Project 3: Classification using MNIST and USPS dataset

Abhav Luthra
Graduate student, Computer Science
#Person 50288904
abhavlut@buffalo.edu

- **Abstract of Problem and Dataset**

  In this project we are applying machine learning methods for classification which is done using methods of Logistic regression, Random Forest, Support Vector Machine and Neural Network on MNIST dataset.

  The dataset is image of digits 0-9 which is divided into 28x28 grayscale picture and feed to the classification model for model training, testing and validation.

  50,000 images are used for training of models, 10,000 images for validation and 10,000 images for validation. 19,999 images of USPS dataset to test the model accuracy on other datasets and prove no lunch theorem.

  Brief Model Description

- **Logistic Regression Model**

  Logistic Regression is the appropriate regression analysis to conduct when the dependent variable is binary in nature. But it is not the case of this dataset and therefore data is converted and we use one hot encoding to convert target value to binary. Logistic Regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

  We use our old linear regression algorithm to try to predict y given x. Howeve, $h_\theta(x)$ to take values larger than 1 or smaller than 0 i.e. $y \in \{0, 1\}$. Our hypotheses is $h_\theta(x)$ to satisfy $0 \leq h_\theta(x) \leq 1$. This is accomplished by plugging $\theta^T x$ into the Logistic Function.

  Logistic Model uses the "Sigmoid Function," also called the "Logistic Function":

$$h_\theta(x) = g(\theta^T x)$$

$$z = \theta^T x$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

  In Logistic Regression, we use the same gradient descent approach, here the cost function is as follows:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \qquad \text{if y} = 1$$

$$\text{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \qquad \text{if y} = 0$$

Vectorized form required in this project is:

$$h = g(X\theta)$$

$$J(\theta) = \frac{1}{m} \cdot \left(-y^T \log(h) - (1-y)^T \log(1-h)\right)$$

Regularized Form used:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\ \log(h_\theta(x^{(i)})) + (1-y^{(i)})\ \log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Regularized cost function is used in this project even though there is not much difference observed between regularized cost function and non-regularized cost function.

The λ, or lambda, is the **regularization parameter**. It determines how much the costs of our theta parameters are inflated.

Cost function is minimized using the gradient descent model explained above in linear regression. SoftMax activation function is applied over the data which gives rise to temporary predictions. We compute the accuracy of the model by checking the number of samples from predicted labels that match the actual label values.

- **Neural Network Model**

Convolutional Neural Network
Convolutional Neural Networks are very similar to ordinary Neural, they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. The network consist of 3 layers - Convolutional Layer**,** Pooling Layer, and Fully-Connected Layer.

Fully- Connected Layer is nothing but a deep neural network which give prediction as 0-9. Convolutional layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. RELU layer will apply an elementwise activation function, such as the max(0,x). Pooling

layer will perform a down sampling operation along the spatial dimensions (width, height).

Deep Neural Network

DNN is, networks composed of several layers. The layers are made of nodes. A node is just a place where computation. A node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs for the task the algorithm is trying to learn. These input-weight products are summed and the sum is passed through a node's so-called activation function, to determine whether and to what extent that signal progresses further through the network to affect the ultimate outcome, say, an act of classification. DNN used here have only 3 Layer and therefore is not a DNN but a simple neural network.

The first layer has hidden nodes followed by the activation function which is 'Relu'. The choice of this activation function is to make the neurons less sparse and the computation less costly. The activation function used for the final layer is 'Softmax'. This is used as the 'softmax' function gives the categorical probability distribution of the output. Hence, a higher probability may suggest that it belongs to that particular class. We also add dropout to the model to prevent overfitting of data. We use 'categorical_crossentropy' loss function and 'rmsprop' as an optimizer. Further, we use early stopping to stop the training if there is no improvement while training.

- **Support Vector Machine**

  A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data, the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side. It has **regularization parameter** and **gamma.** These are tuning parameters in SVM classifier. Varying those we can achieve considerable non-linear classification line with more accuracy in reasonable amount of time.

  For **linear kernel** the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

  $f(x) = B(0) + sum(ai * (x,xi))$

  The **Regularization parameter** tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

  The **gamma** parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Whereas high gamma means the points close to plausible line are considered in calculation.

A margin is a separation of line to the closest class points. A **good margin** is one where this separation is larger for both the classes.

How to chose between Linear and RBF?
 Use linear kernel when number of features is larger than number of observations.
 Use gaussian kernel when number of observations is larger than number of features.
 If number of observations is larger than 50,000 speed could be an issue when using gaussian kernel; hence, one might want to use linear kernel

- **Random Forest**

Random forest is a classifier which is a group of different classifiers known as decision tree. Many trees together form a random forest. Where, nodes of the decision tree. are the features and the edges are the values that a node can take.

Random Forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node.

Random Forests are also very hard to beat in terms of performance. Of course you can probably always find a model that can perform better, like a neural network, but these usually take much more time in the development. And on top of that, they can handle a lot of different feature types, like binary, categorical and numerical.

Confusion Matrix

Confusion matrix is the one way to visualize the accuracy of the classifier. Every cell of matrix represents digits from 0 to 9. Ideally, the confusion matrix should be diagonal, this shows the 100% accuracy. Matrix gives a detailed analysis of misclassified classes and the represents the no of times the classifier got confused with other class.

