Assignment 1: Computer Vision on MNIST Data Set

Gavyn Gallagher

Northwestern University

MSDS 458: Artificial Intelligence and Deep Learning

Syamala Srinivasan

October 9, 2022

Abstract

Our company X is interested in automating a way to scan our customer paperwork that has been filled in by hand. Upon arriving at the clinic a customer must fill out forms. Part of the questions ask for phone numbers and social security numbers. A person reading and recording each number is a time consuming process. Company X wants a new way of registering the hand written numbers that is faster. Withcomputer vision the process of classifying the handwritten digits can be performed.

To test the accuracy of this new computer vision method the dataset MNIST data set was chosen. After proof of the computer vision working on the data set, the model can be implemented using Company X data. For computer vision ten neural network models were created for 4 experiments in this assignment. The goal of the models was to successfully classify the digits in the data set. The first models' performance was weak. In the second model the number of nodes in the hidden layer increased to 2 and the results improved significantly. In the third experiment six models were created of different numbers of hidden layer nodes, epochs, and activation functions. The best performing model was model8 with 32 hidden layer nodes. In experiment 4 PCA was used on the data set to reduce the number of dimensions to 154, then Model8 was reran and the results minisculely improved. Experiment 5 used a random forest to reduce the number of dimensions to the 70 most important features. Model8 then was reran with the input shape equal to 70 and the models performance dropped. From this assignment, we learned that increasing the number of hidden layer nodes increases the model's performance. Also, reducing the number of dimensions can increase or decrease the model's performance. Images of hand written digits in the MNIST data set can be classified through neural networks.

Introduction and Problem Statement

In this assignment, we will explore computer vision through an image data set. The goal of the assignment is to create neural network models to correctly classify the images. Throughout the duration of the assignment ten models were performed. Models one through eight took on variations of a neural network using tensorflow. Principal component analysis (PCA) decomposition was used in model nine. Lastly a random forecast classifier was used in model ten. Through neural networks, PCA, and random forests we will learn what models perform the best on the MNIST data set.

The Modified National Institute of Standards and Technology (MNIST) data set consists of images of handwritten digits. 60,000 images of the ten digits, zero through nine, make up the data set. There are numerous variations of how each digit is written. Depending on how a person writes, a 4 could appear similar to a 9. Likewise other handwritten digits can be misclassified. Each image is 28 X 28 pixels or 784 total pixels. Using Neural networks the digits in the MNIST will be attempted to be classified correctly.

knowledge graph experiments that were conducted on the class corpus from professor Srinivasan's summer 2022 class. Documents in the corpus feature movie reviews. Another experiment will be classifying the data in the corpus based on genre and sentiment. The goal of the assignment is to gain a greater understanding of things in the corpus that are related to each other.

In this assignment, ten models were constructed over the course of four experiments. All models will be evaluated for accuracy and root mean square error scores. In experiment one, a model called “model” was created using a dense neural network of 784 input nodes, a hidden layer of one node, and 10 output nodes was used. Epochs were set to 200 and the activation layer of the hidden layer was softmax. In experiment 2, “model2” was built the same as “model” except the hidden layer had 2 nodes instead of 1. Experiment 1 and 2 will be compared to see which model outperformed the other.

Experiment 3 had a total of 6 models, models 3-8, built and evaluated. With each model in experiment 3 being a slight variation to the original “model”. Model3 is differentiated by its activation layer being sigmoid instead of softmax. Model4 and Model5 increased the number of hidden layer nodes to 4 and 8. Model6 and Model7 increased the number of hidden layer nodes to 4 and 8 as well as increased the epoch size to 500. Lastly, in experiment 3 Model8 increased the number of hidden layer nodes to 32.

Moving into experiment 4, Determined by accuracy and loss scores the best performing model will undergo PCA. The number of features on the training and testing dataset will be reduced from 784 to 154 pixels. Then the best performing model will once again be tested using 154 as the value of its input shape size in the input layer. The goal is to see if PCA had an effect on the model.

Finishing with experiment 5, random forest is used to reduce the dimensionality of the training dataset. The random forest is seeking to find the most important 70 pixels on the images. Then the best model was run with an input shape size of 70 in the input layer.The goal is to see whether the random forest affected the best model scores. After reading this assignment we will be more knowledgeable on the MNIST dataset and how to use neural networks to classify the images.

Literature Review

Comparison of non-linear activation functions for deep neural networks on MNIST classification task

Dabal Pedamonti evaluates the different activation functions of a neural network on the MNIST data set. Various activation functions such as sigmoid, ReLu, Leaky ReLu and Elu were defined and compared. MNIST data set offers a dataset perfect for testing computer vision.

First the authors focused on defining each activation function and what purpose they served. Sigmoid was a popular activation function until ReLu was invented. ReLu solves the gradient vanishing and exploding problem thus increasing its popularity over sigmoid (Pedamonti, 2018) . Next experiments using the MNIST dataset were conducted. Elu has the highest accuracy of the activation functions. It is interesting that the ReLu variants all performed higher than sigmoid.

Through this article several activation functions were defined and experimented on. Understanding of the reason why each was created was learned. Performance of the functions on the MNIST were also a key point. Knowing which activation function to use is an important decision to make in a neural network.

CNN is Preferred over Other Methods for Handwritten Digit Recognition System

This article, by Akriti Singh, also explores classifying the handwritten digits of the MNIST data set. Stated here is the convolution neural networks (CNNS) are preferred over methods for the classifying the MNIST data set. The article explores the positives and negatives of CNNs.

A main point of the article is how CNN compares to other methods. Other methods that were outperformed by CNN include linear regression, support vector model, and K nearest neighbor (Singh, 2020). Accuracy and loss metrics were both better for CNN in this experiment. The authors believe this for the training data obtains more features through the CNN layers. A breakdown of how each layer in a CNN operates occurs. A negative of CNN is how to understand how the method works and what is truly accomplished at each epoch (Singh, 2020). The results of using a neural network outweigh the confusions.

In conclusion, CNNs were used to identify the MNIST digits. CNNs proved to be an excellent method at successful classification. There is a learning curve to understanding CNNs and building an accurate model may take some time. Overall CNNs are a powerful tool at computer vision.

Method(s)

Data preparation was performed on the MNIST data set. One hot encoding was used to get each digit to appear as a binary value. To_careogtrical is a function on python that can convert the digits to one hot encoding. All the images are 28 by 28 pixels. The training and testing data sets have 2 dimensions as their shapes are (60000, 28, 28) and (10000, 28, 28). The training and testing data sets were reshaped in 1 dimensional arrays. Now the training data set had a shape of (60000, 784) and the testing data set had a shape of (10000, 784). Next the training and testing set were divided by 255 to rescale the images.

Models made in this assignment will be made through TensorFlow neural networks. TensorFlow is a library for machine learning that can be accessed via python. According to Mo Daoud, neural networks work by having multiple layers or neurons: an input, hidden, and output layer (Daoud, 2020). Each layer has an activation function where information is either kept or withheld to the next layer. There are several types of activation functions such as ReLu, softmax, sigmoid, and tanh. Nine of the ten models created in this assignment will use Relu in the input layer and softmax in the hidden layer. One model will use sigmoid. In essence, a layer analyzes the output of the previous layer and passes on important information. Each time the data is sent from the input layer to the output layer and then back to the input layer for another cycle it is called an "epoch" (Daoud, 2020). During the experiments in this assignment the amount of epochs will be changed to test if a higher number of epochs improves the scores.

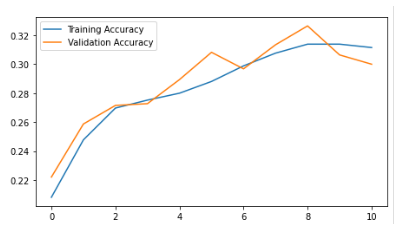
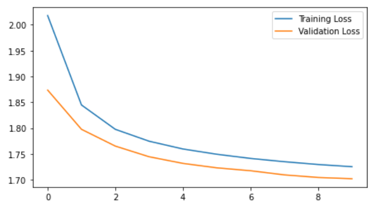
Dimensional reduction was performed with PCA and random forest in experiments 4 & 5. The article, *Seven Techniques for Data Dimensionality Reduction*, details how PCA and random forest can be used to reduce the number of dimensions (Widmann & Silipo, 2015). PCA reduces the amount of variables while maintaining most of the information (Widmann & Silipo, 2015). With the random forest technique, the 70 most important features of pixels were discovered. The training and test data sets would then be transformed to have shapes of (60000, 70) and (10000, 70). The best model from experiments 1-3 would be chosen to be rerun with an input shape of 70. With PCA and random forests the dimensionality of the images can be reduced for hopefully better results.

