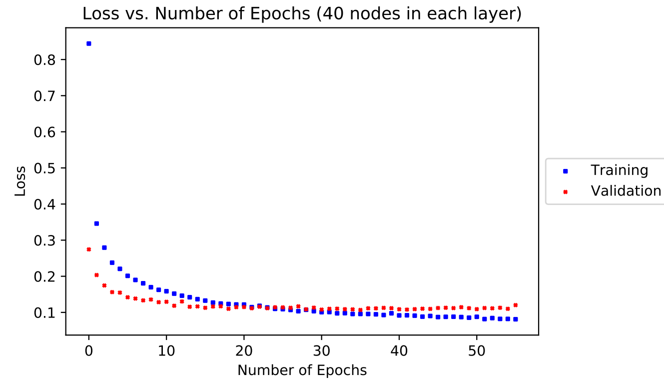








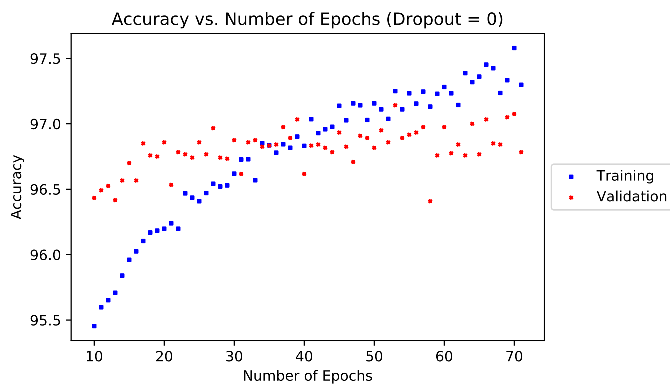


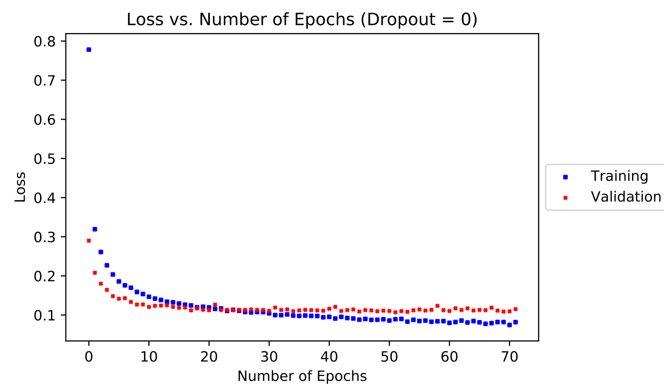


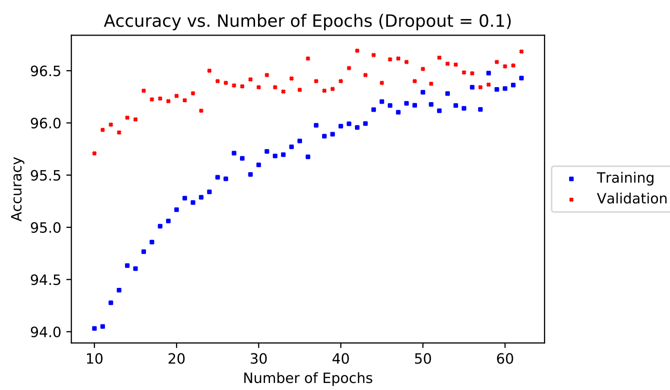
According to above graphs, 40 will be chosen as the number of nodes in each layers. It seems that 30 nodes in each layers gives similar performance, but we can use drop out as a regularization later on. Therefore, number of hidden nodes in each hidden layers will be chosen.