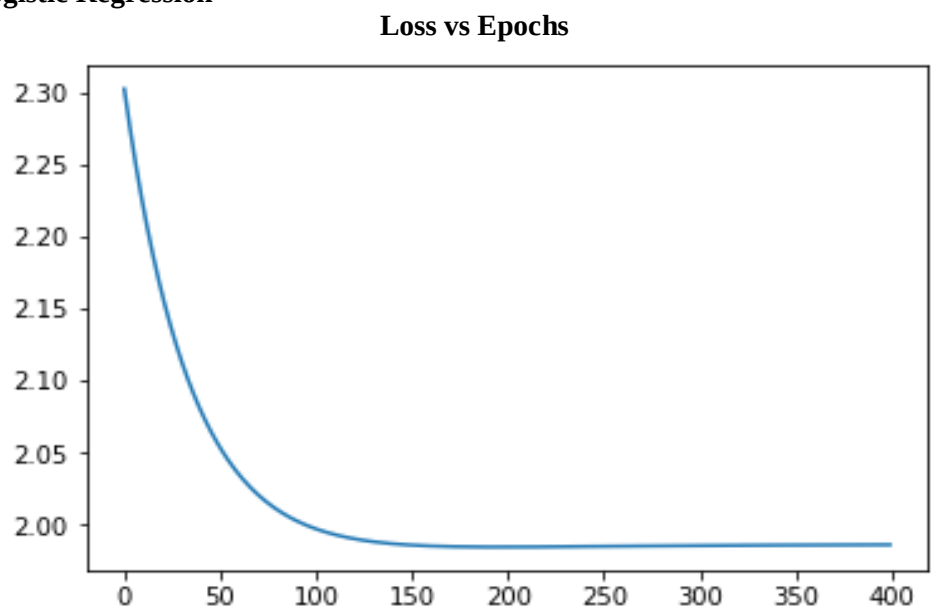
Ensemble Learning

Ensemble is the process of considering the output of multiple classifiers together and predict the output class accordingly. We have implemented 5 classification methods and consider the combined output and check whether we can conclude some better results with the combined results. This process is called as ensemble.

I have done Ensemble Learning by Majority Voting:
Max Voting In this. every classifier has predicted a particular class for every sample. Since the models are well trained, there are higher chances of maximum number of classifiers to predict the label accurately. We take the statistical mode of the row and returns the class that has occurred maximum times and consider it as a final predicted output for the image.

Model Training

1. Logistic Regression



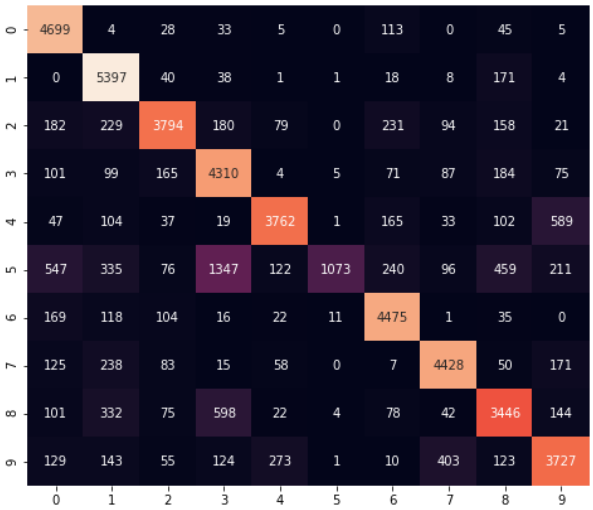
Model Output

	Training	Validation	Testing	USPS
Accuracy	0.7822	0.8005	0.798	0.2892

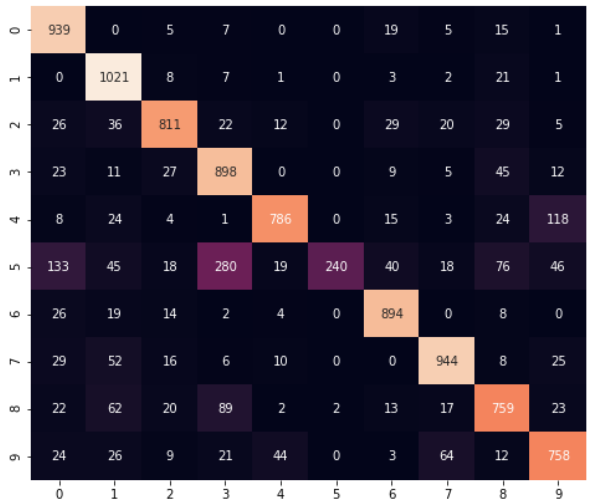
Logistic Regression perform fairly well on MNIST dataset and poorly on USPS.

Loss in logistic regression become constant after 150 + iterations

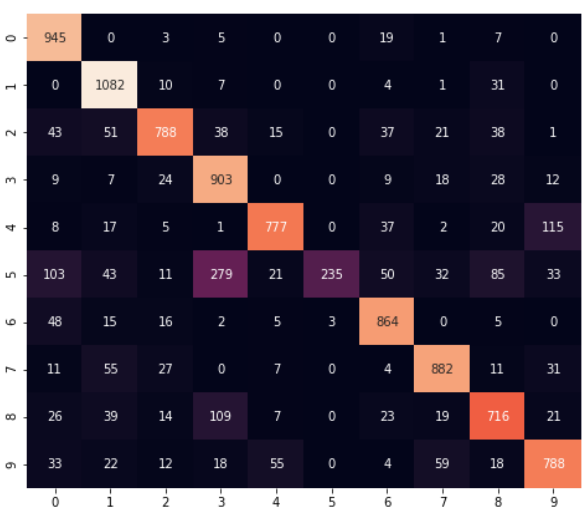
Using confusion matrix we can see that 5 is most of the time predicted as 3 therefore driving the accuracy down in training, testing and validation. Logistic regression model trained on MNIST dataset perform very vaguely on USPS dataset and therefore can't be used to predict values on this dataset.