Results

Experiment 1

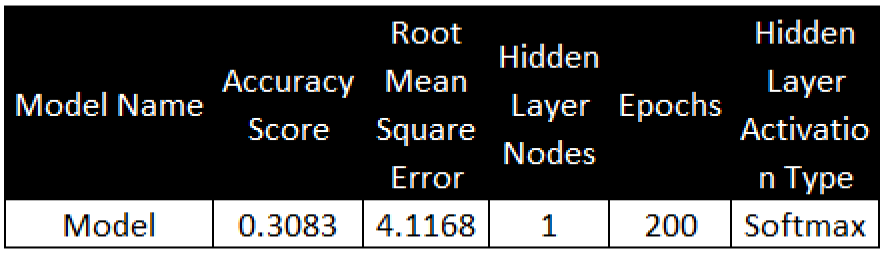
One model was created for experiment 1. This model called "model" was created using a dense neural network of 784 input nodes, a hidden layer of one node, and 10 output nodes. Epochs were set to 200 and the activation layer of the hidden layer was softmax. In Figure 1, the accuracy and loss scores over each iteration of the neural network is shown. The model stopped after 10 epochs or iterations. The graphs show an increase in accuracy and a decrease in loss after each iteration.

Figure 1
Model Accuracy and Loss Graphs

Model Name	Accuracy	Loss
Model		

Scores of model were generated for accuracy and root mean square error. Figure 2 displays these values for model. The accuracy is low at only .3083. Combined with the high root mean square error of 4.1168.

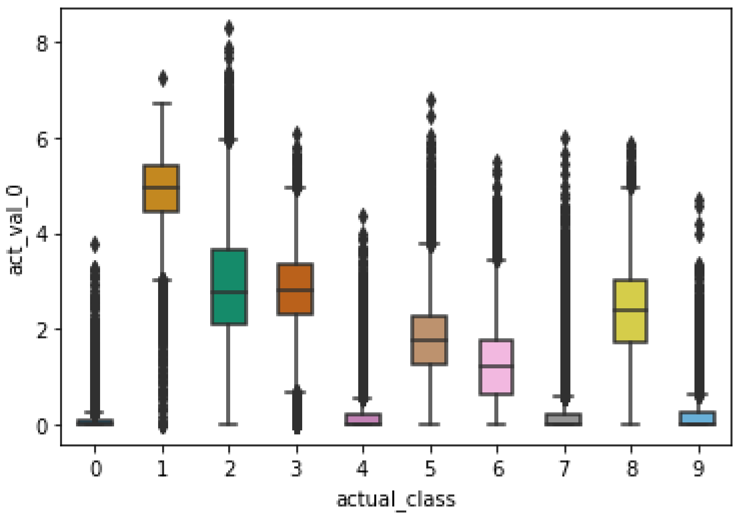
Figure 2
Models' Scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model	0.3083	4.1168	1	200	Softmax

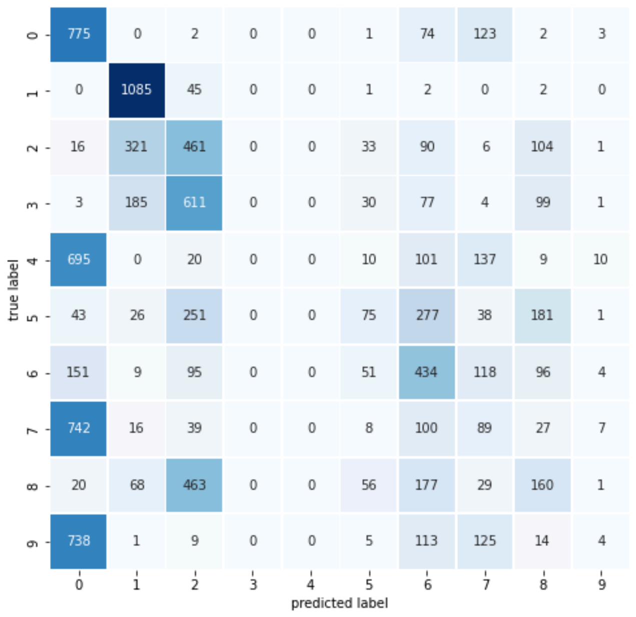
The hidden node activation values are compared against the real class values in the boxplot in figure 3. Here we can see numerous classes have outliers, meaning that the activation layer is not detecting the digits properly. This is not a successful model at classifying the digits.

Figure 3
Models' Boxplot of the correlation of hidden node activation values and class labels



A confusion matrix of model can be seen in figure 4. Theoretically a perfect confusion matrix would show a downward diagonal of dark blue squares. This is not the case here as many of the digits got misclassified with others. For example, 208 eights were mistaken for fives. None of the ten digits had near-perfect or perfect classification.

Figure 4
Models' Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

Experiment 2

Next in experiment 2, a model was created similar to the first model. This model called "model2" had only the number of nodes on the hidden layer increased from 1 to 2. All other architecture and parameters of model2 are matching model. Figure 5, displays the accuracy and loss graphs of model2. The Y-axis on the accuracy graph is much higher than it was on model.

Figure 5
Model2 Accuracy and Loss Graphs

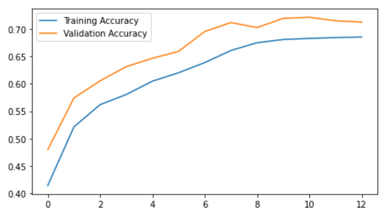
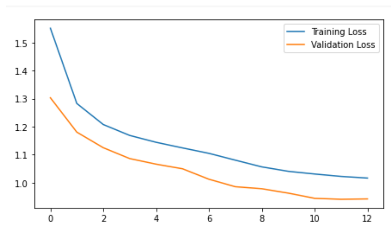
Model Name	Accuracy	Loss
Model2		

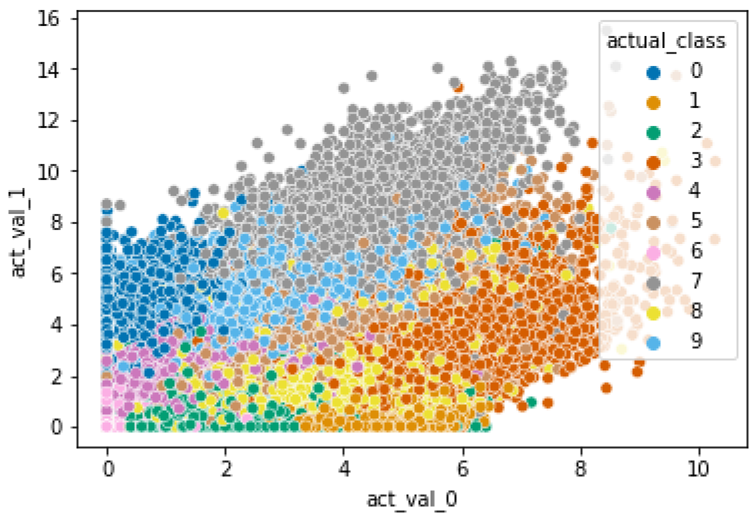
Figure 6 confirms this to be true, as model2 has an accuracy score of 0.6495. Which is over double the accuracy was of model. This means that model2 is more accurate at predicting the digits than model. The loss scores is in a similar vain as there was much improvement over model.

Figure 6
Model2s' Test Scores

Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model2	0.6495	2.3343	2	200	Softmax

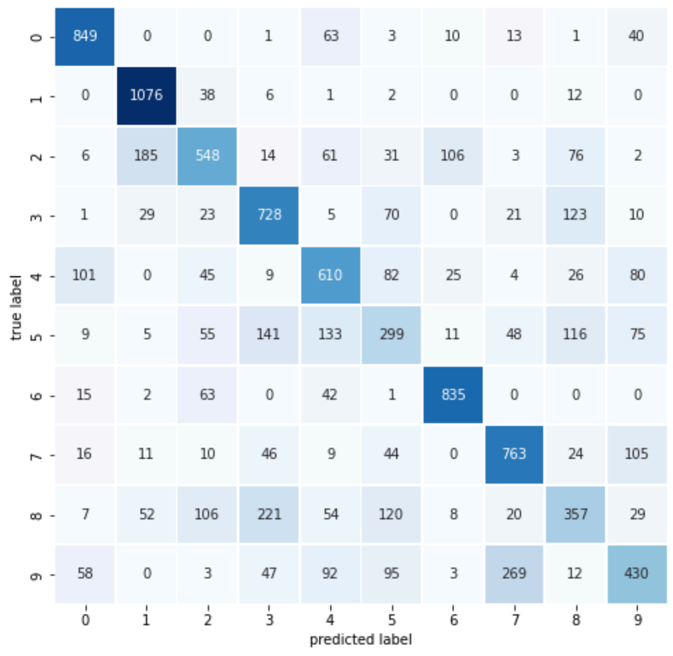
To look into the hidden node activation values of model2 a scatter plot was created. This scatterplot can be seen in figure 7. Classes are denoted by color in the plot. Theoretically we should see clear borders between each **THIS PARAGRAPH NEEDS WORK**

Figure 7
Model2s' Scatter Plot of the correlation of hidden node activation values and class labels



Lastly for experiment 2, a confusion matrix was created. Compared to models' confusion matrix There are less squares in the matrix showing blue color, and those that are blue are of a darker hue. Meaning that model2 outperformed model at classifying the digits. There still are some issues here as 1 & 2, 4&5, and 3 & 8 are getting confused by the model.