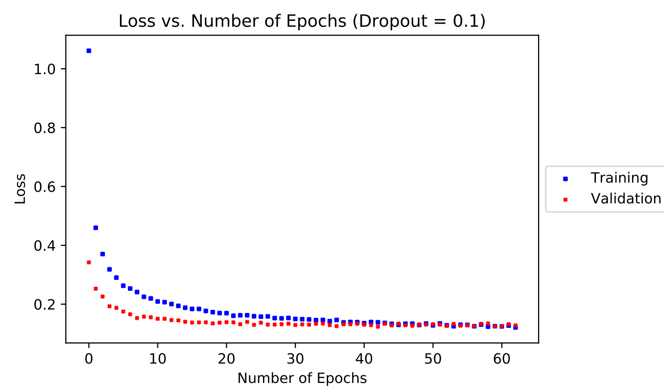
4.2.3 Dropout

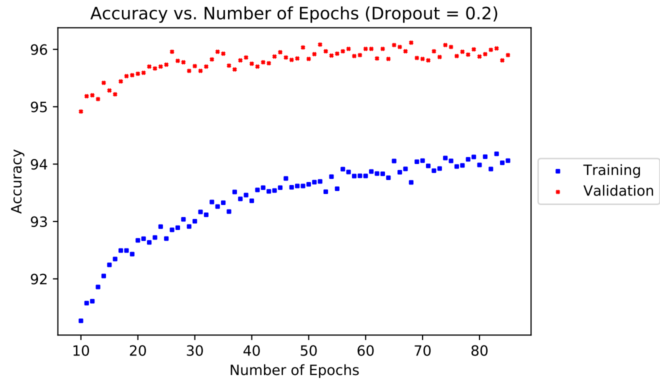
Dropout is to prevent overfitting. It will randomly choose some nodes and ignore them during forward and backward propagation.

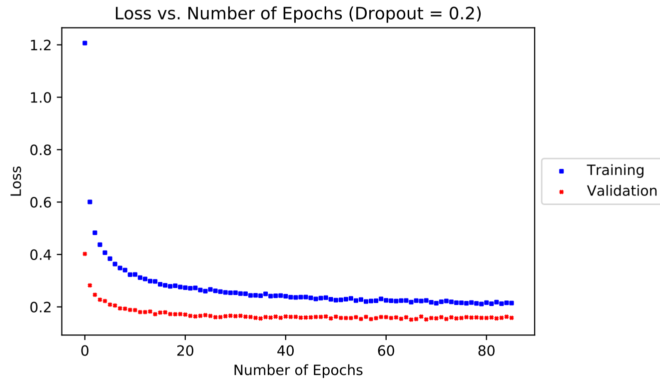








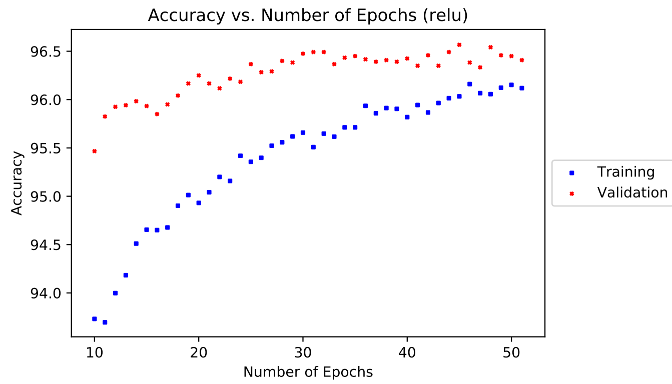


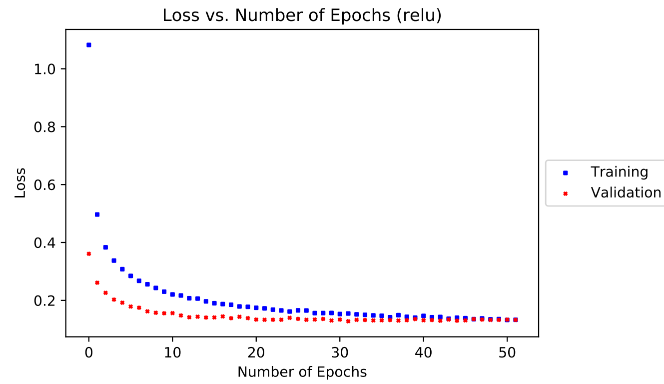


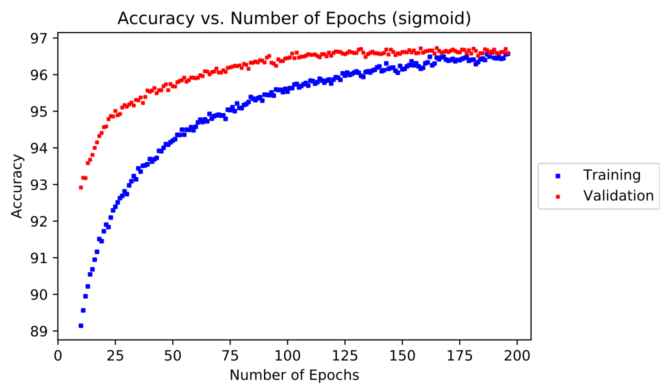
According to above graphs, 0.1 will be chosen as dropout since it gives best performance.