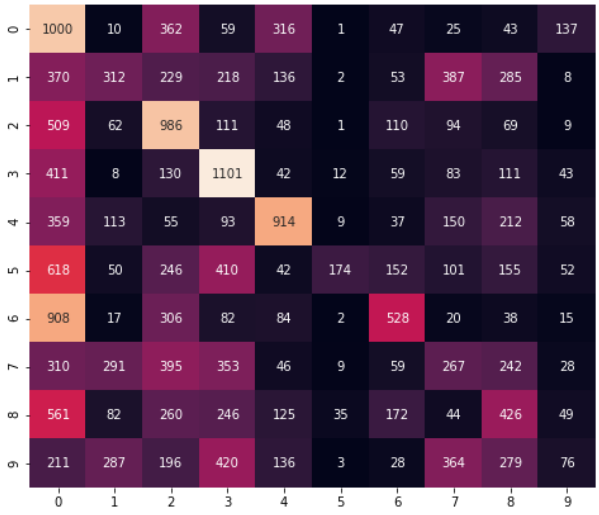
Training



Validation



Testing



USPS

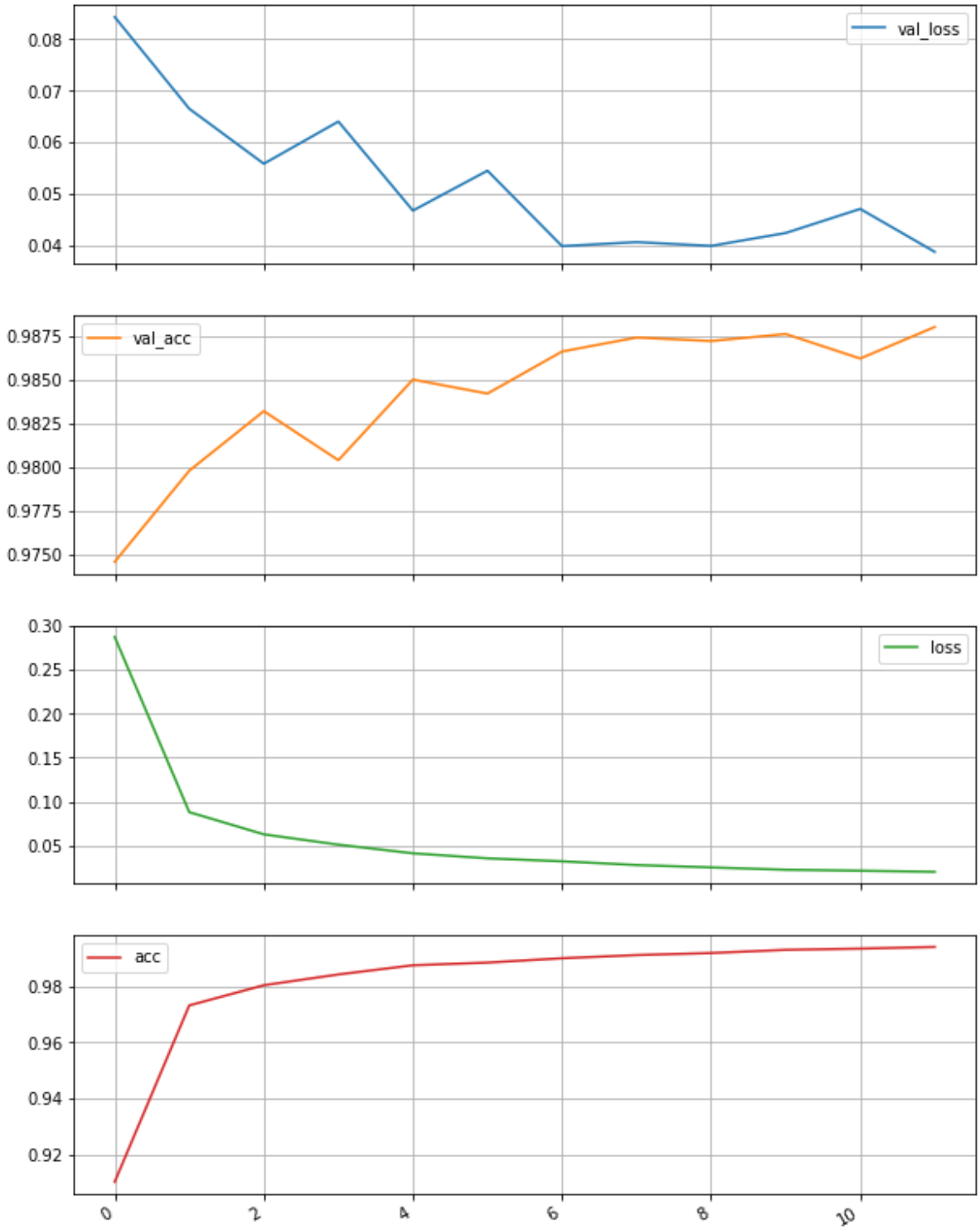
2. Neural Network

a) Convolutional Neural Network

Convolutional Network is proven to be one of the most efficient classifier not only for MNIST dataset but also for USPS dataset as it takes images as input. I have taken the filter size as 32 and kernel size as 3x3 and applied the relu activation function after a lot of testing. Then we pooled the data and put it into a 256 hidden neural network.

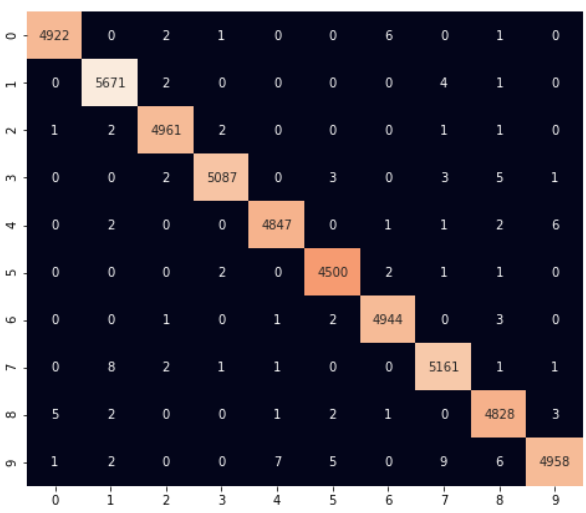
Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320
activation_1 (Activation)	(None, 26, 26, 32)	0
conv2d_2 (Conv2D)	(None, 24, 24, 32)	9248
activation_2 (Activation)	(None, 24, 24, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 12, 12, 32)	0
dropout_1 (Dropout)	(None, 12, 12, 32)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 256)	1179904
activation_3 (Activation)	(None, 256)	0
dropout_2 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 10)	2570
activation_4 (Activation)	(None, 10)	0

Total params: 1,192,042
Trainable params: 1,192,042
Non-trainable params: 0

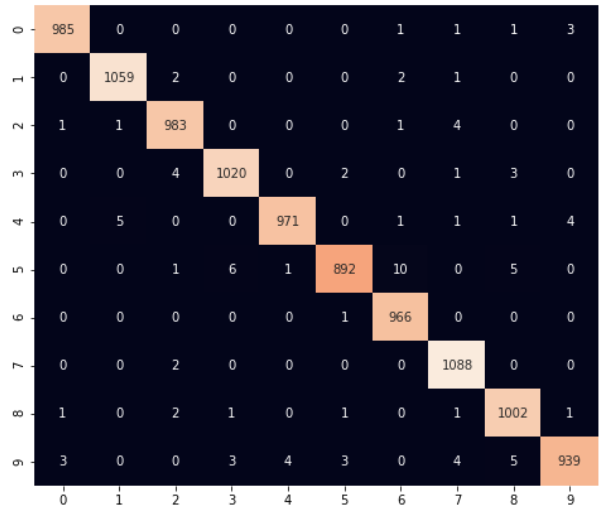


Model gets trained at around 6 epoch and we stop it at 12 epoch. Loss value decreases and accuracy increases as model get trained on training and validation dataset of MNIST. Due to large no of parameters the model training and testing is very time and memory consuming process.