Figure 8
Model2s' Confusion Matrix

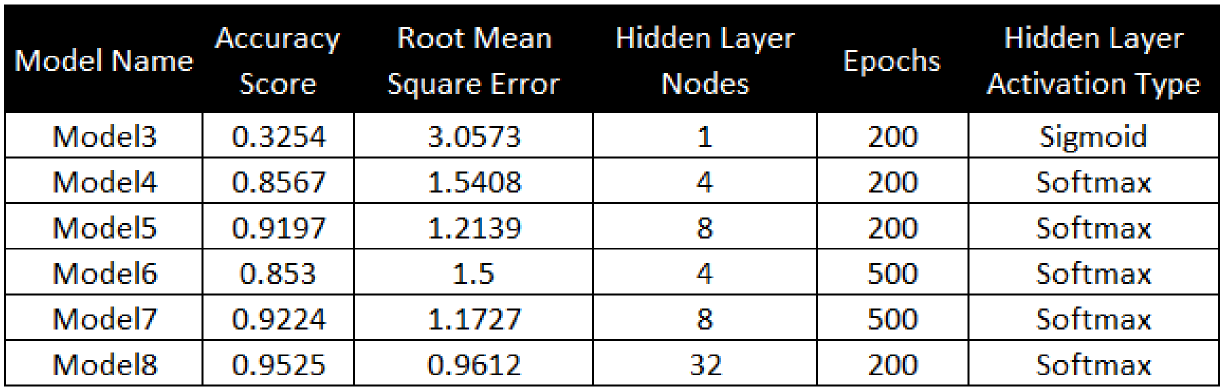


true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	849	0	0	1	63	3	10	13	1	40
1	0	1076	38	6	1	2	0	0	12	0
2	6	185	548	14	61	31	106	3	76	2
3	1	29	23	728	5	70	0	21	123	10
4	101	0	45	9	610	82	25	4	26	80
5	9	5	55	141	133	299	11	48	116	75
6	15	2	63	0	42	1	835	0	0	0
7	16	11	10	46	9	44	0	763	24	105
8	7	52	106	221	54	120	8	20	357	29
9	58	0	3	47	92	95	3	269	12	430

Experiment 3

Six models were created for experiment 3. Each model was built from the models' archietieruce with a slight alteration. To see how each model is different, reference figure 8. The accuracy and root mean square errors scores are displayed in figure 8 as well. Based on these results it appears that increasing the hidden layer nodes does increase the accuracy score and lower the loss score. Also of note, that an increase in the number of epochs does not necessarily improve the scores. Model6 had a higher epoch than Model4 yet a lower accuracy score. Model3 used sigmoid instead of softmax, and the scores were worse than they were in model. Model8 performed the best of the models created in the first 3 experiments.

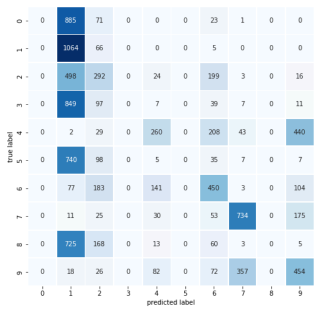
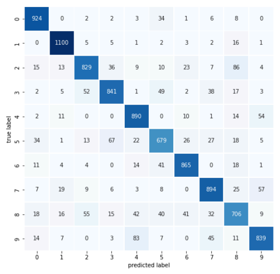
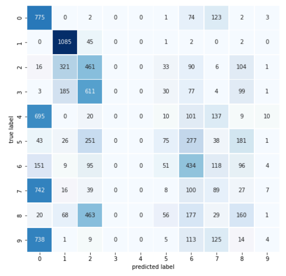
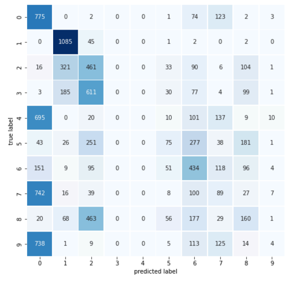
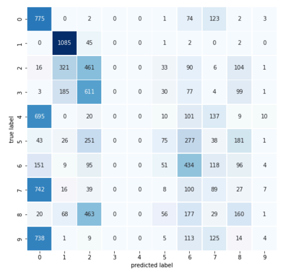
Figure 8
Models 3-8 Test Scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model3	0.3254	3.0573	1	200	Sigmoid
Model4	0.8567	1.5408	4	200	Softmax
Model5	0.9197	1.2139	8	200	Softmax
Model6	0.853	1.5	4	500	Softmax
Model7	0.9224	1.1727	8	500	Softmax
Model8	0.9525	0.9612	32	200	Softmax

Confusion matrices are shown in figure 9. Similar to the accuracy and loss scores the models are easy to distinguish which ones performed better than the others. Model8 shows a clear downward diagonal dark blue line.

Figure 9
3-8 Confusion Matrices

Model Name	*Confusion Matrix*
Model3	
Model4	
Model5	
Model6	
Model7	

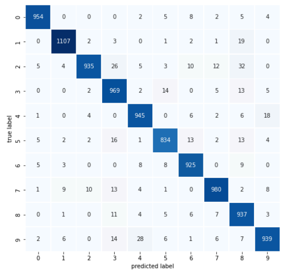
	0	1	2	3	4	5	6	7	8	9
0	0	885	71	0	0	0	23	1	0	0
1	0	1064	66	0	0	0	5	0	0	0
2	0	498	292	0	24	0	199	3	0	16
3	0	849	97	0	7	0	39	7	0	11
4	0	2	29	0	260	0	208	43	0	440
5	0	740	98	0	5	0	35	7	0	7
6	0	77	183	0	141	0	450	3	0	104
7	0	11	25	0	30	0	53	734	0	175
8	0	725	168	0	13	0	60	3	0	5
9	0	18	26	0	82	0	72	357	0	454

	0	1	2	3	4	5	6	7	8	9
0	924	0	2	2	3	34	1	6	8	0
1	0	1100	5	5	1	2	3	2	16	1
2	15	13	829	36	9	10	23	7	86	4
3	2	5	52	841	1	49	2	38	17	3
4	2	11	0	0	890	0	10	1	14	54
5	34	1	13	67	22	679	26	27	18	5
6	11	4	4	0	14	41	865	0	18	1
7	7	19	9	6	3	8	0	894	25	57
8	18	16	55	15	42	40	41	32	706	9
9	14	7	0	3	83	7	0	45	11	839

	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

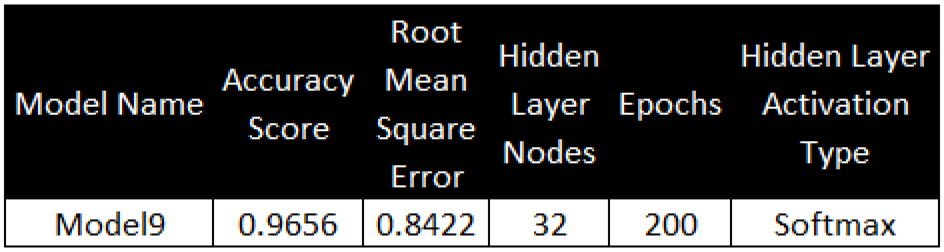
	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

Model8	

Experiment 4

In experiment 4, the best model previously created was to be run again. This time with an input shape size equal to 154. PCA reduced the dimensionality of the pixels of the images from 784 to 154. Model8 was the highest performing model and was chosen for this experiment. Results of this new model called "model9" are shown in figure 10. The accuracy and loss scores are slight improvements over model8. Meaning that PCA worked and that the reduced dimensions did have a positive impact.