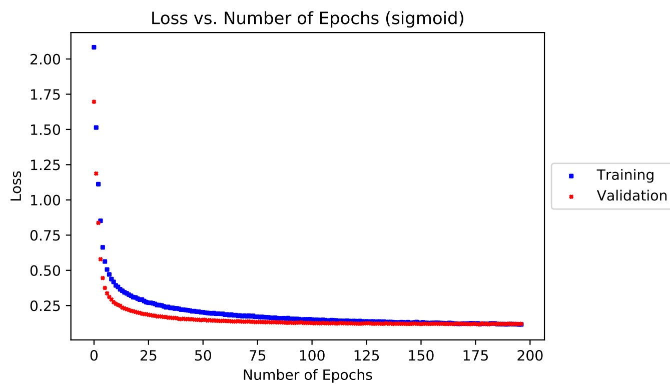
4.2.4 Activation Function

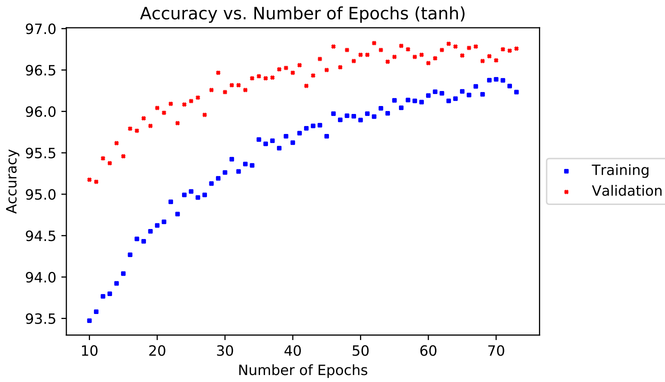
Different activation functions will lead to different performance.

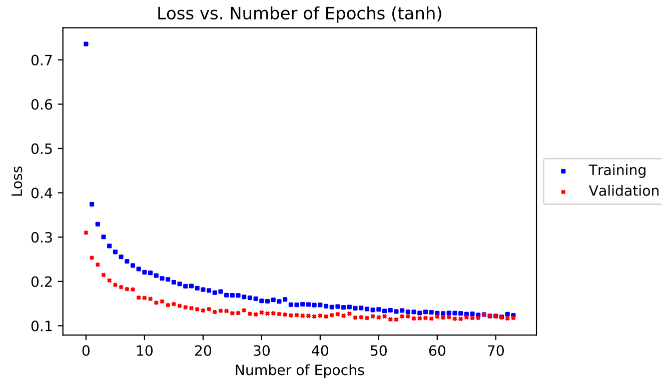












According to above graphs, it is obvious that sigmoid as activation function converges too slow. tanh performs slightly better than relu function. Hence, tanh will be chosen as the activation function for each nodes.

4.3 Performance Summary

Hyper-Parameters	Optimal Value
Drop Out	0.1
Input Nodes	784
First Dense Layer Nodes	40
Activation Function of First Hidden Layer	tanh
Second Dense Layer Nodes	40
Activation Function of Second Hidden Layer	tanh
Second Dense Layer Nodes	40
Activation Function of Third Hidden Layer	tanh
Second Dense Layer Nodes	40
Activation Function of Fourth Hidden Layer	tanh
Ouput Nodes	10
Activation Function of Output Node	Softmax
Number of Epochs	300
Batch Size	128
Early Patient	20

Dataset	Error
Training	0.0881
Validation	0.1089
Testing	0.1032
USPS	1.1548

Dataset	Accuracy
Training	97.42 %
Validation	97.00 %
Testing	97.27 %
USPS	77.13 %

Confusion Matrix for MNIST Test Data

	Actual Values										
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	965	1	3	0	1	3	4	2	1	0
	1	0	1119	1	1	0	3	4	2	5	0
	2	3	1	1013	3	1	0	2	5	4	0
	3	0	1	11	975	0	11	0	7	3	2
	4	1	0	1	0	955	0	7	4	2	12
	5	2	0	0	7	1	869	5	0	5	3
	6	5	2	0	0	8	4	935	0	4	0
	7	2	3	11	4	2	0	0	997	2	7
	8	5	2	6	10	3	5	1	8	931	3
	9	2	2	1	4	16	4	1	8	3	968

By observing the above confusion matrix, the model performs excellent in general. It performs best in classifying '1'.