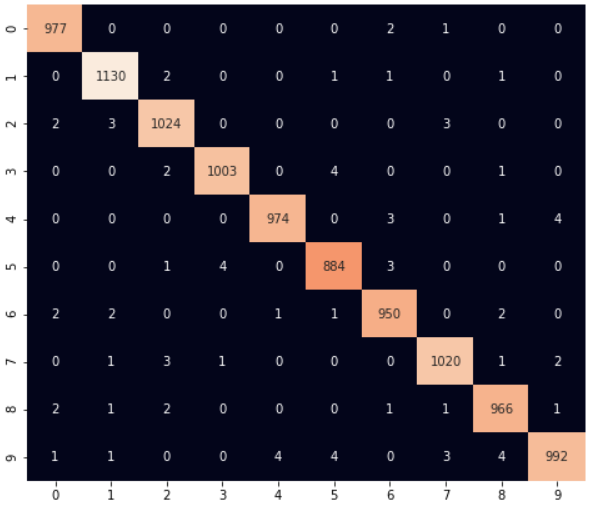
Using confusion matrix we can see that model perform perfectly on training, testing and validation dataset of MNIST. CNN model trained on MNIST dataset perform 60% on USPS dataset which is much better than all of the other classifiers.



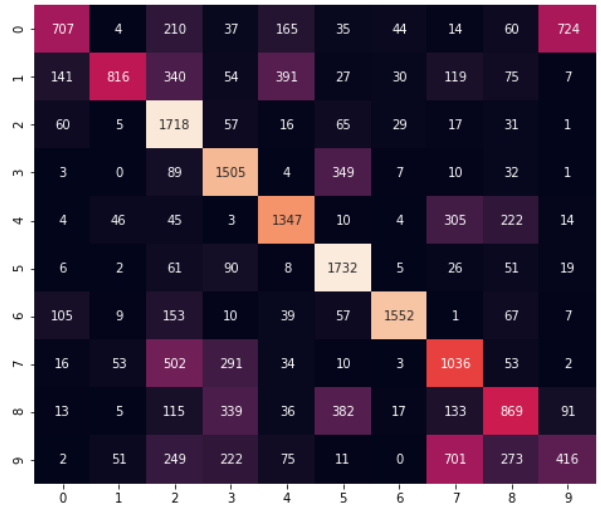
Training



Validation



Testing



USPS

b) Deep Neural network(Single Layer)

I trained the model on 256 and 512 layers and observed similar accuracy and therefore took 256 hidden layer model for final evaluation as 256 layer are more than enough.

