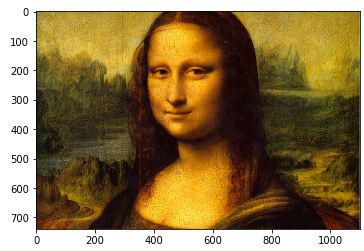
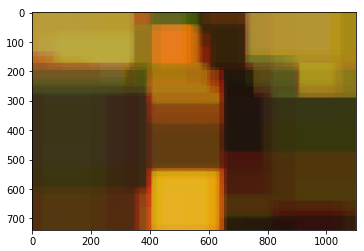
Ryan Farr – rlf238

November 25th, 2018

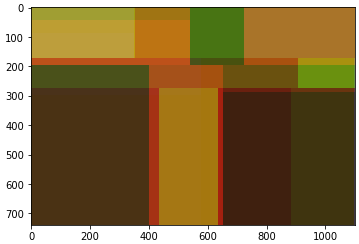
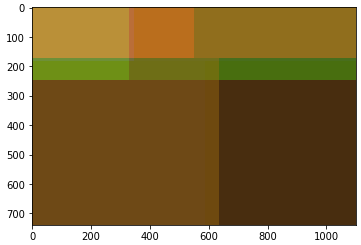
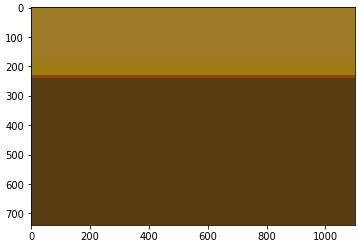
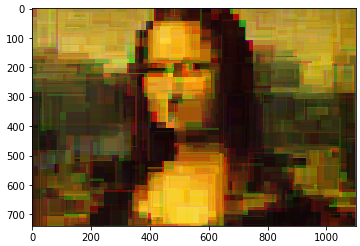
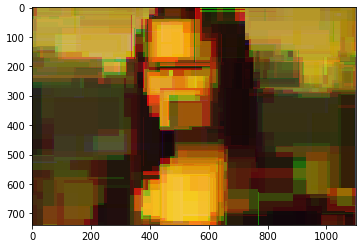
CS 5785 – Applied Machine Learning

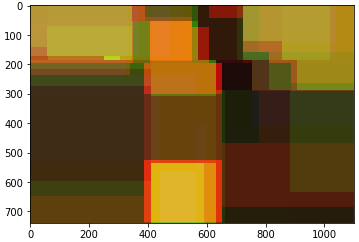
**Homework 4 Programming Exercises**

1. Approximating images with neural networks
   1. Asdf
2. Random forests for image approximation.
   1. I’ve chosen to work with the following Mona Lisa image.
   2. 5,000 points were randomly sampled from the original image. No preprocessing was deemed necessary; any subtraction or normalization only changes the value of the splitting points but not their relative position, which is to say that preprocessing these points shouldn’t change the output.
   3. To process the output, I’ve chosen to learn three separate functions, one for each color. I opted not to do grayscale because I wanted the output to look as similar to the input as possible. The pixel intensities already fell between 0 and 1 and no other processing was deemed necessary for similar reasons to part b: it shouldn’t affect the output.
   4. For random forest regression I’ve chosen to use sklearn’s RandomForestRegressor (documentation for which is cited in *main.ipynb*). Below is the output with a maximum depth of five for each of ten trees.



* 1. Experimentation
     1. Below are the results using a single decision tree and values of 1, 2, 3, 5, 10, and 15 for depth.





As can be seen, the depth heavily influences the result. With additional depth we see a larger number of colors used