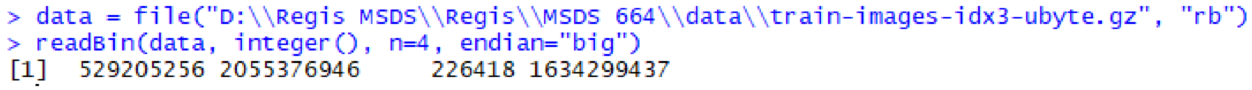
Introduction

Today I wanted to use deep learning techniques in R to train a dataset using the h2o package. I wanted to train a dataset to investigate prediction performance with a limit of 2 to 3 runs in h2o's package. I decided to use the h2o package to train and test the dataset for the MNIST digit classification dataset. I could have used a different dataset, but I wanted to get more of a feel for the h20 package before doing so.

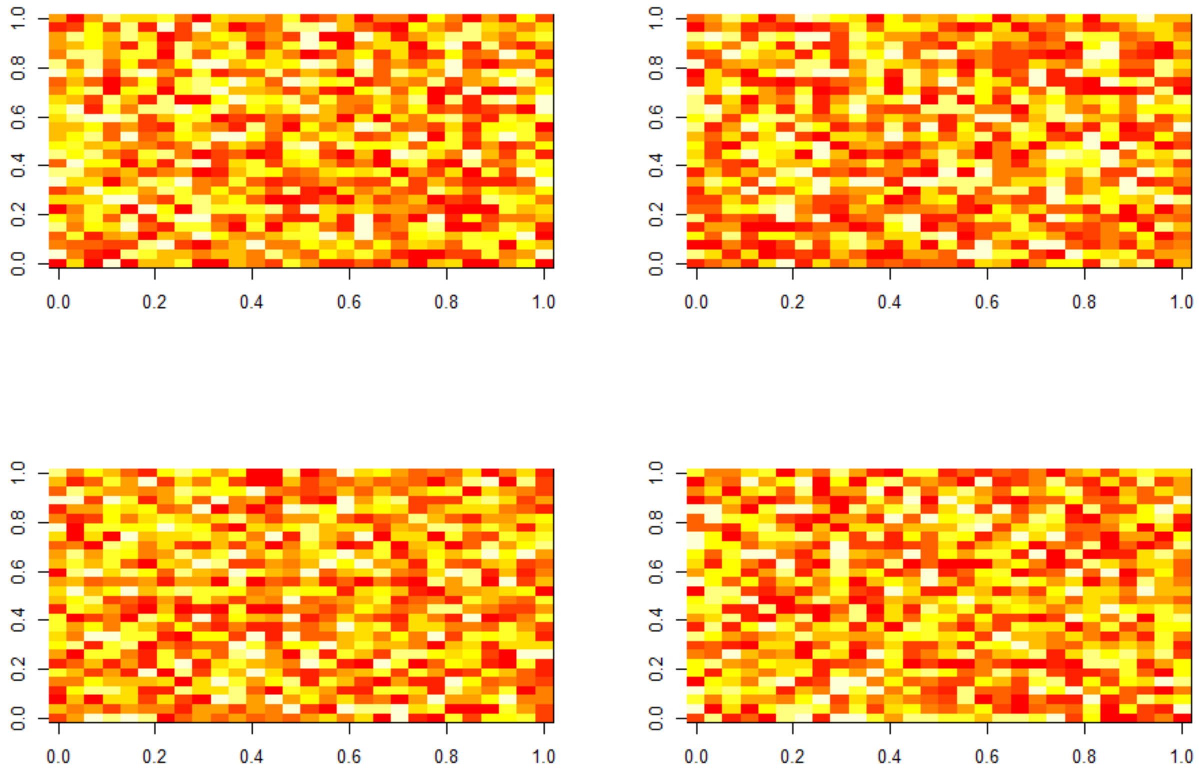
Methods and Results

First, we read in the data set and upload all the packages. Then we are going to use the image data from the site to complete the classification exercise. We read in the data first and then must make sure it is in the correct format. I wanted to load the data and then use the readBin() command to look at is before going further.