first_dense_layer_nodes = 256

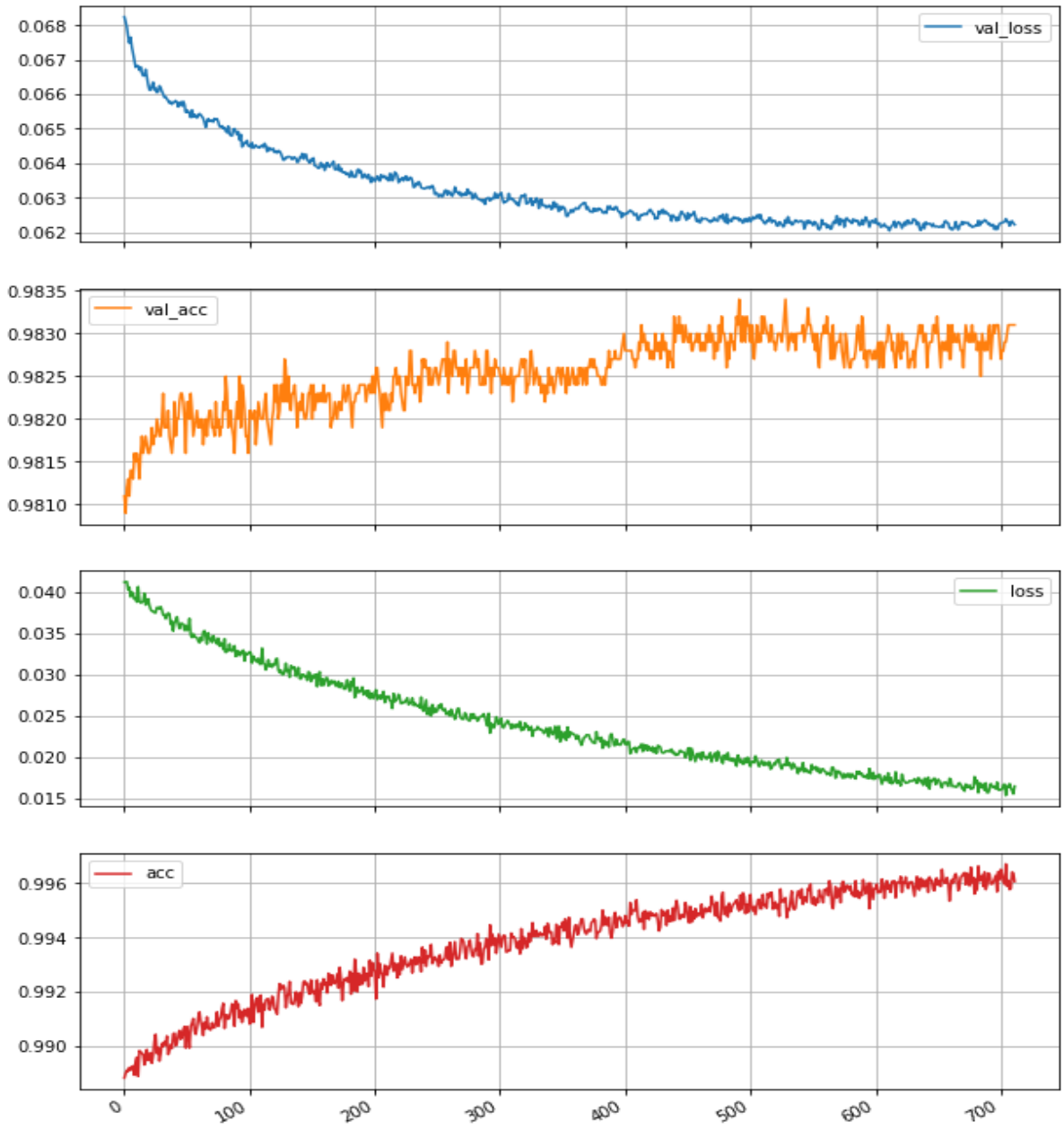
Model Layout

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 256)	200960
activation_1 (Activation)	(None, 256)	0
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 10)	2570
activation_2 (Activation)	(None, 10)	0

Total params: 203,530
Trainable params: 203,530
Non-trainable params: 0

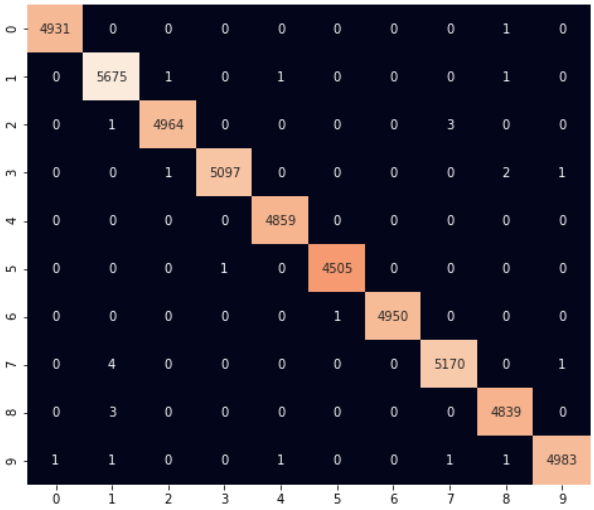
Model Output

	Training	Validation	Testing	USPS
Accuracy	0.9961	0.9831	0.9817	0.5009

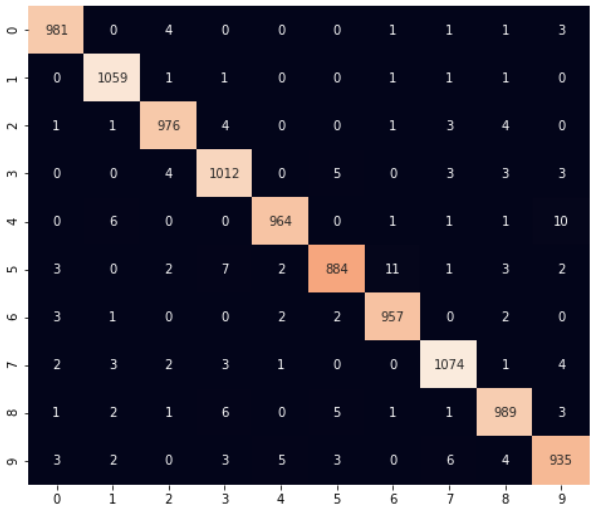


Model gets trained at around 600 epoch and do early stopping at 700 epochs. Loss value decreases and accuracy increases as model get trained on training and validation dataset of MNIST.

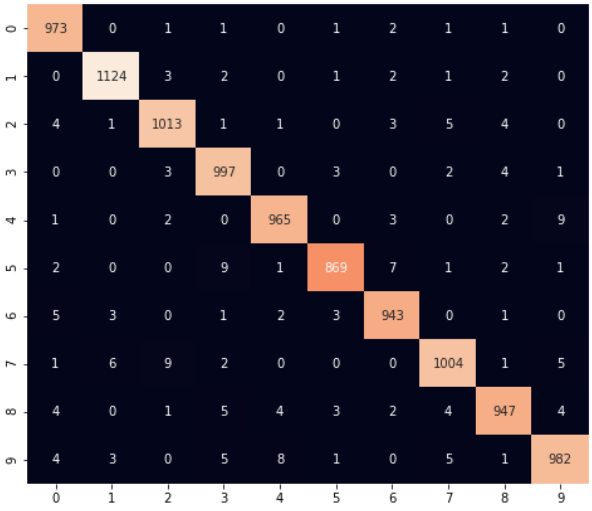
Using confusion matrix we can see that model perform perfectly on training, testing and validation dataset of MNIST. DNN model trained on MNIST dataset perform 50% on USPS dataset and therefore predicted values have equal chance of being correct and not and therefore we can't use it to predict value in USPS dataset.