Confusion Matrix for USPS Dataset

	Actual Values										
Predicted Values		0	1	2	3	4	5	6	7	8	9
	0	1455	6	66	6	224	103	89	30	13	8
	1	1	1720	57	20	60	59	4	23	42	14
	2	28	29	1710	66	38	22	13	52	34	7
	3	11	9	26	1682	4	29	18	107	69	45
	4	14	4	23	43	1859	5	19	9	17	16
	5	86	8	148	7	141	1502	77	2	28	1
	6	9	45	80	20	69	21	1647	0	96	13
	7	33	39	115	115	49	276	53	1273	16	31
	8	33	39	115	115	49	276	53	16	1273	31
	9	10	9	16	23	559	47	3	359	67	907

By observing the above confusion matrix, the model performs good. It does good in classifying '1' and '2'.

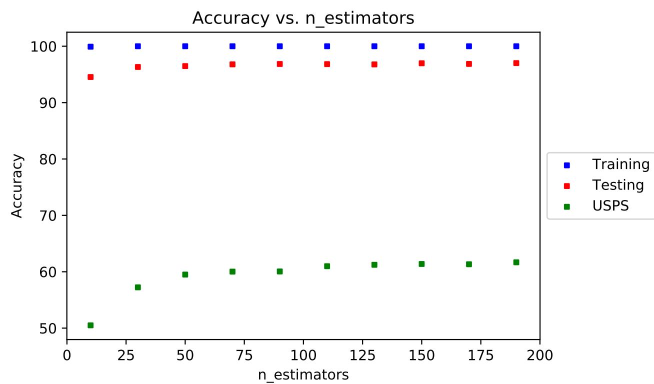
5 Random Forests

5.1 Description

Random forests are bagged decision tree models that split on a subset of features on each split. Essentially, a decision tree splits the data into smaller data groups based on the features of the data until we have a small enough set of data that only has data points under one label. In a decision tree model, these splits are chosen according to a purity measure. That is, at each node, we want information gain to be maximized.

5.2 Hyper-parameters

5.2.1 n_estimators



According to the graph above, based on USPS dataset accuracy, n_estimators = 200 performs the best. Hence, 200 will be the n_estimators.

5.2.2 Performance Summary

Hyper-Parameters	Optimal Value
n_estimators	200

Dataset	Accuracy
Training	100 %
Testing	96.78 %
USPS	62.11 %

Confusion Matrix for MNIST Test Dataset

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	975	0	0	0	0	2	0	1	2	0
	1	0	1122	4	3	0	2	2	0	1	1
	2	6	0	1004	4	2	0	2	9	5	0
	3	1	0	10	974	0	5	0	8	10	2
	4	3	0	2	0	939	0	7	1	3	27
	5	4	0	0	18	2	854	8	1	4	1
	6	8	3	3	0	2	4	935	0	3	0
	7	2	5	19	1	1	0	0	982	4	14
	8	6	0	4	8	4	9	4	4	926	9
	9	7	4	1	13	15	0	2	5	9	953

By observing above confusion matrix, it also performs excellent.

Confusion Matrix for USPS Dataset

	Actual Values										
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	964	5	177	45	68	378	137	86	106	34
	1	114	1253	95	50	135	42	50	138	107	16
	2	100	8	1430	127	33	147	11	111	26	6
	3	22	2	136	1409	25	266	11	105	13	11
	4	70	26	110	17	1453	92	32	165	15	20
	5	165	8	116	106	37	1434	10	90	24	10
	6	174	10	249	55	76	205	1180	35	6	10
	7	117	25	71	12	24	44	4	1695	6	2
	8	26	14	166	157	111	479	50	71	914	12
	9	83	11	186	124	328	145	15	504	34	570

By observing above confusion matrix, it performs average.