Figure 10
Model9s' Test Scores

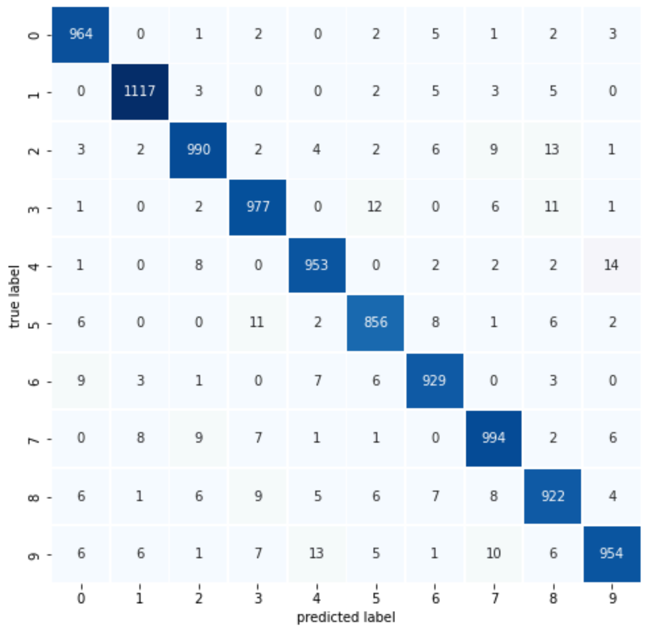


Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model9	0.9656	0.8422	32	200	Softmax

The confusion matrix for model9, shown in figure 11, is near perfect. Almost all of the digits in the test data are predicted correctly. There are still a few misclassified, any more improvement on this model may result in overfitting. Overall PCA was a success.

Figure 11

Model9s' Confusion Matrix

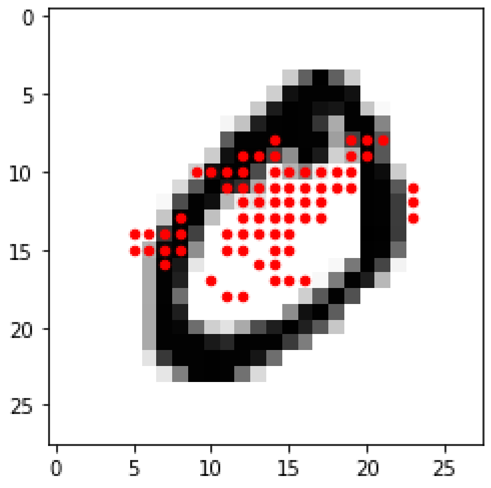


true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	964	0	1	2	0	2	5	1	2	3
1	0	1117	3	0	0	2	5	3	5	0
2	3	2	990	2	4	2	6	9	13	1
3	1	0	2	977	0	12	0	6	11	1
4	1	0	8	0	953	0	2	2	2	14
5	6	0	0	11	2	856	8	1	6	2
6	9	3	1	0	7	6	929	0	3	0
7	0	8	9	7	1	1	0	994	2	6
8	6	1	6	9	5	6	7	8	922	4
9	6	6	1	7	13	5	1	10	6	954

Experiment 5

A random forest was used in experiment 5 to reduce the dimensionality of the pixels to 70. Similar to experiment 4, the best model of models 1-8 was chosen for this experiment. Model8 being the best model was selected. Then the input shape in the input layer was set to 70. Figure 12 shows the 70 most important pixels in an image. This was surprising as the pixels are in the center and towards the left.

Figure 12
Model10's 70 most important pixels



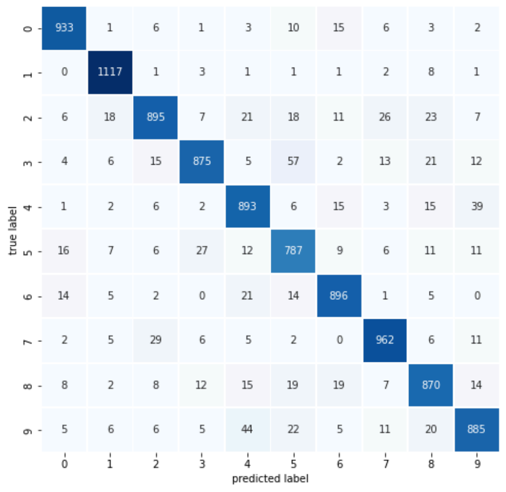
Scores of the new model named “model10” are displayed in figure 13. The accuracy score of 0.8566 is lower than model8s’ accuracy score. In turn, the PCA model9 also performed greater than model10. The random forest decreased the performance of model8. By reducing the dimensions to 70 the ability to classify the digits diminished.

Figure 13
Model10's Test Scores

Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model10	0.853	1.5	32	200	Softmax

The confusion matrix for model10 is shown in figure 14. There are several light blue squares. Digits 4 and 9 face the greatest confusion. This is still a good to very good confusion matrix. Model10 did worse than model8 and model9.

Figure 14
Model10's Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	933	1	6	1	3	10	15	6	3	2
1	0	1117	1	3	1	1	1	2	8	1
2	6	18	895	7	21	18	11	26	23	7
3	4	6	15	875	5	57	2	13	21	12
4	1	2	6	2	893	6	15	3	15	39
5	16	7	6	27	12	787	9	6	11	11
6	14	5	2	0	21	14	896	1	5	0
7	2	5	29	6	5	2	0	962	6	11
8	8	2	8	12	15	19	19	7	870	14
9	5	6	6	5	44	22	5	11	20	885

Conclusion

In conclusion, this assignment was to use computer vision to classify handwritten digits. Once the accuracy is adequate the neural network model can be used to classify the handwritten digits of Company Xs' forms. The neural network would allow for a quicker way of recording the information off of clients forms.

How the models performed, Model8 performed the best of the models in experiments 1-3. Model8 used to differentiate from the original model by having the number of nodes in the hidden layer increase to 32. There was a pattern between increasing the number of nodes in the hidden layer and increasing the performance of the models. Model2 outperformed the model greatly with 1 more node in the hidden layer. Altering epoch size and changing the activation layer type had little impact on the scores.

PCA and random forest were used to decrease the dimensions of pixels of the images of the data set. PCA improved upon model8 with a higher accuracy and lower loss score. While random forest reduced the amount of pixels from 784 to 70. Although the model still performed well it did not perform as well as model8 or the PCA model called model9. Overall, the neural network models created in this assignment did excellent at classifying the digits 0 through 9 in the MNIST data set.

Overall recommendations for the project is to keep testing with new models. Company X should wait a bit longer to achieve a greater model performance before implementing the model. The models 1-8 all included a few neurons, with the maximum being 32. A larger number of neurons should be tested and then compared to the results of previous experiments. Also the T-SNE plot would be a useful addition into visualizing the predictions. Company X will eventually have a computer vision neural network model that can automate the process of reading handwritten digits.

References

Singh, A. (2020). CNN is Preferred over Other Methods for Handwritten Digit Recognition System. *International Journal of Science and Research (IJSR)*, *9*(5).

Daoud, M. (2020, May 31). *Neurons, Activation Functions, Back-Propagation, Epoch, Gradient Descent: What are these?* Medium. https://towardsdatascience.com/neurons-activation-functions-back-propagation-epoch-gradient-descent-what-are-these-c80349c6c452

Pedamonti, D. (2018). Comparison of non-linear activation functions for deep neural networks on MNIST classification task. arXiv preprint arXiv:1804.02763.

