**Module 3 Predictive Analytics**

**Author’s Name:** Shivani Sharma

**Course Number:** ALY6020

**Course Title:** Predictive Analytics

**Academic Term:** Fall 2019 CPS Analytics

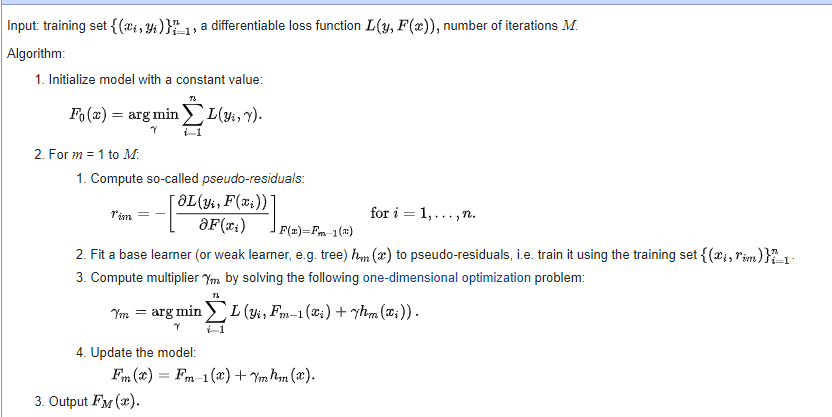
**Instructor’s Name:** Marco Montes de Oca

**Assignment Completion Date:** 10-06-2020



**Introduction**

We have been provided with a problem statement based on transforming images into 7 segment display using “Gradient Boosting Algorithm”. This algorithm is used for the classification/regression. We construct ensembles on “ Decision Trees” and then trees are added one by one to the ensemble and fitted to correct the “predicted errors” of the model. Here is the algorithm of this model:



There are following points that we need to take care of in this algorithm:

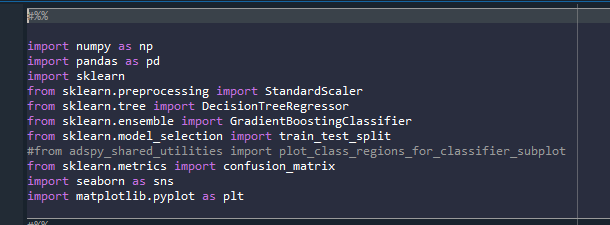
1. Size of the trees we are creating
2. Regularization technique to avoid overfitting
3. Shrinkage i.e learning rate
4. Number of records in the tree’s leaves
5. When to penalize the tree for handling the complexity

And as per the Problem statement that we have in this week, we have to first figure out the values of different segments of activation pattern of the digital digit and then we have to apply gradient boosting model on all the segments individually.

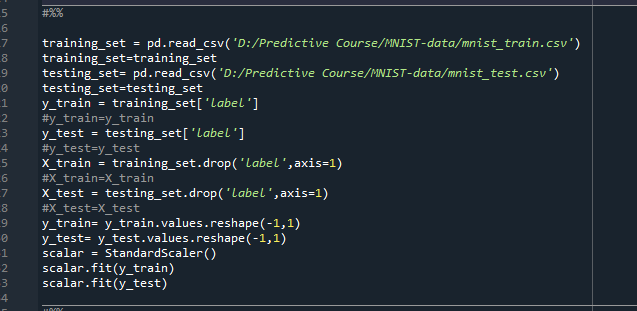
I will explain all the segments with logic in analysis section.

**Analysis**

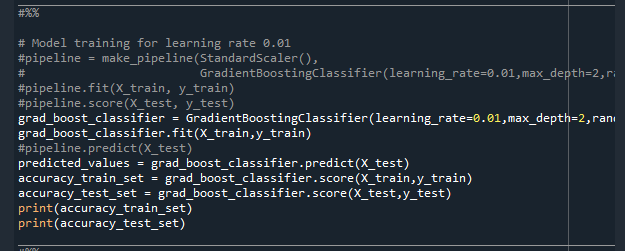
Here are the libraries that are required :