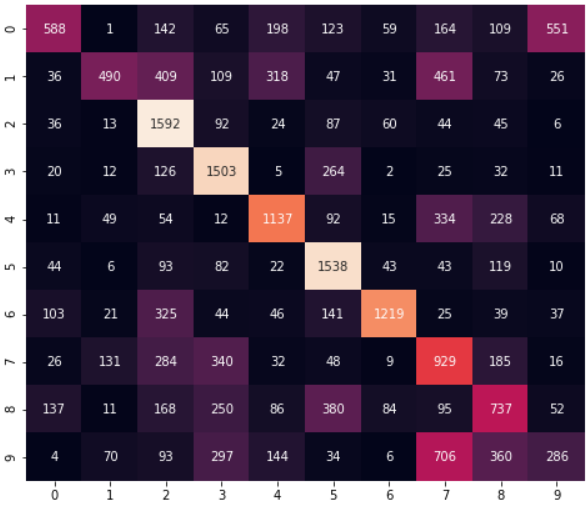
Training



Validation



Testing



USPS

first_dense_layer_nodes = 512

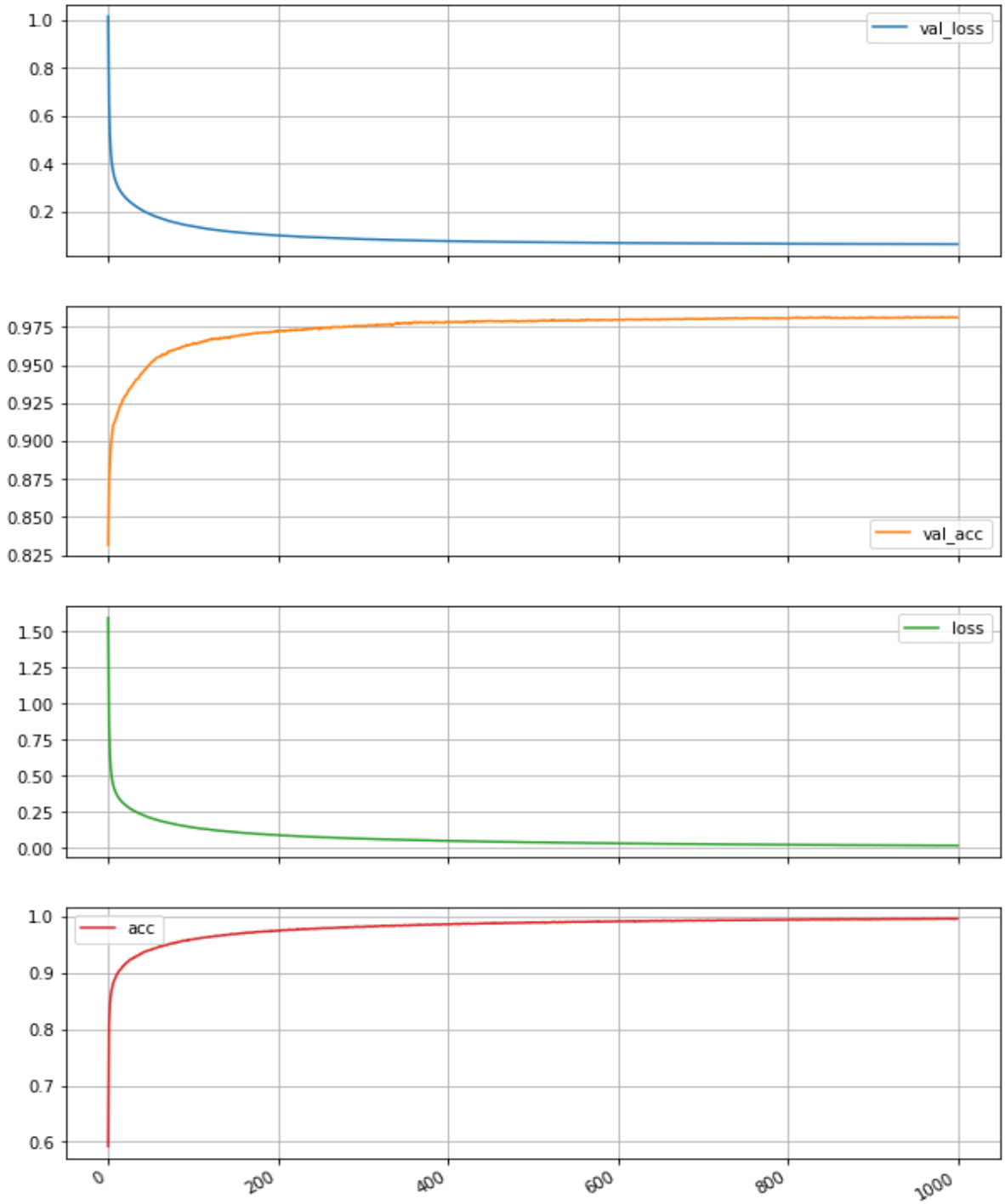
Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 512)	401920
activation_3 (Activation)	(None, 512)	0
dropout_2 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 10)	5130
activation_4 (Activation)	(None, 10)	0

Total params: 407,050
Trainable params: 407,050
Non-trainable params: 0

Model Output

	Training	Validation	Testing	USPS
Accuracy	0.9961	0.9812	0.9822	0.4904

This model give same training accuracy as 256 hidden layer model but validation, testing and USPS accuracy decreases a little. But the graph is much more smother as it captures noises perfectly.

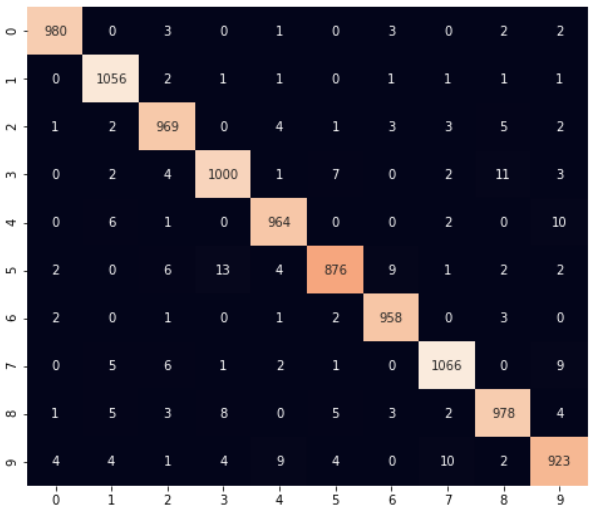


Model is stopped at 1000 epochs even though it has early stopping. Loss value decreases and accuracy increases as model get trained on training and validation dataset of MNIST.