Widmann, M., & Silipo, R. (2015, May 12). *Seven Techniques for Data Dimensionality Reduction*. KNIME. https://www.knime.com/blog/seven-techniques-for-data-dimensionality-reduction

Appendix

Figure 1
Model Accuracy

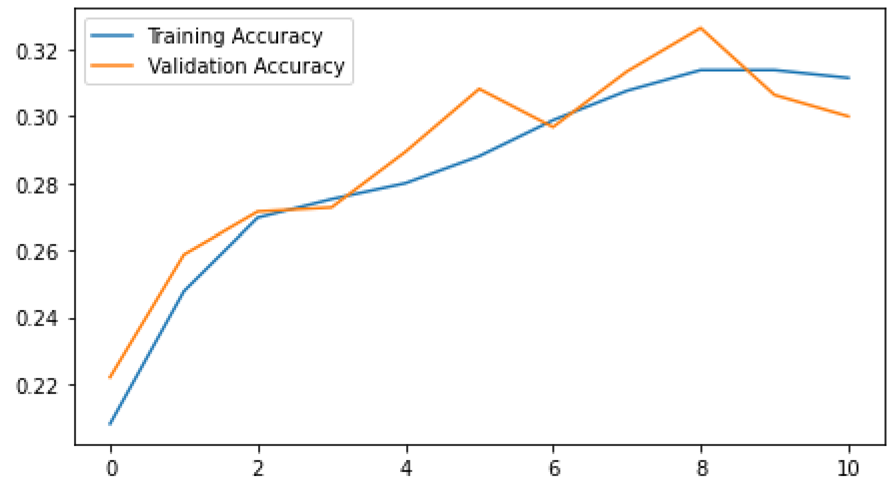


Figure 2
Model Loss

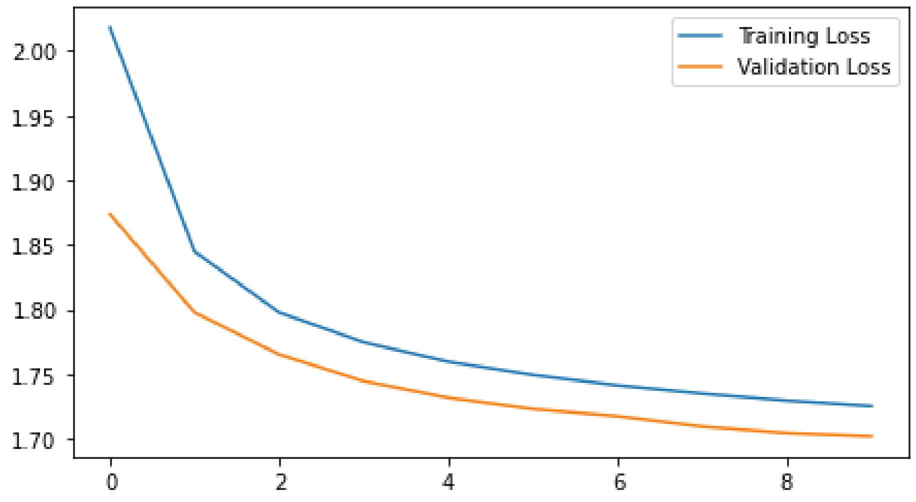
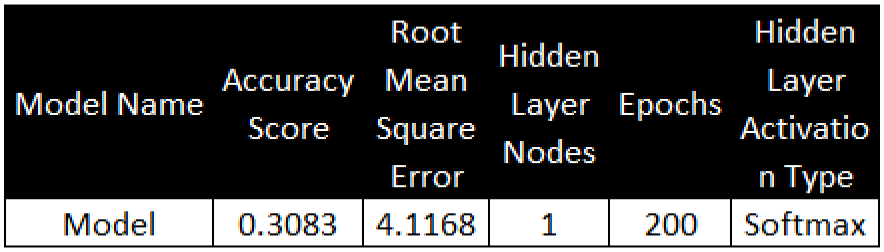


Figure 3
Models' Scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model	0.3083	4.1168	1	200	Softmax

Figure 4

Models' Boxplot of the correlation of hidden node activation values and class labels

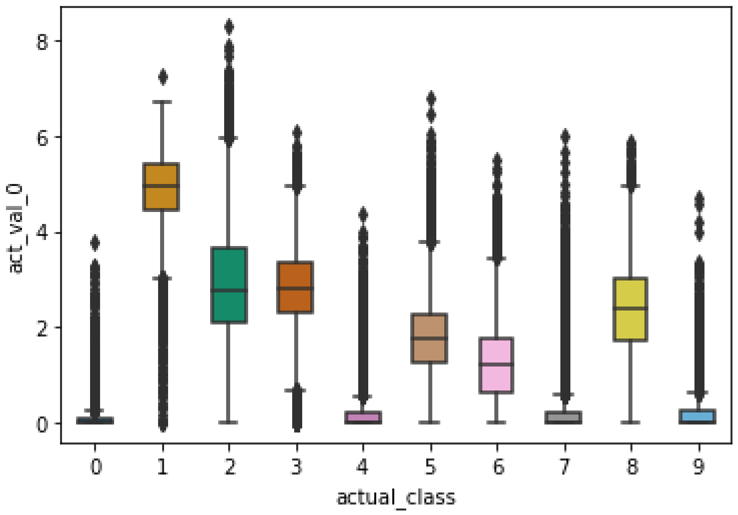
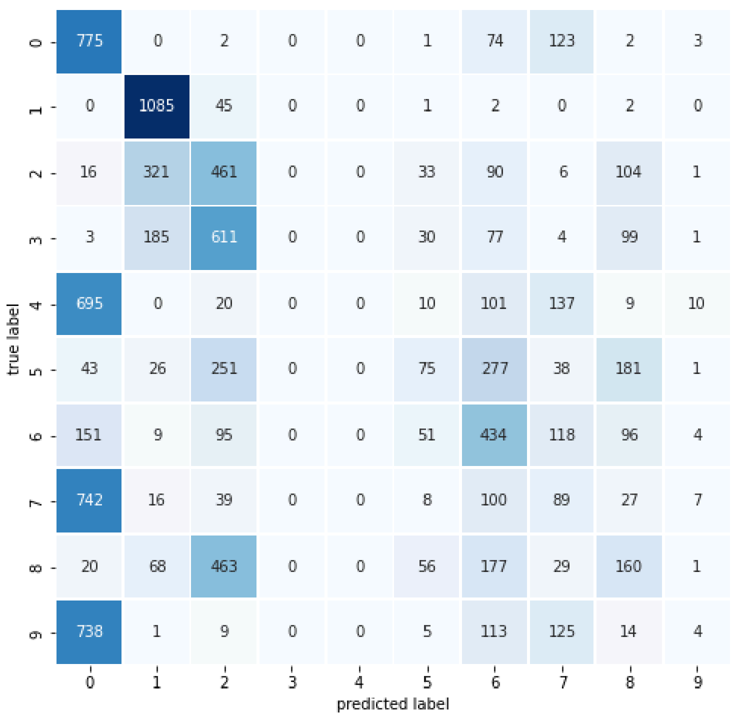


Figure 5

Models' Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

Figure 6
Model2 Accuracy

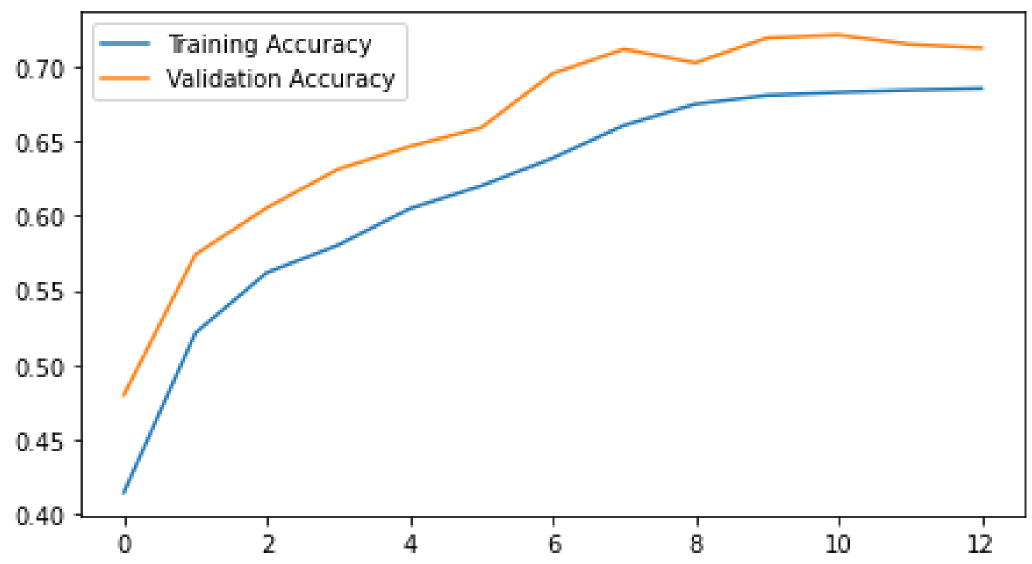


Figure 7
Model2 Loss

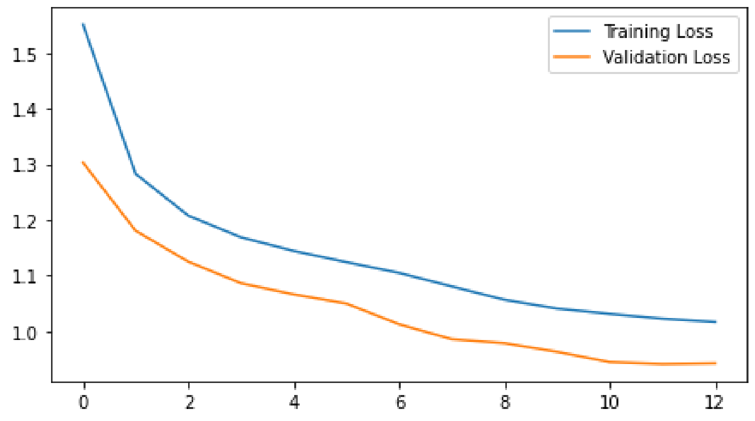
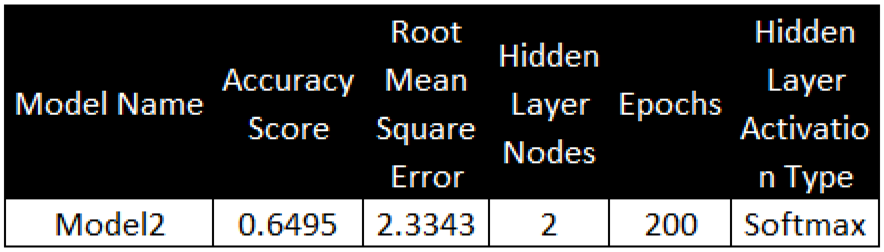