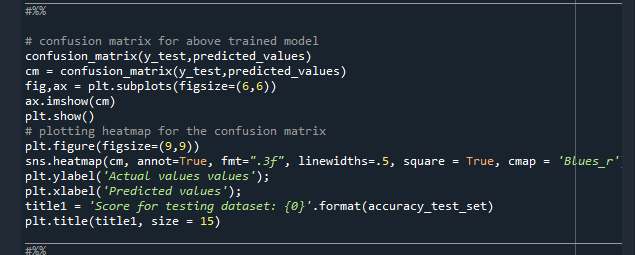


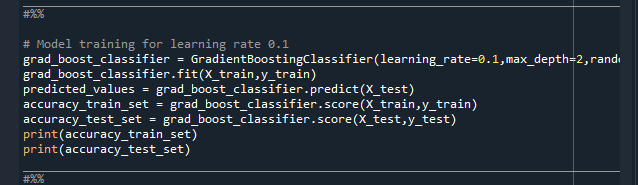
Then I read the whole data first without finding segments first just to test the model and check it:

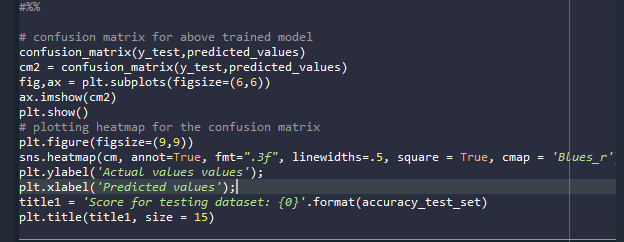


I tested model for learning rate 0.01 and 0.1 and found that accuracy is more for 0.1 so I decided to build 7 models on learning rate 0.1 for the segments of digit digits.