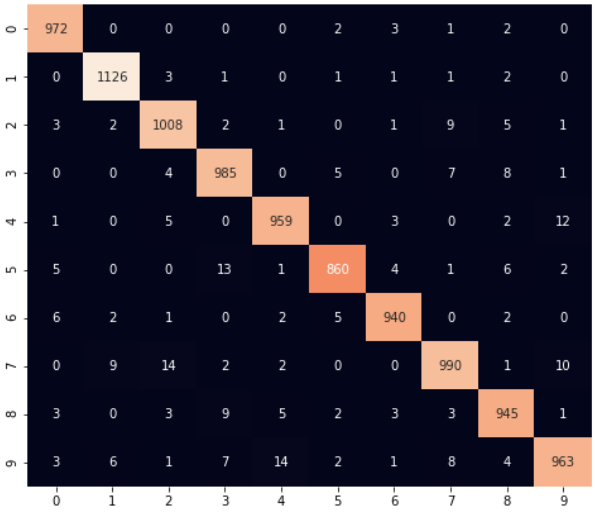
3. SVM

a) Radical basis Function

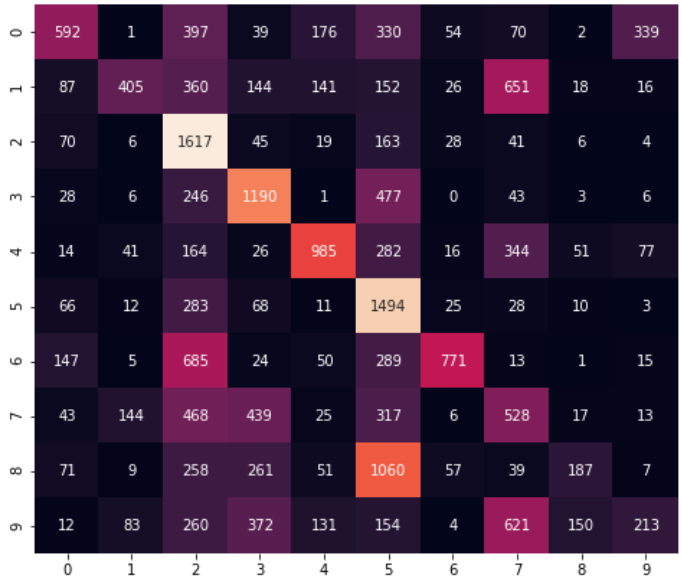
	Training	Validation	Testing	USPS
C=2; g = 0.005	0.97904	0.9744	0.9711	0.3997
C=2; g = 0.001	0.94.39	0.9470	0.9456	0.3852
C=2.5; g = 0.05	0.98124	0.9761	0.9727	0.3988
C=1.5; g = 0.05	0.97586	0.9718	0.9687	0.3983
C=3; g = 0.05	**0.98366**	**0.9770**	**0.9748**	**0.3991**



Validation



Testing

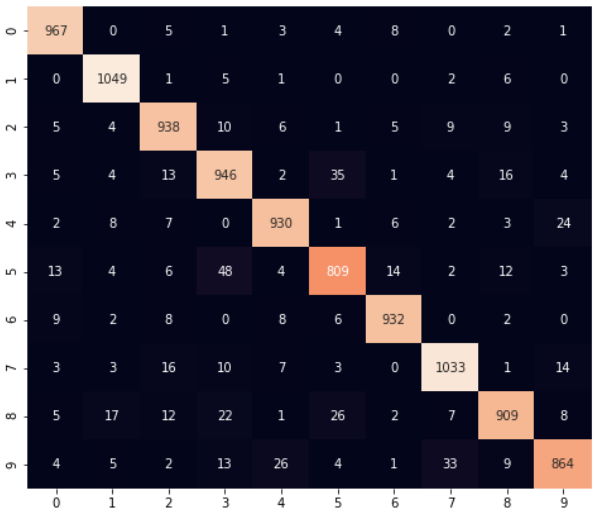


USPS

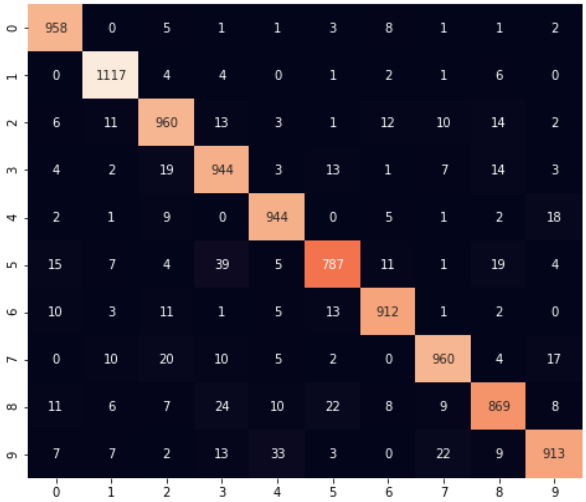
Using confusion matrix we can see that model perform well on training, testing and validation dataset of MNIST. SVM model trained on MNIST dataset perform less than 40% on USPS dataset and therefore predicted values can't be used for this dataset

b) Linear Kernel

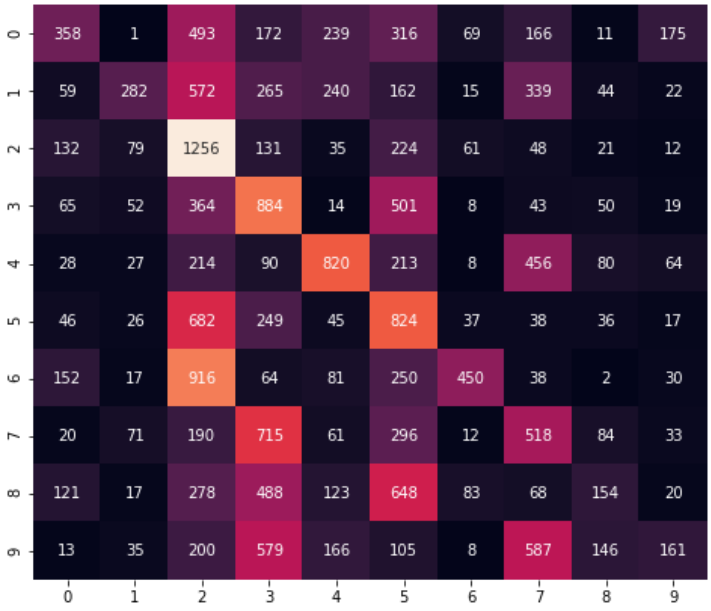
	Training	Validation	Testing	USPS
C=2; g = 0.005	0.97582	0.9377	0.9364	0.2854
default	0.92618	0.9208	0.9155	0.2605



Validation



Testing



USPS

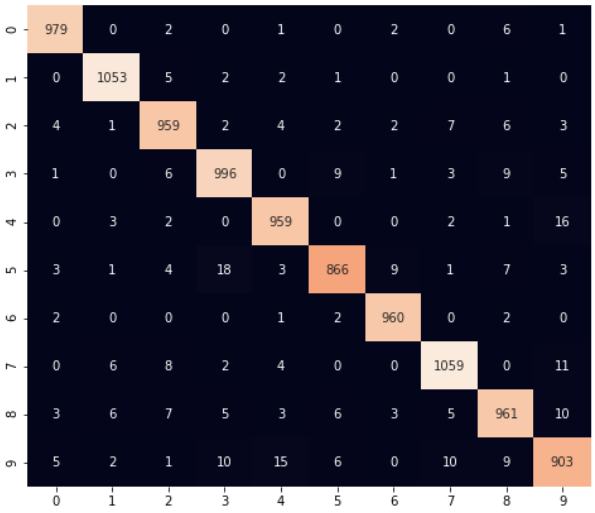
Using confusion matrix we can see that model perform well on training, testing and validation dataset of MNIST. SVM model trained on MNIST dataset perform less than 30% on USPS dataset and therefore predicted values can't be used for this dataset

Linear Kernel overall performance is much lower than radical basis function as therefore is it is not considered for ensemble learning.

