Christopher Prats
MAP4191 Final Project

For the final project, I decided to analyze the first data set provided by the professor utilizing linear regression, ridge regression, and lasso regression. This data set has 2000 samples and 10000 predictors. To begin, I will describe some of the issues I faced when performing this task. The main issue I faced was performance problems relating to the size of the provided data. This led to analysis being a multi hour task on a high end, multi-threaded CPU (Ryzen 5 3600) with large amounts of RAM (32gb) and swap space (128gb). (Had my hardware been lower end, I would've been forced to choose a different data set due to the memory usage and calculation times involved.) This demonstrates why many real world applications of regression (such as for machine learning and artificial intelligence) is done utilizing GPU acceleration. Additionally, I faced difficulty utilizing this data set as provided and had to combine the data provided from importing the X and Y data sets into a single variable in order to properly create the required models. However, these complications did force me to learn many "tricks" in order to improve the performance of performing regression in R, such as creating additional workers to multi-thread training with the parallel and doParallel libraries.

To perform the calculations done in this project, I primarily utilized glmnetUtils, caret, and the built in R libraries; data was imported utilizing the read.table function and calculations were a mixture of built in libraries and imported libraries. Linear regression was done with the lm function, which is built into R. Both lasso and ridge regression where done utilizing the glmnet and cv.glmnet functions found in the glmnetUtils library. Training was done for all three types of regression utilizing the train function found in the caret library. This function had built in methods for lm, ridge, and lasso in order to perform training utilizing a defined trainControl. I set my trainControl to utilize k-fold cross validation with 5 folds. Additionally, the trainControl allows you to enable parallelization, which I found necessary in order to achieve reasonable run times with my R script. (My first run prior to

parallelization took over 12 hours, until I realized I was only utilizing 1 CPU thread and researched methods to parallelize training. This reduced the total run time to a measly 3 hours.)

The training results demonstrated that traditional linear regression utilizing the lm function performed the worst, while lasso regression performed the best for this model. Ridge regression was consistently in the middle of the 2 other types of regression. The last training run of the script prior to writing this report yielded the following values:

Regression Type	RMSE (Lower is better)	Rsquared (Higher is better)	MAE (Lower is better)
Linear Regression	3459.927	0.00231	2781.114
Ridge Regression (Best Tuning parameter)	409.863	0.00504	262.725
Lasso Regression (Best tuning parameter)	44.386	0.00847	35.455

This table demonstrates that lasso regression provides the best results. Additionally, it did not take much longer than ridge regression (around 5 times the time, but both ridge and lasso regression were much slower than linear regression). A difference between linear, ridge, and lasso regression is that ridge regression attempts to minimize coefficients to improve the model, whereas lasso regression attempts to set some coefficients to 0 in order to improve the model (and regular linear regression uses coefficients as is, no regularization or elimination). This leads to linear and ridge regression having a full 10,000 coefficients (equal to the number of predictors), whereas lasso regression can have a maximum number of coefficients equal to the number of samples (a theoretical maximum of 2000 non-zero coefficients in a model of this size). In this case, lasso had 75 non-zero coefficients. In other words, lasso found the most efficient model to be one with only 75 non-zero coefficients. Accounting for the fact that lasso was the most efficient model overall, the 75 predictors corresponding to these coefficients have the strongest relationship with the response, Y.

In summary, lasso regression was the best model for this data set. It provided the best results overall at a reasonable increase in computational complexity. It was slower than the other types of regression, but provided a justifiable increase in R^2 and decrease in RMSE and MAE. I found this project to be a great learning experience in utilizing and analyzing large datasets utilizing the R programming language.