Figure 8
Model2s' Scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model2	0.6495	2.3343	2	200	Softmax

Figure 9

Model2s' Scatter Plot of the correlation of hidden node activation values and class labels

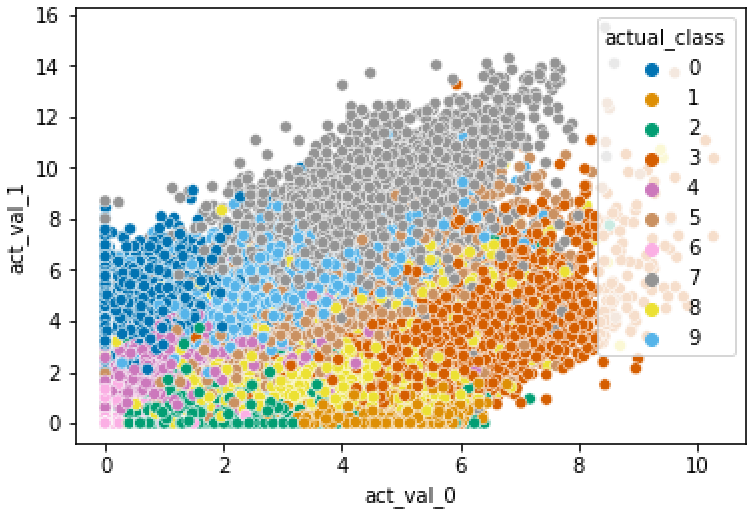
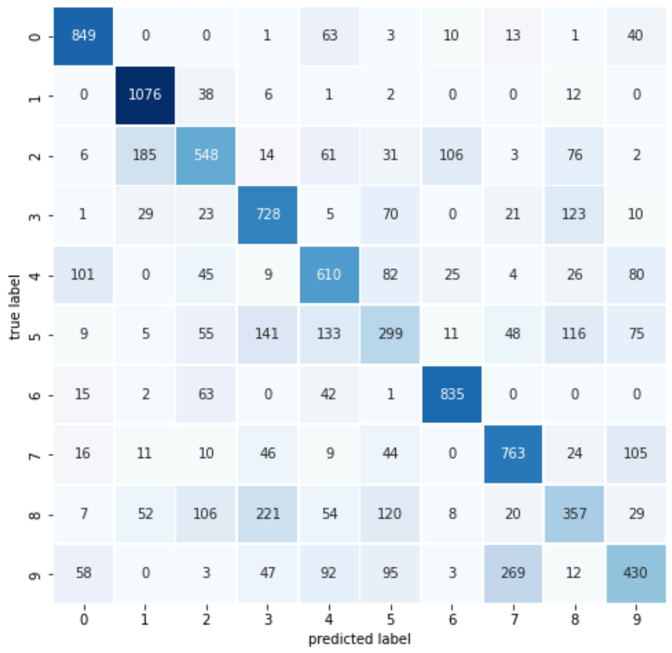


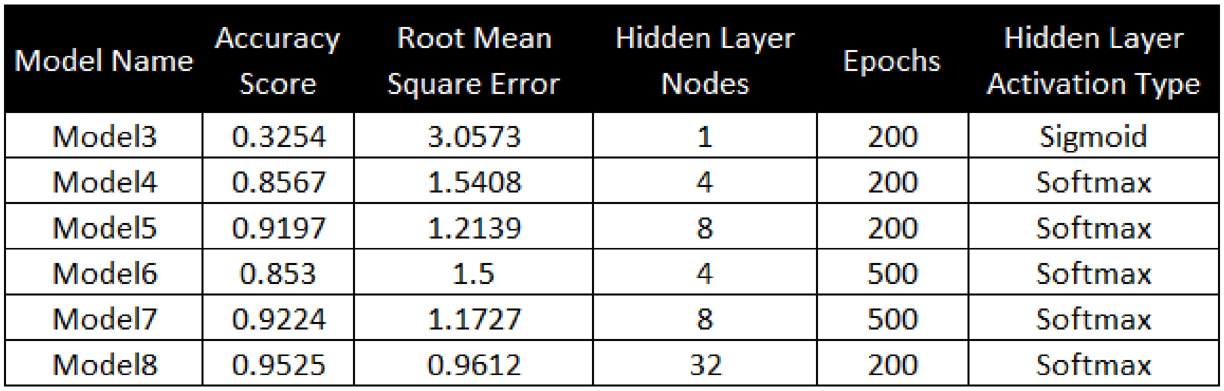
Figure 10

Model2s' Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	849	0	0	1	63	3	10	13	1	40
1	0	1076	38	6	1	2	0	0	12	0
2	6	185	548	14	61	31	106	3	76	2
3	1	29	23	728	5	70	0	21	123	10
4	101	0	45	9	610	82	25	4	26	80
5	9	5	55	141	133	299	11	48	116	75
6	15	2	63	0	42	1	835	0	0	0
7	16	11	10	46	9	44	0	763	24	105
8	7	52	106	221	54	120	8	20	357	29
9	58	0	3	47	92	95	3	269	12	430

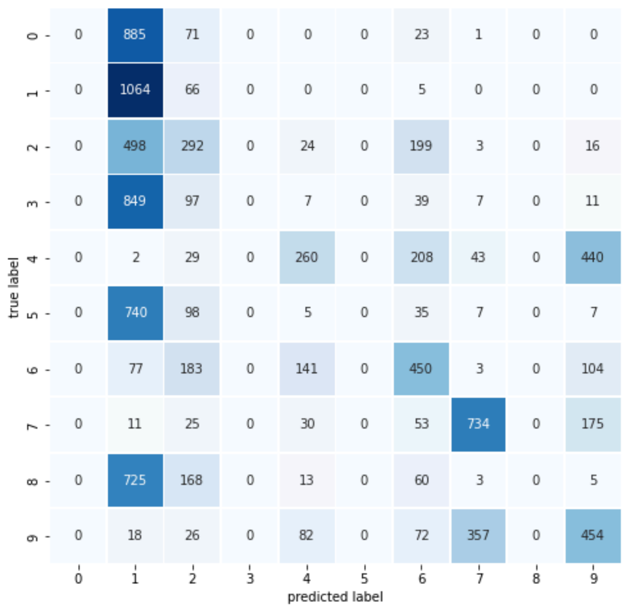
Figure 11
Models 3-8 scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model3	0.3254	3.0573	1	200	Sigmoid
Model4	0.8567	1.5408	4	200	Softmax
Model5	0.9197	1.2139	8	200	Softmax
Model6	0.853	1.5	4	500	Softmax
Model7	0.9224	1.1727	8	500	Softmax
Model8	0.9525	0.9612	32	200	Softmax