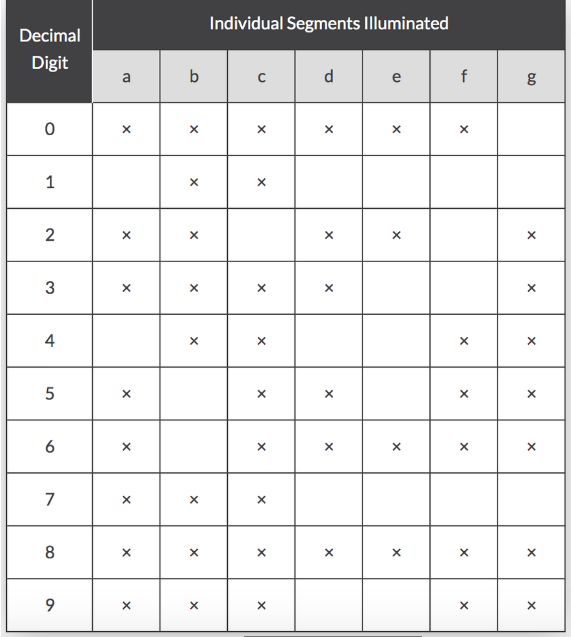




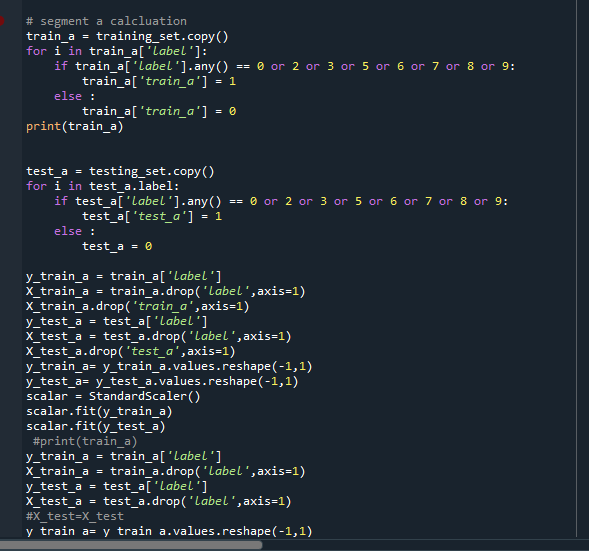


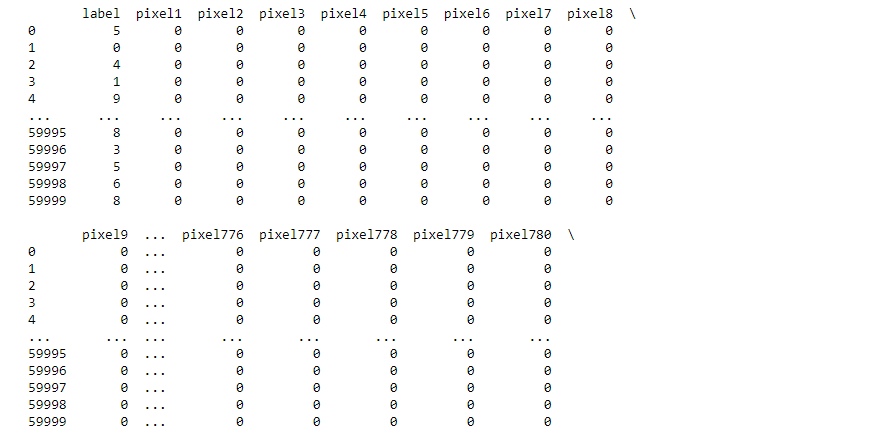
Now After testing the model I worked on preparing datasets for each segment separately for which I decided to find out values based on the activation pattern provided. According to this pattern I did following steps:

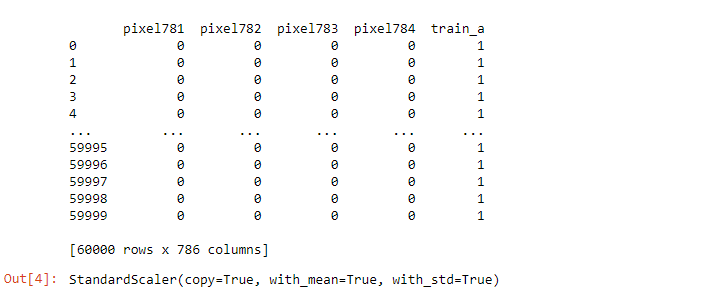
1. Created a column train\_a and test\_a for both the data sets separately
2. Iterate through train\_a and test\_a data set on the condition that if the values in the column label is either 0,2,3,5,6,7,8,9 then set the value as 1 in the column train\_a else set the value as 0
3. Then splitted the data set into target and dependent variable i.e X\_train\_a ,X\_test\_a, y\_train\_a and y\_test\_a.
4. Then build the model on top of it
5. Predicted values and find out the confusion matrix.

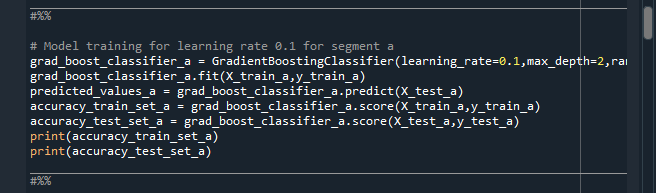


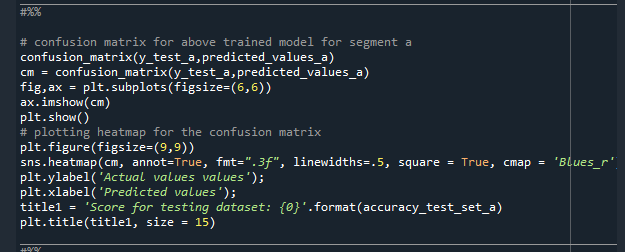
Here is the logic and output for all:









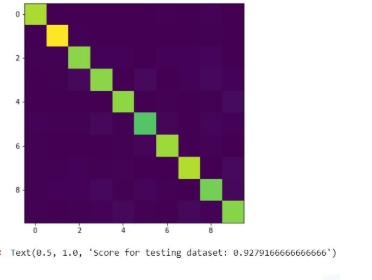


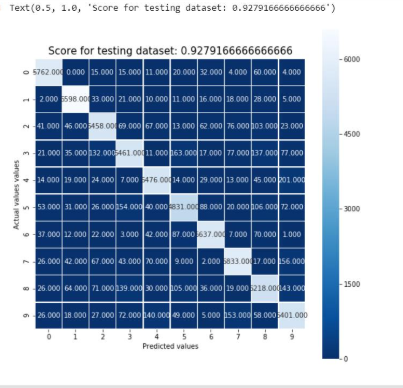
Same logic for b-g segments and find out the all the parameters for all segments.