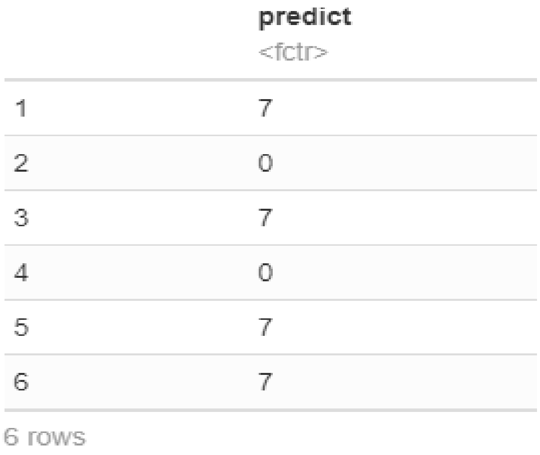


```
> data = file("D:\\Regis MSDS\\Regis\\MSDS 664\\data\\train-images-idx3-ubyte.gz", "rb")
> readBin(data, integer(), n=4, endian="big")
[1]  529205256 2055376946     226418 1634299437
```

I then read in the data to its proper .gz form and make sure that it shows up in the correct images that it's supposed to have.

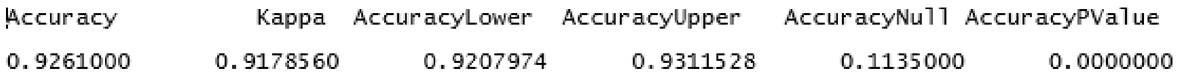


We then start the h2o process and load in the data. We want to make sure we can build models for testing and interpretation. So, we will also change the format of the data in order to do this in the h2o package. Once the data is loaded we will go ahead and build 3 models max to see how each prediction does. We will use the caret package that we loaded in order to run the models that we will get from the h2o objects we see. We start with a prediction model with the data we have.



	predict <fctr>
1	7
2	0
3	7
4	0
5	7
6	7

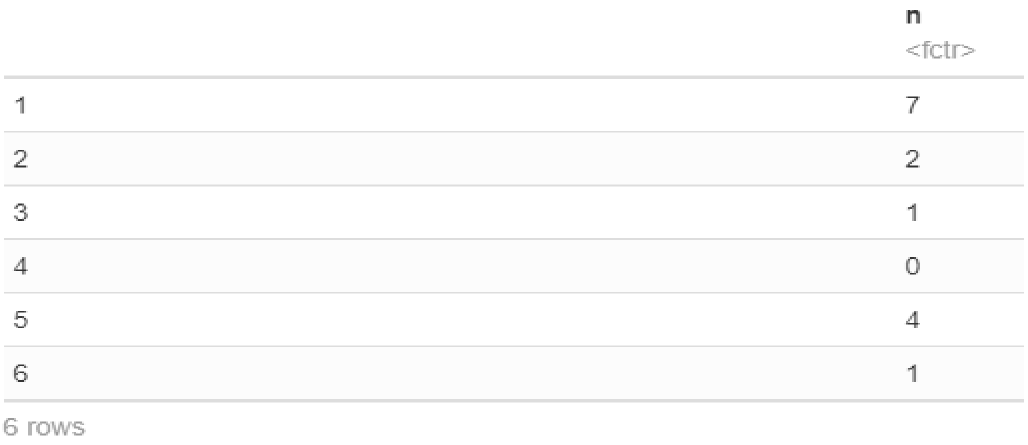
6 rows



```
Accuracy           Kappa  AccuracyLower  AccuracyUpper   AccuracyNull AccuracyPValue
0.9261000      0.9178560      0.9207974      0.9311528      0.1135000      0.0000000
```

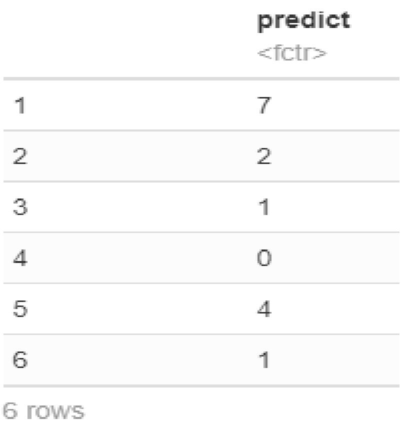
As we can see the prediction that came out was not very good at all. Yes, the accuracy was high, however for the wrong numbers. We are seeing 7's come out of the prediction, however we are not supposed to have that. One thing we notice is that we originally had two nodes for the 3 layers we had and 10 epochs as well. What we can do is increase those numbers and see if it helps by retesting with the caret package.