6 Support Vector Machine

6.1 Description

Support vector machines attempt to pass a linearly separable hyperplane through a dataset in order to classify the data into two groups. The best hyperplane is the one that maximizes the margin. The margin is the distance between the hyperplane and a few close points. These close points are the support vectors because they control the hyperplane. This is the Maximum Margin Classifier. It maximizes the margin of the hyperplane. This is the best hyperplane because it reduces the generalization error the most. If we add new data, the Maximum Margin Classifier is the best hyperplane to correctly classify the new data. **In this task, we are only required to do three models of SVM mentioned on project description appendix 3.**

6.2 Models

6.2.1 Linear Kernel (other parameters are kept default)

Dataset	Accuracy
Training	94.76 %
Testing	94.44 %
USPS	54.15 %

Confusion Matrix for MNIST Test Dataset

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
	0	965	0	0	2	0	6	4	1	1	1
	1	0	1123	4	2	0	1	1	0	4	0
	2	6	2	970	10	8	3	9	7	16	1
	3	6	2	17	941	0	20	0	8	14	2
Predicted Values	4	1	1	5	0	940	0	7	1	2	25
	5	9	4	5	34	5	807	8	1	17	2
	6	7	2	10	0	5	11	922	1	0	0
	7	2	6	21	5	9	1	0	968	0	16
	8	5	3	9	22	9	34	7	4	880	1
	9	6	6	3	9	30	8	0	21	8	918

By observing above confusion matrix, the model performs good in MNIST Test Dataset.

Confusion Matrix for USPS Dataset

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
	0	868	1	273	40	48	487	170	26	71	16
	1	174	1051	169	102	191	98	45	39	59	71
	2	102	37	1245	142	56	295	37	50	25	10
	3	41	30	175	1186	10	434	8	57	48	11
Predicted Values	4	70	5	155	21	1255	142	132	132	43	45
	5	140	10	184	153	21	1373	21	30	50	18
	6	130	7	423	46	100	261	947	25	34	27
	7	96	53	149	64	87	85	7	1395	45	19
	8	59	45	194	143	85	518	69	49	831	7
	9	86	4	190	189	328	137	8	374	61	623

By observing above confusion matrix, the model performs average in USPS Dataset.

6.2.2 RBF kernel with value of gamma setting to 1 (all other parameters are kept default)

Dataset	Accuracy
Training	100 %
Testing	40.22 %
USPS	10.34 %

Confusion Matrix for MNIST Test Dataset

Actual Values											
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	301	0	0	679	0	0	0	0	0	0
	1	0	1026	0	109	0	0	0	0	0	0
	2	0	0	120	912	0	0	0	0	0	0
	3	0	0	0	1010	0	0	0	0	0	0
	4	0	0	0	736	245	0	0	0	0	1
	5	0	0	0	859	0	33	0	0	0	1
	6	0	0	0	673	0	0	285	0	0	0
	7	0	0	0	503	1	0	0	522	0	2
	8	0	0	0	912	0	0	0	0	62	0
	9	0	0	0	588	1	0	0	1	0	419

By observing above confusion matrix, the model performs bad in MNIST Test Dataset.

Confusion Matrix for USPS Dataset

Actual Values											
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	0	0	0	2000	0	0	0	0	0	0
	1	0	27	0	1973	0	0	0	0	0	0
	2	0	0	3	1996	0	0	0	0	0	0
	3	0	0	0	2000	0	0	0	0	0	0
	4	0	0	0	1999	1	0	0	0	0	0
	5	0	0	0	2000	0	0	0	0	0	0
	6	0	0	0	1999	0	0	1	0	0	0
	7	0	0	0	1960	0	0	0	40	0	0
	8	0	0	0	2000	0	0	0	0	0	0
	9	0	0	0	2000	0	0	0	0	0	0

By observing above confusion matrix, the model performs very bad in USPS Dataset.