Figure 12

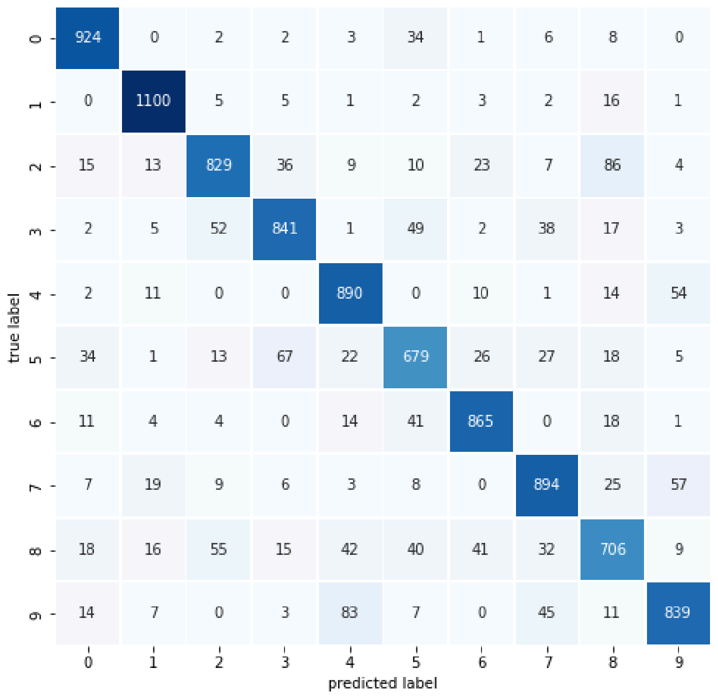
Model3s' Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	0	885	71	0	0	0	23	1	0	0
1	0	1064	66	0	0	0	5	0	0	0
2	0	498	292	0	24	0	199	3	0	16
3	0	849	97	0	7	0	39	7	0	11
4	0	2	29	0	260	0	208	43	0	440
5	0	740	98	0	5	0	35	7	0	7
6	0	77	183	0	141	0	450	3	0	104
7	0	11	25	0	30	0	53	734	0	175
8	0	725	168	0	13	0	60	3	0	5
9	0	18	26	0	82	0	72	357	0	454

Figure 13

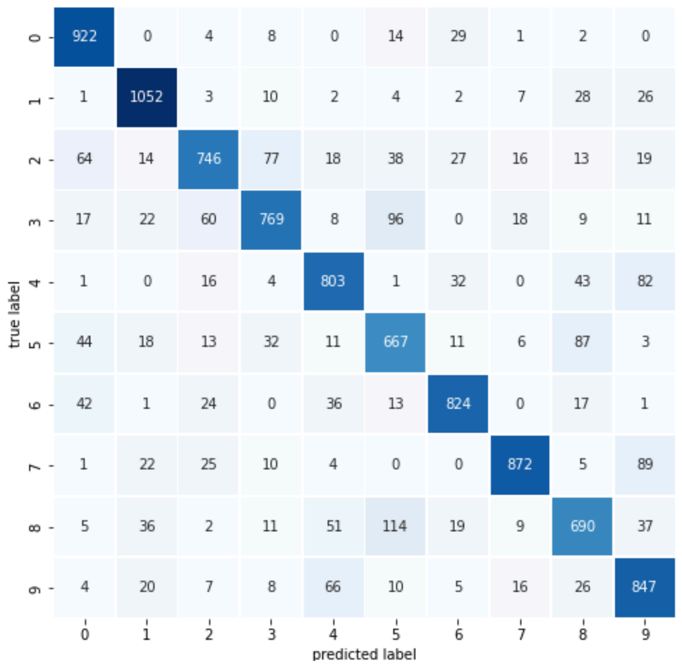
Model 4 Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	924	0	2	2	3	34	1	6	8	0
1	0	1100	5	5	1	2	3	2	16	1
2	15	13	829	36	9	10	23	7	86	4
3	2	5	52	841	1	49	2	38	17	3
4	2	11	0	0	890	0	10	1	14	54
5	34	1	13	67	22	679	26	27	18	5
6	11	4	4	0	14	41	865	0	18	1
7	7	19	9	6	3	8	0	894	25	57
8	18	16	55	15	42	40	41	32	706	9
9	14	7	0	3	83	7	0	45	11	839

Figure 14

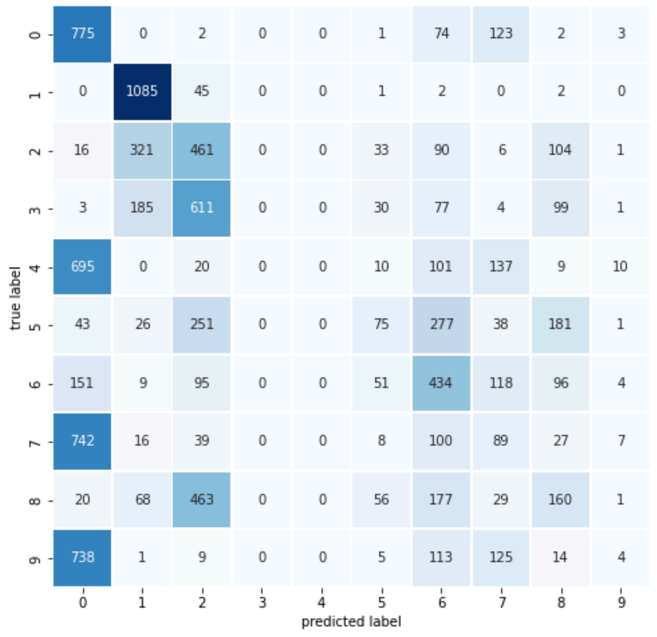
Model 5 Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	922	0	4	8	0	14	29	1	2	0
1	1	1052	3	10	2	4	2	7	28	26
2	64	14	746	77	18	38	27	16	13	19
3	17	22	60	769	8	96	0	18	9	11
4	1	0	16	4	803	1	32	0	43	82
5	44	18	13	32	11	667	11	6	87	3
6	42	1	24	0	36	13	824	0	17	1
7	1	22	25	10	4	0	0	872	5	89
8	5	36	2	11	51	114	19	9	690	37
9	4	20	7	8	66	10	5	16	26	847

Figure 15

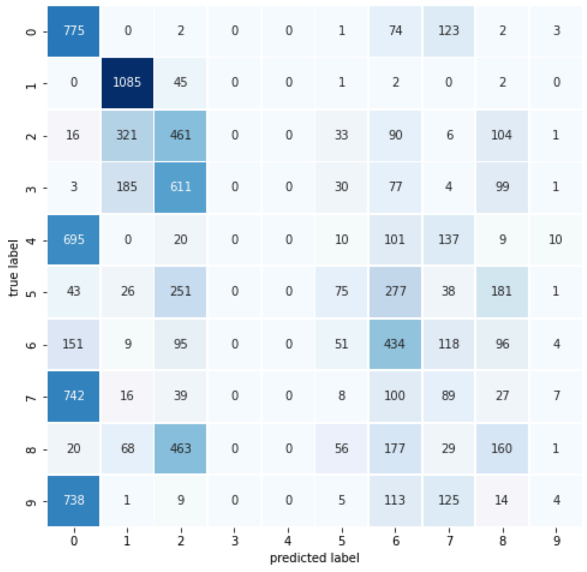
Model 5 Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

Figure 16

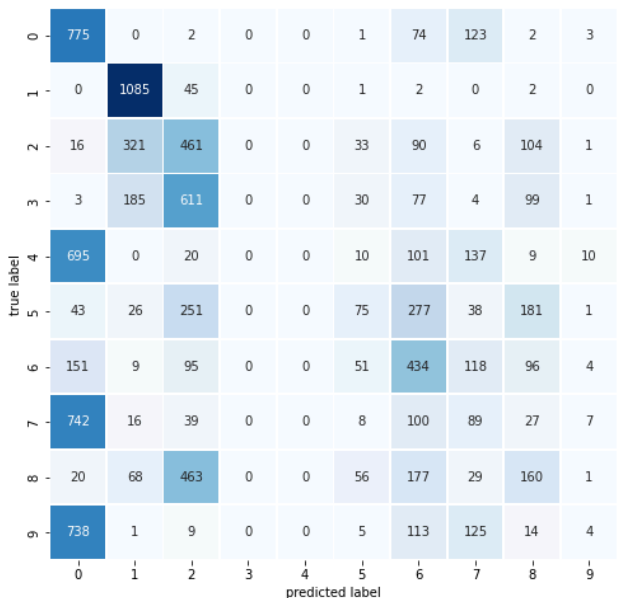
Model 6 Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

