Michael Baldwin

Josh Engelsma

Adam Terwilliger

January 19, 2016

CIS 678 – Machine Learning

Project 1

**Abstract**

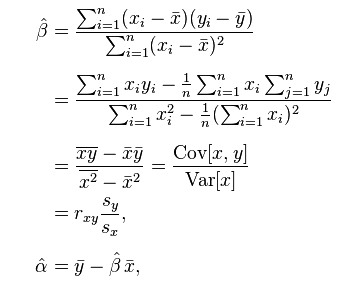
To demonstrate our programming abilities as an introductory project in CIS 678 – Machine Learning, we explored book sales data with basic regression analysis. Using Python, we were able to generate a simple linear regression model that predicted the amount of downloads using the amount of hours since the book was released. Additionally, we explored polynomial regression as we noted a second and third order model fits the model more appropriately. To visualize our results, we utilized matplotlib to bin the data by time of day, day of week, and day of month; as well as, D3.js to generate the scatterplot of the dataset with the three distinct lines representing unique order polynomial models.

**Implementation details**

Our backend program is written in Python 3.0 and frontend in Javascript/CSS with the D3 package/libraries installed. These programs were executed locally on each member’s respective Macbook Pro (2012).

**Summary of Problem**

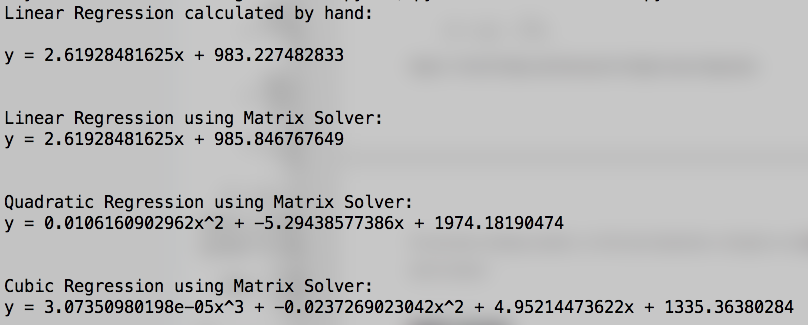
Linear regression is the foundational method when first understanding supervised learning. In supervised learning, we look to characterize a set of variables or model that can predict a known response variable. In this instance, we have just a single explanatory variable (simple linear regression), time in hours since the release date, and we are predicting amount of downloads as the response. A simple linear regression line can be represented with a slope and intercept, as seen in Figure 1.



**Figure 1. Proof of Slope and Intercept for Simple Linear Regression**

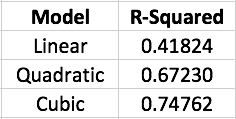
**Results**

We note that in looking to predict *y*, we first must characterize *x*, through its *covariance* with *y* and its *variance*. As such, we have calculated the slope and intercept using this method by hand, as well as, with a matrix solver in Python that allows for us to apply higher order polynomial regression models; seen in the sample output from Figure 2.



**Figure 2. Sample Output from Python Regression program**

We can observe the performance of our models through R2 values in Table 1, and visually in Figure 3. As we uncovered, the linear fit to the data explains only approximately 42% of the variation in amount of downloads. However, as it appears in Figure 3, we see that the quadratic and cubic fits to the data show great promise explaining approximately 67% and 75% of the variation in amount of downloads; respectively.



**Table 1. R-Squared values for First, Second, and Third order models**

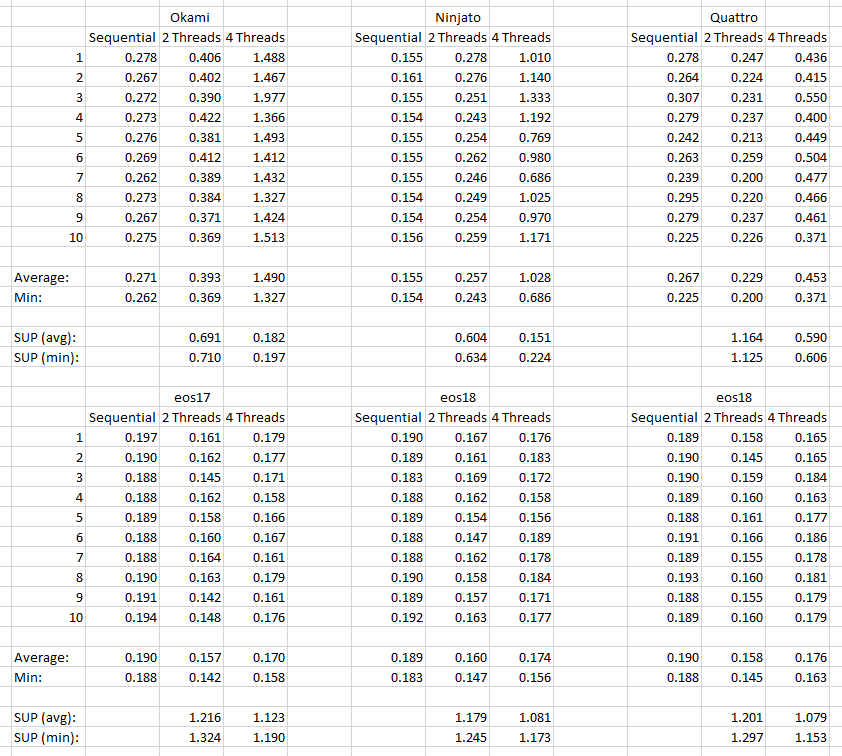
**Discussion**

Reasoning behind okami and ninjato not performing well would be a great area for future study. Additionally, we noted an attempt to get runtime for 8 threads; however, these trials were taking quite long (>30 seconds), so we did not continue with these measurements. To pinpoint the bottleneck in our mutual exclusion and conditional variable pthread code would be the biggest area of improvement. Once that is taken care of, runtime analyses on greater size threads (8, 12, 16, and 24) would be a valuable endeavor.

**Future Work**

**Credits**

**Appendix**

****

**Credits**

Efficient read of file c++ :

<http://stackoverflow.com/questions/2602013/read-whole-ascii-file-into-c-stdstring>

Dynamic allocation of 2D arrays:

<http://www.fredosaurus.com/notes-cpp/newdelete/50dynamalloc.html>

<http://stackoverflow.com/questions/14829105/2d-dynamic-memory-allocation-array-in-c>