What we can do is increase the node and epoch numbers and see if that helps. We will also experiment with adding a 5-fold data validation for a third model and in a 4th model we will reduce the input dropout ratio, which is a feature that controls what ratio of features is dropped for a training row. It seemed that since we are adding validation, we could also lower this number and see if it helps any.



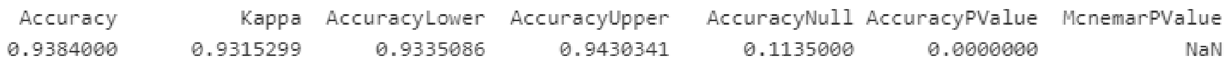
	n <fctr>
1	7
2	2
3	1
4	0
5	4
6	1

6 rows



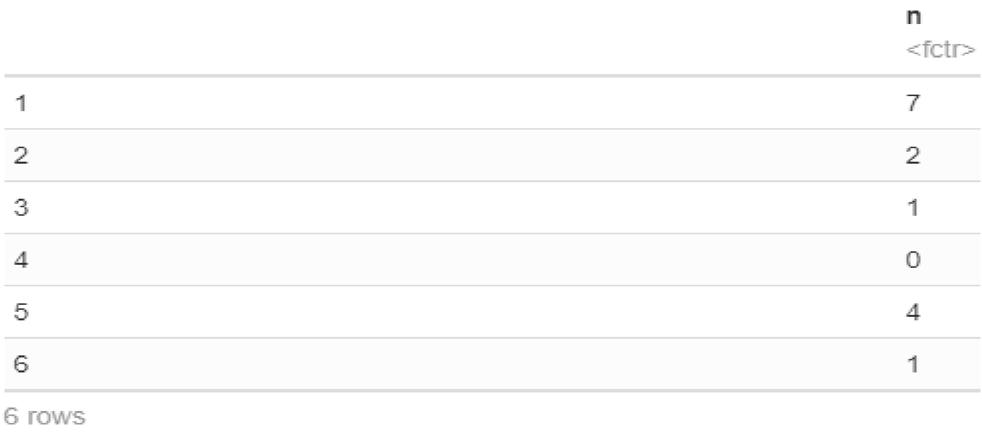
	predict <fctr>
1	7
2	2
3	1
4	0
5	4
6	1

6 rows



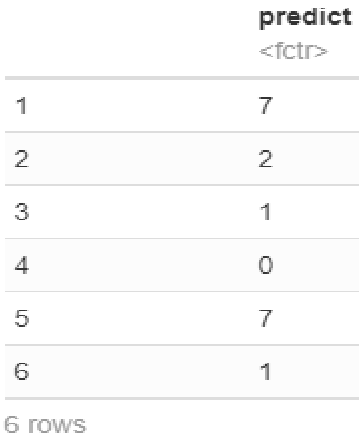
```
 Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull AccuracyPValue  McnemarPValue 
0.9384000      0.9315299      0.9335086      0.9430341      0.1135000      0.0000000            NaN 
```

In model 2 we also see a high accuracy, however that might be due to the large increase in the amount of operations that were done.