6.2.3 RBF kernel with all parameters setting to default values

Dataset	Accuracy
Training Loss	96.78 %
Testing Loss	96.48 %
USPS Loss	60.88 %

Confusion Matrix for MNIST Test Dataset

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	969	0	1	1	0	4	3	1	1	0
	1	0	1125	3	2	0	1	2	0	2	0
	2	4	1	996	5	5	0	3	9	8	4
	3	2	0	8	972	0	7	0	9	8	4
	4	1	1	5	0	947	0	4	2	2	20
	5	6	2	1	18	1	849	6	0	7	2
	6	9	2	2	0	4	7	933	0	1	0
	7	1	11	13	4	4	0	0	978	2	15
	8	3	0	3	8	6	8	5	4	935	2
	9	4	5	2	8	22	2	1	8	5	952

By observing above confusion matrix, the model performs good in MNIST Test Dataset.

Confusion Matrix for USPS Dataset

		Actual Values									
		0	1	2	3	4	5	6	7	8	9
Predicted Values	0	883	2	225	30	71	322	214	24	208	21
	1	126	1161	125	33	198	77	52	42	117	69
	2	83	15	1400	97	47	194	37	78	41	7
	3	34	11	158	1325	16	318	10	66	48	14
	4	46	10	113	5	1508	112	90	53	28	35
	5	151	8	150	109	27	1435	15	37	51	17
	6	130	2	305	31	118	200	1135	14	47	18
	7	111	52	91	20	82	71	7	1545	15	6
	8	41	36	152	122	92	350	90	48	1050	19
	9	101	8	156	98	350	114	16	354	53	750

By observing above confusion matrix, the model performs average in USPS Dataset.

6.3 Final Models for SVM

Based on above results, SVM with root basic function will all parameters setting to default values will be chosen.

7 Ensemble Classifier

Ensemble learning uses multiple machine learning models to try to make better predictions on a dataset. An ensemble model works by training different models on a dataset and having each model make predictions individually. The predictions of these models are then combined in the ensemble model to make a final prediction.

Every model has its strengths and weaknesses. Ensemble models can be beneficial by combining individual models to help hide the weaknesses of an individual model. In this task, I will combine 5 models, which are two different Neural Network, Logistic Regression, SVM and Random Forest. Each of them is performing best among same models and models' hyper-parameters are listed as above.

Accuracy of Ensemble Classifier for MNIST Test Dataset = 97.35 %
Accuracy of Ensemble Classifier For USPS Dataset = 74.99 %

8 Models Comparison

Models	Accuracy in MNIST Test Dataset	Accuracy in USPS Dataset
Logistic Regression	**91.93%**	**32.71%**
Deep Neural Network	**97.27%**	**77.13%**
Random Forest	**96.78%**	**62.11%**
SVM	**94.44%**	**54.15%**
Ensemble Classifier	**97.35%**	**74.99%**

According to above comparisons, Ensemble Classifier performs the best in MNIST Test Dataset whereas Deep Neural Network performs the best in USPS Dataset. The result supports the theorem because Ensemble Classifier performs better in MNIST Dataset than Deep Neural Network.

By observing the confusion matrixes, it is obvious that Deep Neural Network performs the best.

9 Conclusion

The "**No Free Lunch Theorem**" states that there is no one mode that works best for every problem. The assumptions of a great model for one problem may not hold for another problem. My result supports the theorem because Deep Neural Network performs the best in MNIST Test Dataset whereas Ensemble Classifier performs the best in USPS Dataset.

Overall, by observing those confusion matrix, Deep Neural Network performs the best in this project. Each classifier's strengths and weakness have been discussed above.

10 Reference

1. Introduction to Machine Learning by Professor Srihari
https://cedar.buffalo.edu/ srihari/CSE574/index.html

2. Deep Learning Coursera by Andrew Ng
https://www.coursera.org/learn/machine-learning/home/welcome

3. Thoughts on Machine Learning – Dealing with Skewed Classes
https://florianhartl.com/thoughts-on-machine-learning-dealing-with-skewed-classes.html

4. Machine Learning FAQ
https://sebastianraschka.com/faq/docs/closed-form-vs-gd.html

5. Logistic Regression Towards Data Science
https://towardsdatascience.com/logistic-regression-2b555e5f80e6

6. Softmax Regression
https://www.kdnuggets.com/2016/07/softmax-regression-related-logistic-regression.html

7. Tensorflow, MNIST and your own handwritten digits
https://medium.com/@o.kroeger/tensorflow-mnist-and-your-own-handwritten-digits-4d1cd32bbab4