**Conclusion**

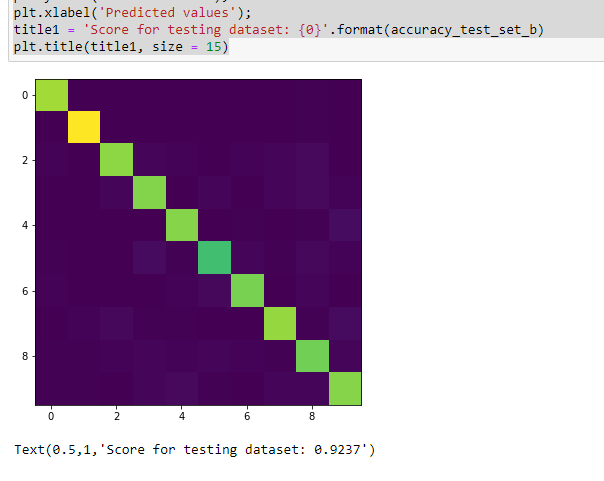
The confusion matrix and graphs for all the segments are nearly same and observed very small difference in the values of accuracy for all the segments. And the average accuracy is 92% for all the segments. Here are the results for each segment:

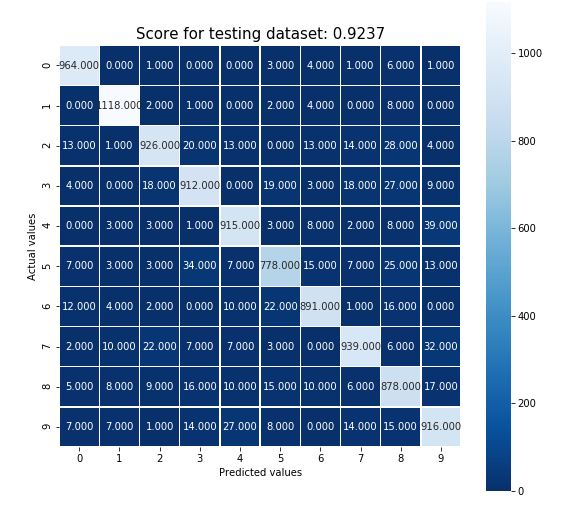
a.)



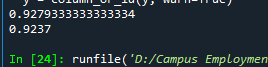


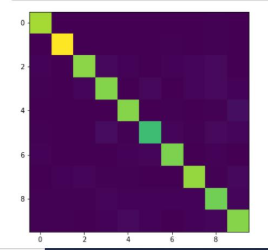
b.)

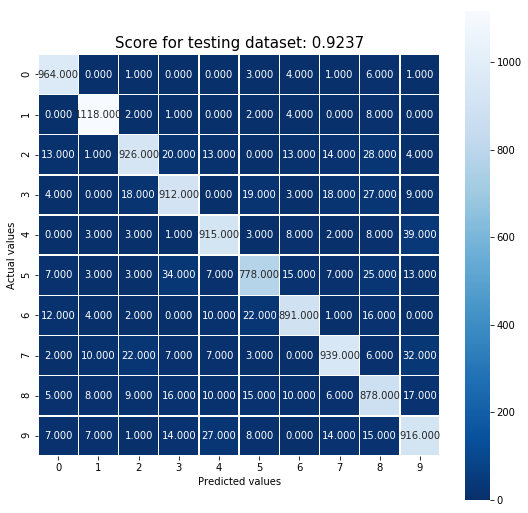




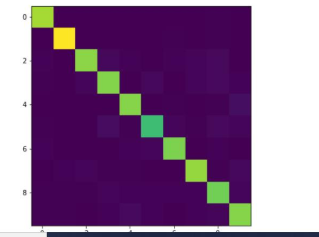
c.)