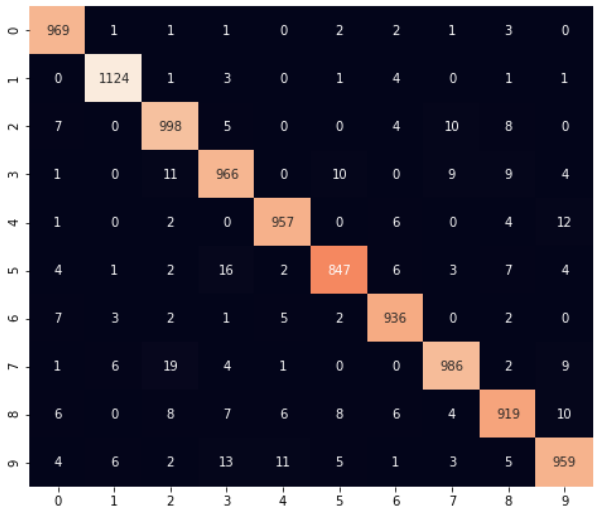
4. Random Forest

For Tree size = 50

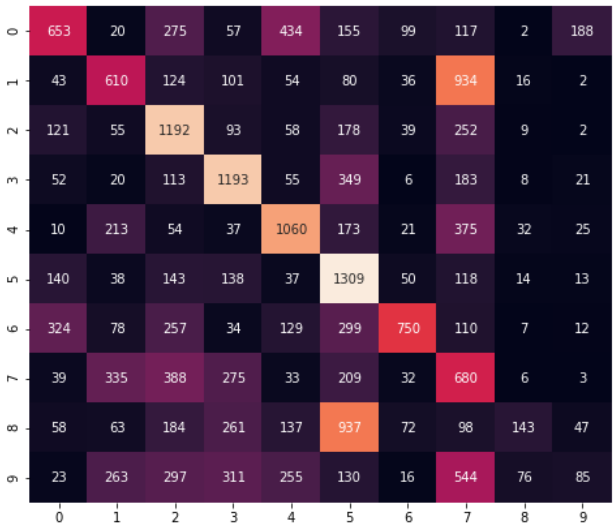
Model perform perfectly for MNIST dataset but perform below 40% for USPS and therefore cannot be used for USPS dataset.



Validation



Testing



USPS

Model Accuracy for Various Tree Sizes

Tree Size	Training	Validation	Testing	USPS
5	0.99348	0.9204	0.9188	0.2699
10	0.99928	0.9498	0.9431	0.3072
15	0.99966	0.9581	0.9539	0.3355
20	0.9999	0.9613	0.9567	0.3429
50	**1.0**	**0.9695**	**0.9661**	**0.3838**
100	1.0	0.9721	0.9684	0.3906
200	1.0	0.9745	0.9688	0.3983

Tree size of 50 is used as after that improvements are very marginal.

Summary

	Logistic Regression	Random Forest	SVM	DNN	CNN
Train	78.36	100	98.36	99.61	99.75
Test	80.05	96.61	97.48	98.17	99.2
Validation	80.77	96.95	97.70	98.31	99.05
USPS	28.95	38.38	39.91	50.09	58.49

Models selected for Ensemble learning

Ensemble Learning done through Majority Voting

	Training	Validation	Testing	USPS
Accuracy	0.99858	0.9839	0.9806	0.5152

We can observe that Accuracy observed is after ensemble is much larger or similar than all of the individual model and therefore give trust to the system as output is correct.

1. No Free Lunch Theorem
   The **"No Free Lunch" theorem** states that there is no one model that works best for every problem. Now the USPS dataset and MNIST dataset have the same problem of identifying numbers from images but the dataset might be procured differently so, even though it's the same problem. The models did not perform very well on USPS data. Thereby proving that this model cannot be used on any dataset of images that has numeric data. It can only perform well on data which is very similar to it.

2. Relative Strength and Weakness of Each Classifier

   Convolutional Network give overall best performance.

   Convolutional Network
   Advantages - Convolutional networks make the assumption of locality, and hence are more powerful for images or time series. We can save a lot of neuron wiring if you don't (directly) wire distant pixels to the same neuron.
   Disadvantage – Takes a lot more processing power due to added complexity.

   Random Forest
   Advantages -Reduction in overfitting: by averaging several trees, there is a significantly lower risk of overfitting. Also, by using multiple trees, you reduce the chance of stumbling across a classifier that doesn't perform well because of the relationship between the train and test data.
   Disadvantage -could be more complexity, hard to visualize the model or understand why it predicted something.

   SVM

   Advantages - SVM's are very good when we have no idea on the data. Works well with even unstructured and semi structured data like text, Images and trees. The kernel trick is real strength of SVM. With an appropriate kernel function, we can solve any complex problem. SVM models have generalization in practice, the risk of overfitting is less in SVM.

Disadvantages - Long training time for large datasets. Difficult to understand and interpret the final model, variable weights and individual impact.

Logistic Regression

Advantages - it is more robust: the independent or dependent variables don't have to be normally distributed, or have equal variance in each group. It does not assume a linear relationship between the independent and dependent variables. We can add explicit interaction and power terms

Disadvantage – It trains slower than rest of the model i.e. require higher training data to achieve higher accuracy.

3. Is overall combined model better than individual model?
   Yes, except for Convolutional Neural Network, the combined model give much better accuracy than individual model as they overlap each other weakness and make the combined model strong.