Figure 17

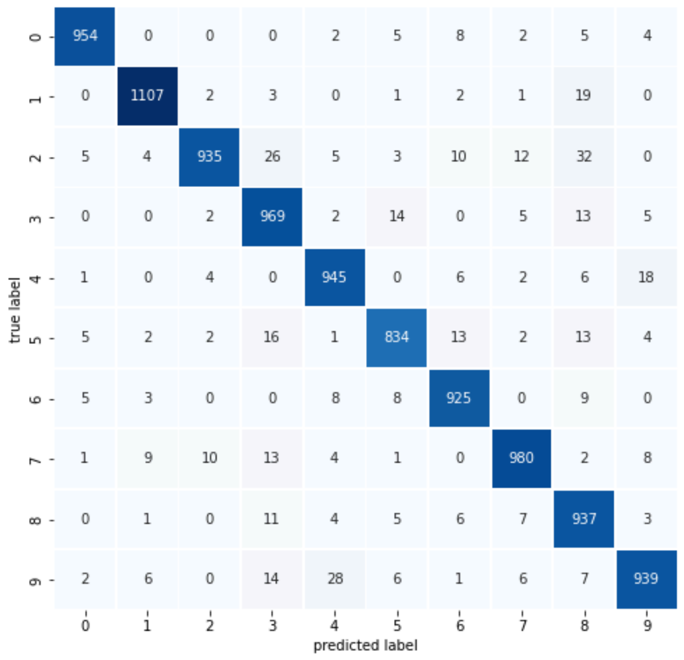
Model 7 Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	775	0	2	0	0	1	74	123	2	3
1	0	1085	45	0	0	1	2	0	2	0
2	16	321	461	0	0	33	90	6	104	1
3	3	185	611	0	0	30	77	4	99	1
4	695	0	20	0	0	10	101	137	9	10
5	43	26	251	0	0	75	277	38	181	1
6	151	9	95	0	0	51	434	118	96	4
7	742	16	39	0	0	8	100	89	27	7
8	20	68	463	0	0	56	177	29	160	1
9	738	1	9	0	0	5	113	125	14	4

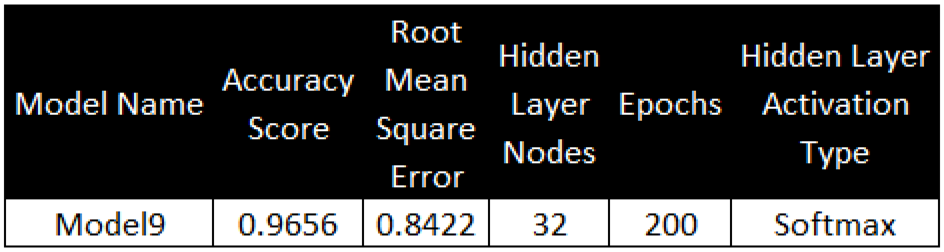
Figure 18

Model 8 Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	954	0	0	0	2	5	8	2	5	4
1	0	1107	2	3	0	1	2	1	19	0
2	5	4	935	26	5	3	10	12	32	0
3	0	0	2	969	2	14	0	5	13	5
4	1	0	4	0	945	0	6	2	6	18
5	5	2	2	16	1	834	13	2	13	4
6	5	3	0	0	8	8	925	0	9	0
7	1	9	10	13	4	1	0	980	2	8
8	0	1	0	11	4	5	6	7	937	3
9	2	6	0	14	28	6	1	6	7	939

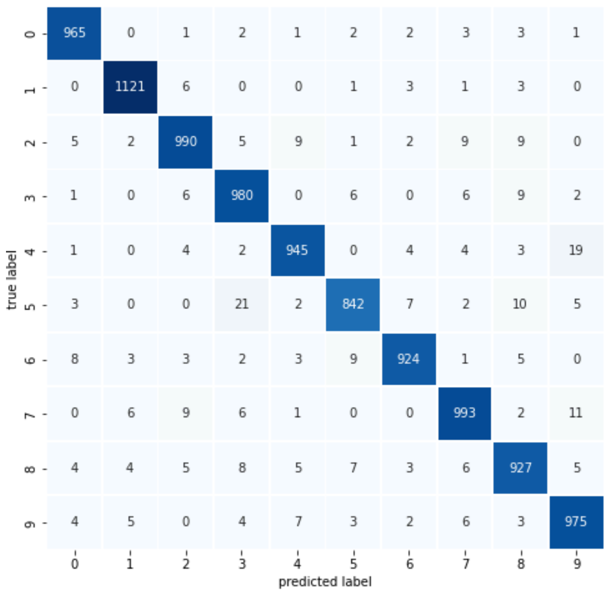
Figure 19
Model9s' scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activation Type
Model9	0.9656	0.8422	32	200	Softmax

Figure 20

Model9s' Confusion Matrix



true label \ predicted label	0	1	2	3	4	5	6	7	8	9
0	965	0	1	2	1	2	2	3	3	1
1	0	1121	6	0	0	1	3	1	3	0
2	5	2	990	5	9	1	2	9	9	0
3	1	0	6	980	0	6	0	6	9	2
4	1	0	4	2	945	0	4	4	3	19
5	3	0	0	21	2	842	7	2	10	5
6	8	3	3	2	3	9	924	1	5	0
7	0	6	9	6	1	0	0	993	2	11
8	4	4	5	8	5	7	3	6	927	5
9	4	5	0	4	7	3	2	6	3	975

Figure 21
Model10's 70 most important pixels

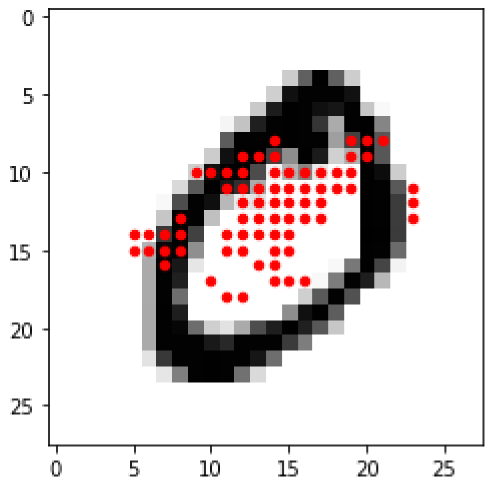
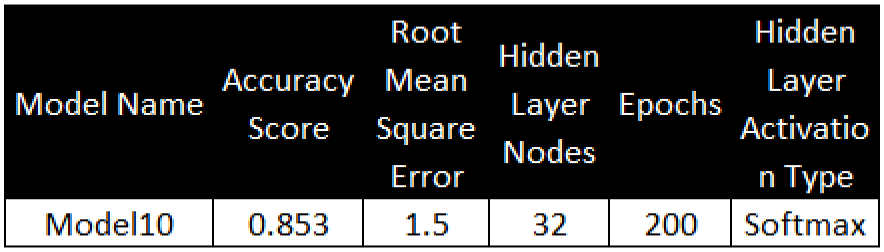


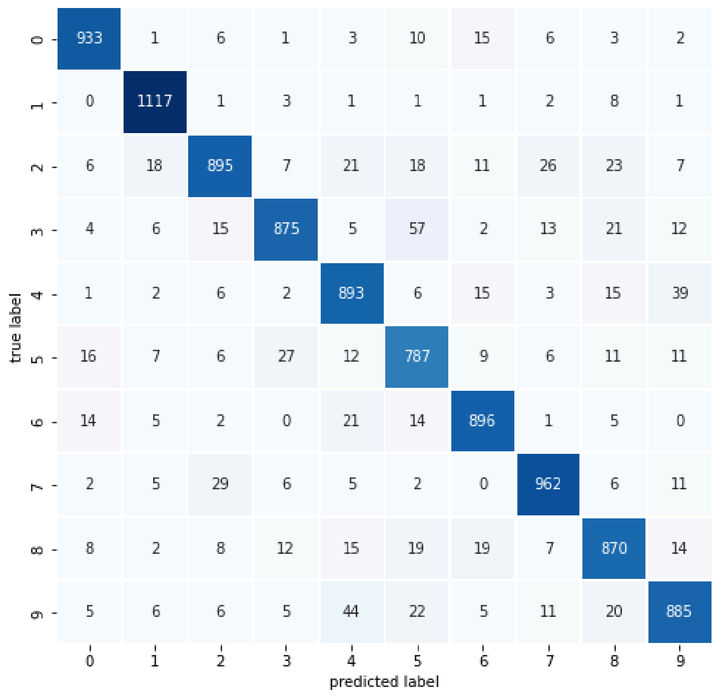
Figure 22
Model10's Scores



Model Name	Accuracy Score	Root Mean Square Error	Hidden Layer Nodes	Epochs	Hidden Layer Activatio n Type
Model10	0.853	1.5	32	200	Softmax

Figure 23

Model10's Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	933	1	6	1	3	10	15	6	3	2
1	0	1117	1	3	1	1	1	2	8	1
2	6	18	895	7	21	18	11	26	23	7
3	4	6	15	875	5	57	2	13	21	12
4	1	2	6	2	893	6	15	3	15	39
5	16	7	6	27	12	787	9	6	11	11
6	14	5	2	0	21	14	896	1	5	0
7	2	5	29	6	5	2	0	962	6	11
8	8	2	8	12	15	19	19	7	870	14
9	5	6	6	5	44	22	5	11	20	885