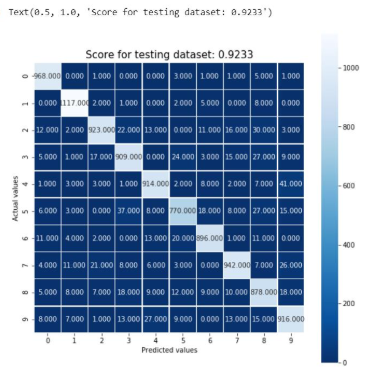




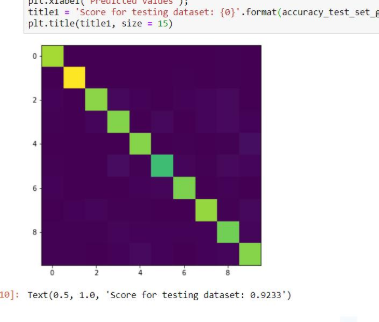


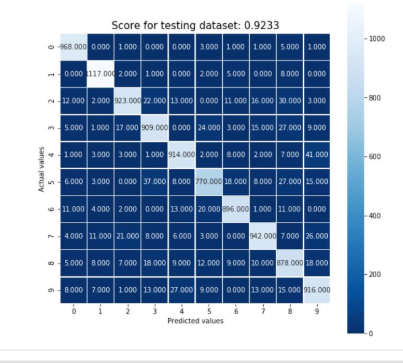
d.)





g.)





# References

1. Nsantos. (n.d.). Nsantos/raspberry-pi. Retrieved October 06, 2020, from https://github.com/nsantos/raspberry-pi/blob/41da3c3747e489baed82eef13600071d6d06f2ef/7-segment/7-segment-load-threaded.py
2. PrathameshDeshpande. (2020, June 20). PrathameshDeshpande/Minst\_kaggle. Retrieved October 06, 2020, from <https://github.com/PrathameshDeshpande/Minst_kaggle/blob/master/CNN_MNIST_K.py>
3. Dinesh. (2020, May 28). Transforming MNIST images into their corresponding seven segment display representation using... Retrieved October 06, 2020, from <https://medium.com/@deneshj19/transforming-mnist-images-into-their-corresponding-seven-segment-display-representation-using-82eb733eb20a>
4. Brownlee, J. (2020, June 30). Rescaling Data for Machine Learning in Python with Scikit-Learn. Retrieved October 06, 2020, from https://machinelearningmastery.com/rescaling-data-for-machine-learning-in-python-with-scikit-learn/
5. Brownlee, J. (2020, August 14). How to Configure the Gradient Boosting Algorithm. Retrieved October 06, 2020, from https://machinelearningmastery.com/configure-gradient-boosting-algorithm/
6. Aarshay JainAarshay graduated from MS in Data Science at Columbia University in 2017 and is currently an ML Engineer at Spotify New York. He works at an intersection or applied research and engineering while designing ML solutions to move product metrics. (2020, July 05). Gradient Boosting: Hyperparameter Tuning Python. Retrieved October 06, 2020, from https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/
7. Nelson, D. (n.d.). Gradient Boosting Classifiers in Python with Scikit-Learn. Retrieved October 06, 2020, from https://stackabuse.com/gradient-boosting-classifiers-in-python-with-scikit-learn/
8. Andrei, Rosebrock, A., Richard, James-Hung, Tham, Singh, P., . . . Wijoyo, A. (2020, April 18). Recognizing digits with OpenCV and Python. Retrieved October 06, 2020, from https://www.pyimagesearch.com/2017/02/13/recognizing-digits-with-opencv-and-python/
9. Administrator, Says, M., Shahmeer, M., Says, J., James, Says, R., . . . P, P. (2017, December 24). BCD to 7 Segment LED Display Decoder Circuit Diagram and Working. Retrieved October 06, 2020, from https://www.electronicshub.org/bcd-7-segment-led-display-decoder-circuit/
10. Brownlee, J. (2020, September 07). How to Develop a Gradient Boosting Machine Ensemble in Python. Retrieved October 06, 2020, from https://machinelearningmastery.com/gradient-boosting-machine-ensemble-in-python/
11. Dawson, C. (2019, September 26). Gradient Boosting in Python from Scratch. Retrieved October 06, 2020, from https://towardsdatascience.com/gradient-boosting-in-python-from-scratch-4a3d9077367
12. https://www.youtube.com/watch?v=M2yxdJR40fU