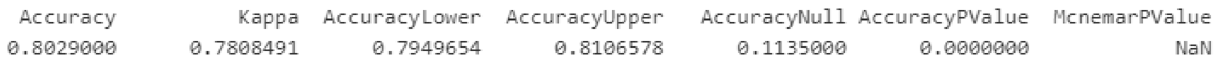
	n <fctr>
1	7
2	2
3	1
4	0
5	4
6	1

6 rows



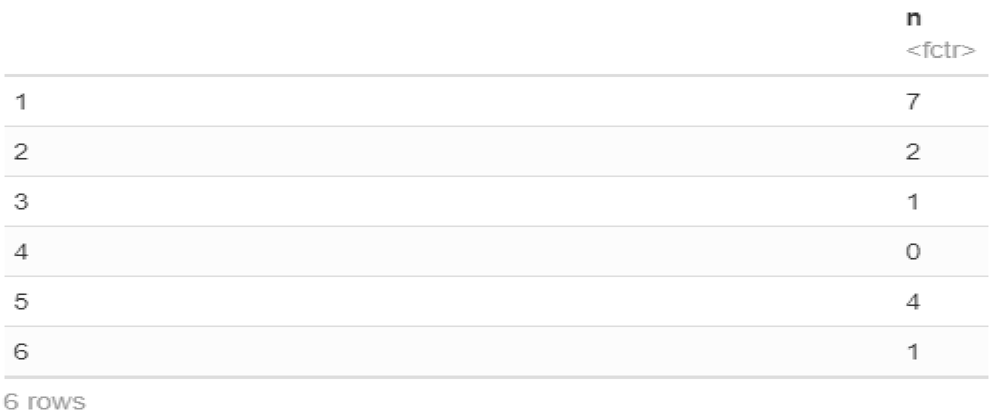
	predict <fctr>
1	7
2	2
3	1
4	0
5	7
6	1

6 rows



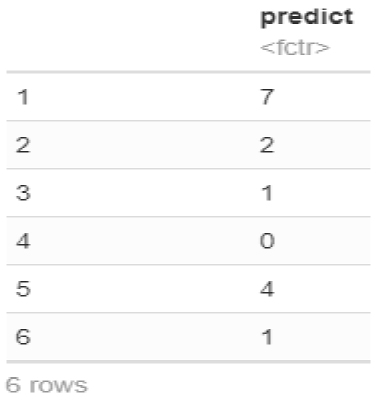
```
 Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull AccuracyPValue  McnemarPValue 
0.8029000      0.7808491      0.7949654      0.8106578      0.1135000      0.0000000            NaN 
```

Here in model three we see an impact on the accuracy and how low it got when we reduced the layers used.



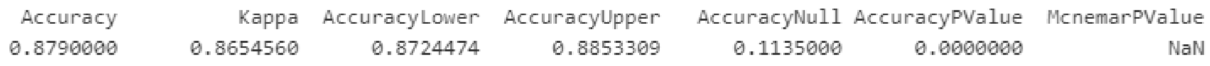
	n <fctr>
1	7
2	2
3	1
4	0
5	4
6	1

6 rows



	predict <fctr>
1	7
2	2
3	1
4	0
5	4
6	1

6 rows



```
      Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull AccuracyPValue  McnemarPValue 
     0.8790000      0.8654560      0.8724474      0.8853309      0.1135000      0.0000000            NaN 
```

We finally see here in model 4 that it was an improvement on model 3 but not an overall improvement like we had in model two.

Analysis and Interpretations

It appears the second model was the overall best with the layers that we used in terms of accuracy. However, there does not seem to be any finite amount of modifications one can di in order to find the best model. If we wanted to find the best model with the best accuracy, then we could truly go for a very long time with how many models we could make to find the best one. We could ultimately find almost infinite models along the way using techniques such as variable importance in order to truly find the most accurate model. So, in order to be able to complete this assignment it seems that model two was our best model for classification purposes. We can say that it is not perfect or even the absolute best we could find, but if given more time we could find a better model. However, the second model did in fact classify the digits sufficiently, and we can see that in the models when the first few digits match up the strength is better.

Conclusion

This analysis helped facilitate the classification of the data and understand the interpretations of the h2o neural network results. This would be a great way to look at data from multiple angles and be able to classify basic digits and other images on a larger scale.

References:

1. MNIST database. Avaialable at: http://yann.lecun.com/exdb/mnist/
2. Candel, A., LeDell, E., Parmar, V., and Arora, A. (2016). Deep Learning with H2O. Retrieved from: https://github.com/h2oai/h2o-3/blob/master/h2o-docs/src/booklets/v2_2015/PDFs/online/DeepLearning_Vignette.pdf
3. LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep Learning. Nature. Volume 521. Retrieved from: http://www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf