REPORT

Part 1

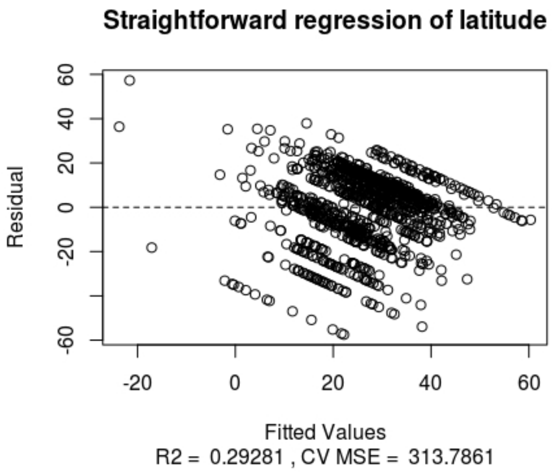
1. *Linear regression with various regularizers The UCI Machine Learning dataset repository hosts a dataset giving features of music, and the latitude and longitude from which that music originates here. Investigate methods to predict latitude and longitude from these features, as below. There are actually two versions of this dataset. Either one is OK by me, but I think you'll find the one with more independent variables more interesting. You should ignore outliers (by this I mean you should ignore the whole question; do not try to deal with them). You should regard latitude and longitude as entirely independent.*

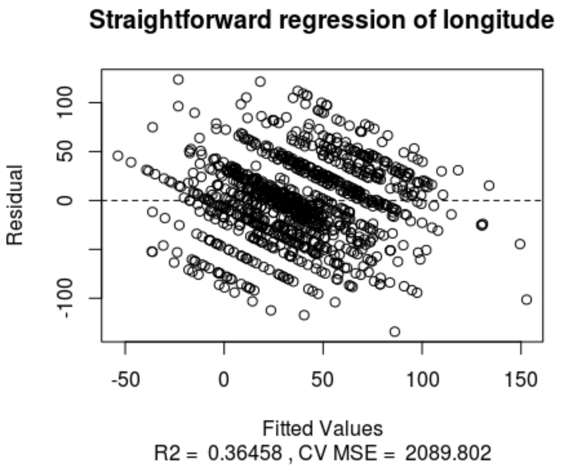
a) First, build a straightforward linear regression of latitude (resp. longitude) against features. What is the R-squared? Plot a graph evaluating each regression

I used the larger dataset with more variables. There have been multiple variations in numerous Piazza posts regarding how certain portions of this assignment should be interpreted. I tried to account for all possible recommendations and hints announced on Piazza. My code runs without errors on my machine.

Both graphs evaluating each “straightforward” regression (one for latitude and one for longitude) are shown below with the R-squared and cross-validated MSE indicated directly on the graphs. After lm(), I also did a 100-fold cross-validation using cv.lm() in order to be able to compare the results with the regularized models presented further in this report. The more folds I did, the smaller MSE became. I decided to stop at 100 as this indicated a reasonable decrease in MSE (you might want to make this smaller for the code to run faster).

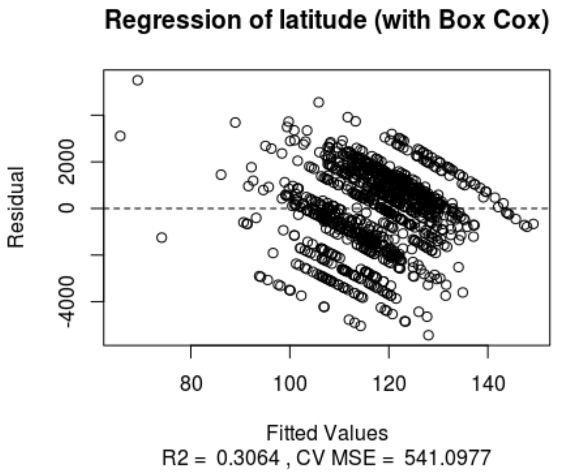
As there were too many changing requirements to watch for on Piazza, I did not have enough time to estimate the R-squared shrinkage based on the cross-validation, but, according to the materials available online, R-squared usually shrinks at a modest rate and, considering that both R-squared values are very close, they would most probably shrink similarly. As an alternative metric, I also used RMSE based on the train function from the caret package.

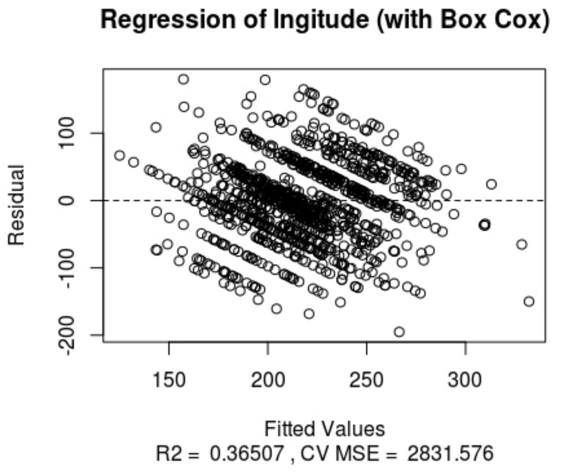




b) Does a Box-Cox transformation improve the regressions? Notice that the dependent variable has some negative values, which Box-Cox doesn't like. You can deal with this by remembering that these are angles, so you get to choose the origin. why do you say so? For the rest of the exercise, use the transformation if it does improve things, otherwise, use the raw data

To avoid negative values I converted latitude by adding 90 (as its range is -90:90) and longitude by adding 180 (as its range is -180:180, although 90 could have been enough in this particular case because the min(longitude) is more than -90). To be consistent, I used the same approach as in a) including the 100-fold cross-validation and MSE/RMSE estimation. I applied the Box Cox transformation to y using the boxcox() function, then I ran linear regression, after which I applied the reverse Box Cox transformation to the regression's fitted values in order to bring them back to the original coordinates (otherwise the values were around 7 digits). Both graphs with the same output parameters are shown below.

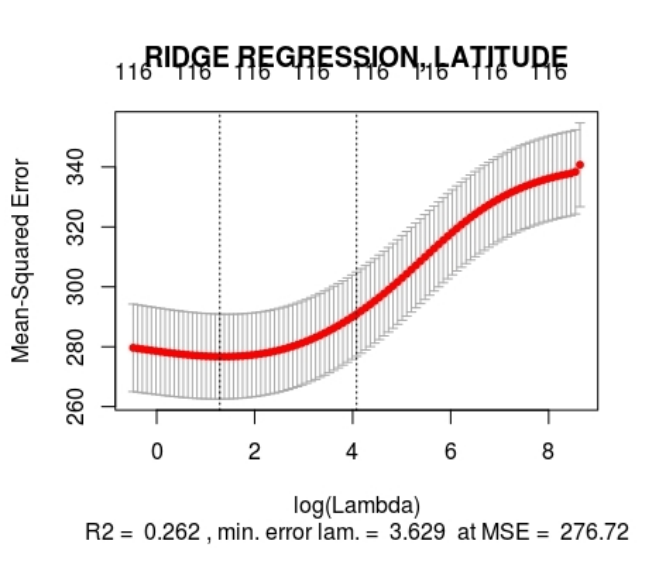


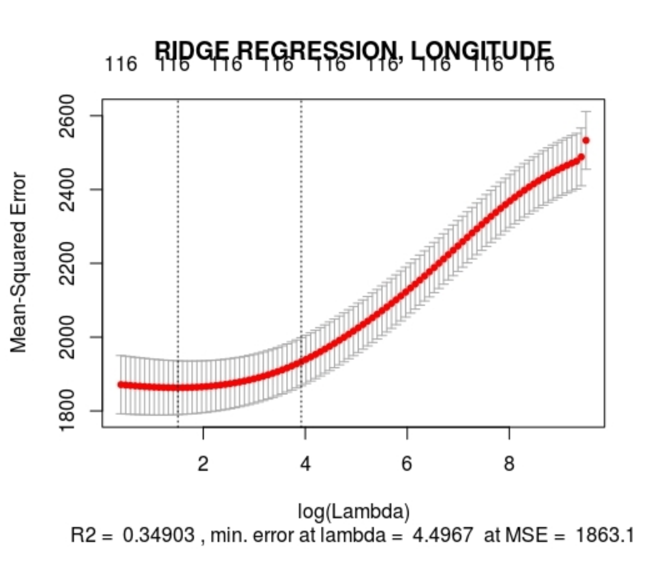


We can see that, after the Box Cox transformation, there is a very marginal improvement in R-squared in both cases (0.2928 vs. 0.3064 for latitude, and 0.3646 vs. 0.36507 for longitude), the distribution of residuals doesn't change much, but there is a significant increase in the MSE value (314 vs. 541 for longitude, and 2090 vs. 2832 for longitude). Because of this I made a conclusion that, evidently, the Box Cox transformation does not improve the regression and used the plain regression results for the Ridge, Lasso, and Elastic Net regularized regressions.

The conclusion is somewhat confirmed by running the train function from the caret package to examine RMSE values for these four regressions. The results for the regressions of latitude are as follows: 16.5 without Box Cox and 16.4 with it; for longitude: 42.8 without Box Cox and 42.6 with it. The difference is insignificant, and subsequent runs of the program produce slightly different results around the same values which confirms my conclusion that Box Cox does not really improve the regression.

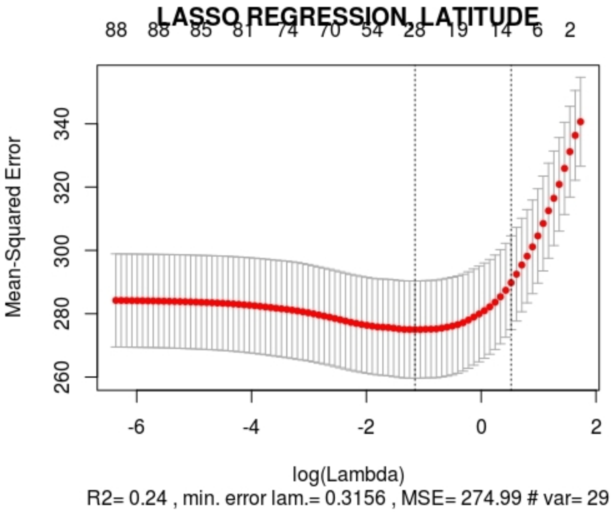
c) Use glmnet to produce:
A regression regularized by L2 (equivalently, a ridge regression). You should estimate the regularization coefficient that produces the minimum error. Is the regularized regression better than the unregularized regression?

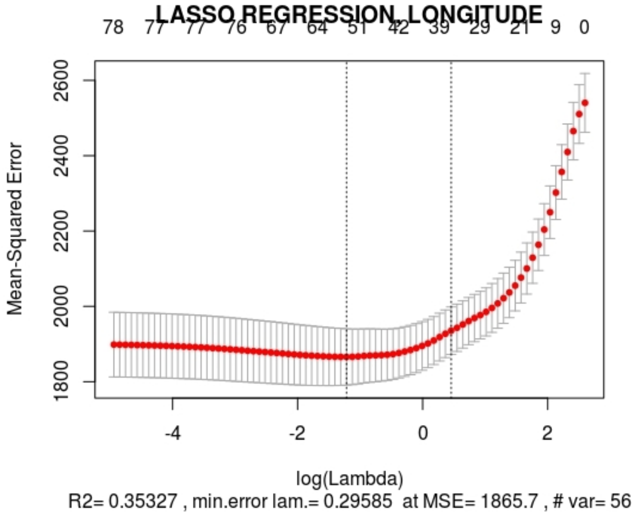




The regularization coefficient that produces the minimum error is shown on each plot along with the respective MSE. Compared to the plain unregularized regression, there is a 10% drop in R2 and a 12% drop in MSE for latitude, and a 9.5% drop in R2 and a 9% drop in MSE for longitude. According to Clay Ford and prof. Cosma Shalizi (http://data.library.virginia.edu/is-r-squared-useless/), “R-squared does not measure goodness of fit. It can be arbitrarily low when the model is completely correct.” This and other statements in this article made me vote for MSE as a better measure for goodness of a model. Thus, this is a better regression than the unregularized one.

A regression regularized by L1 (equivalently, a lasso regression). You should estimate the regularization coefficient that produces the minimum error. How many variables are used by this regression? Is the regularized regression better than the unregularized regression?

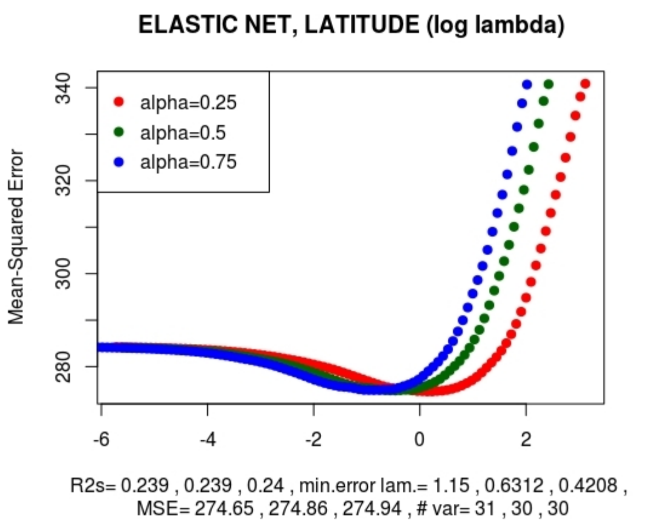


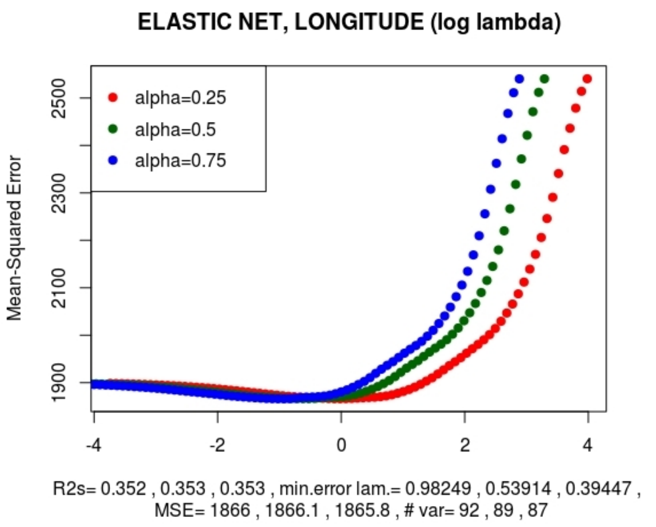


The regularization coefficient that produces the minimum error, MSE, and the number of variables used by the regressions are shown on each plot (the number of variables was obtained from the coef() object, but it is also confirmed by the first vertical line in each case). Here, we have a greater decrease in R2 from the regularized value for latitude, and a smaller one for longitude compared to the Ridge regression, and the changes in the MSE are also oppositely directed. The difference from the Ridge regression is insignificant, so I can say that these results are in line with Ridge and better than the regularized regression. It should be noted however that the Lasso regression of latitude uses has only 29 variables which means that this is a much simpler model than the Ridge regression latitude. The same is true for longitude with only 56 variables compared to 116 in Ridge.

A regression regularized by elastic net (equivalently, a regression regularized by a convex combination of L1 and L2). Try three values of alpha, the weight setting how big L1 and L2 are. You should estimate the regularization coefficient that produces the minimum error. How many variables are used by this regression? Is the regularized regression better than the unregularized regression?

The three values of alpha I tried were 0.25, 0.5, and 0.75. The R2s, regularization coefficients producing the minimum error, MSEs, and the number of variables are shown on each plot below in the increasing order of alpha, respectively (meaning the first number corresponds to alpha = 0.25, etc.)





These plots are somewhat in line with the two previous regressions. The R2s for latitude are slightly smaller than for the Ridge and Lasso regressions while MSE is almost the same as for Lasso. For longitude, they stay the same as Lasso and slightly higher than Ridge. The difference in MSEs is also slightly noticeable compared to Ridge and Lasso. The number of variable is also small for latitude, but significantly larger for longitude compared to Lasso.

On a side note, there is an interesting observation about the regularization coefficients that are the smallest for the Lasso regressions (alpha = 1), the largest for the Ridge regression (alpha = 0), and are in the middle for the elastic net regression which is, probably, logical as they may be correlated with the alpha parameter.

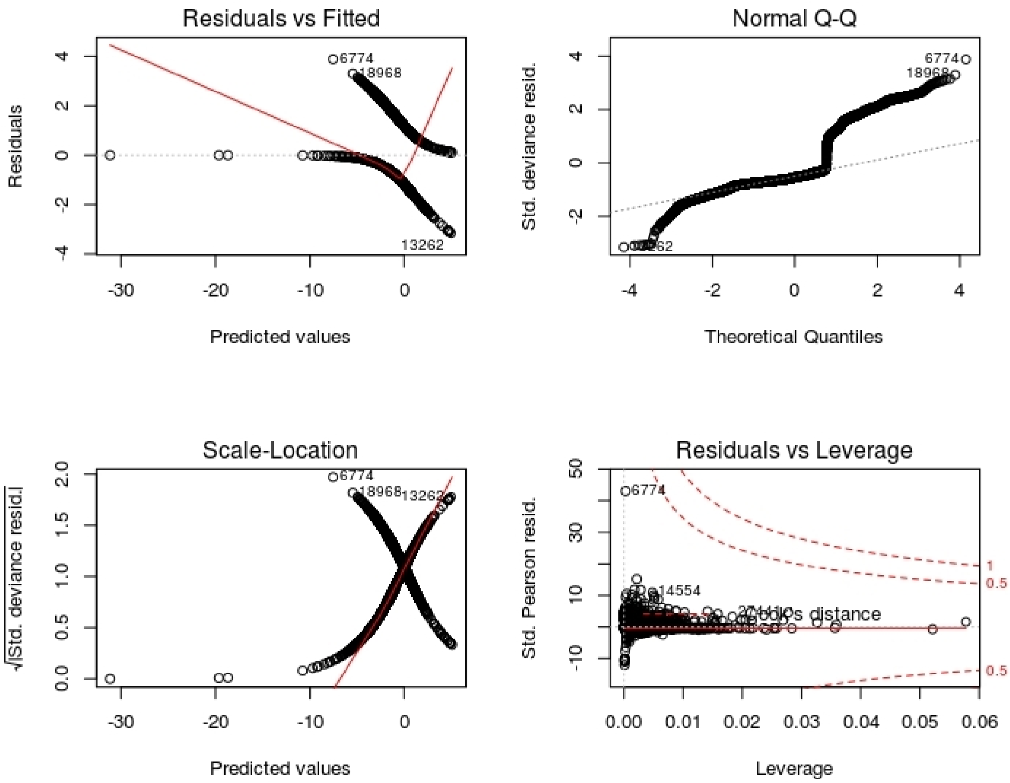
To summarize the results obtained, I can say that out of the seven regressions for each of latitude and longitude the best model is the Lasso regression for both latitude and longitude because it has smaller MSE compared to the unregularized regressions and a considerably smaller number of variable participating in the regression compared to Ridge and Elastic Net regressions, which makes it a simpler model both for latitude and longitude, while the rest of its parameters (R2 and MSE) are insignificantly different from Ridge and Elastic Net. The best lambdas, according the Lasso model, are 0.316 for latitude and 0.296 for longitude.

For your information, I also included the code for the misclassification error for the regularized models in order to compare them from this perspective (running cv.glmnet with family=”binomial”, and type.measure=”class”), however it takes a huge amount of time to run on my machine which prevented me from properly analyzing and reporting the results. This is something to work on in the future.

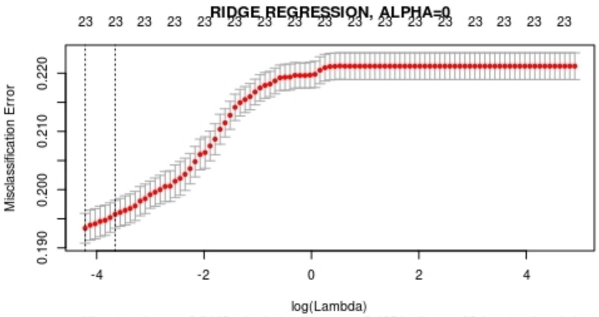
Part 2

2. Logistic regression. The UCI Machine Learning dataset repository hosts a dataset giving whether a Taiwanese credit card user defaults against a variety of features here*. Use logistic regression to predict whether the user defaults. You should ignore outliers, but you should try the various regularization schemes we have discussed.*

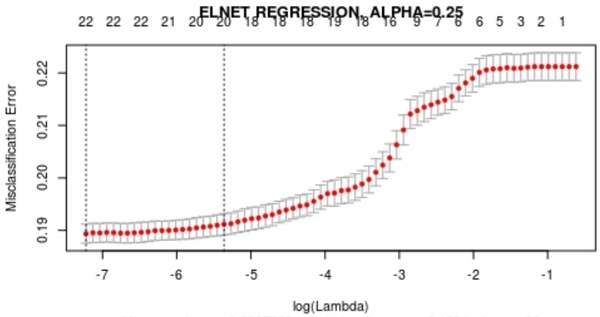
The plots for the unregularized generalized linear model are shown below. Deviance of this regression is 27877, misclassification error = 0.195, and all 23 variables were used according to the regression formula.



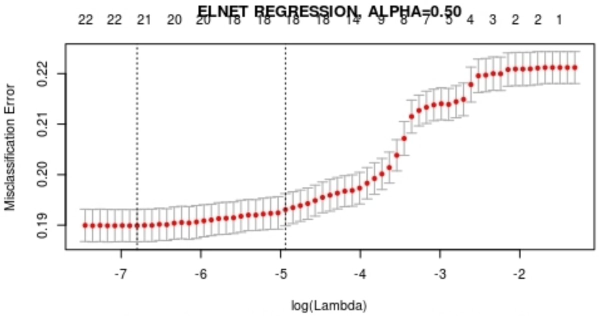
The plots for the regularized Ridge, Lasso, and Elastic Net regressions are presented below. They contain information about the best lambda (regularization coefficient producing the minimum error), misclassification error at this lambda, which in all cases is better than for the unregularized regression, and the number of variables, which is 23 for Ridge, and 22 for Lasso and Elastic Net and was computed numerically in the code and confirmed by the information on the plots.



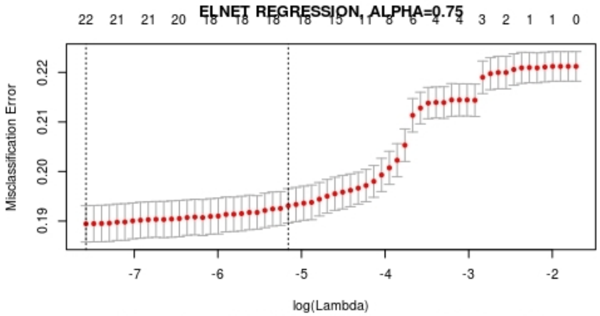
Min. error lam. = 0.0148 at misclassif. error = 0.1934 , # var = 23 (see on the plot)



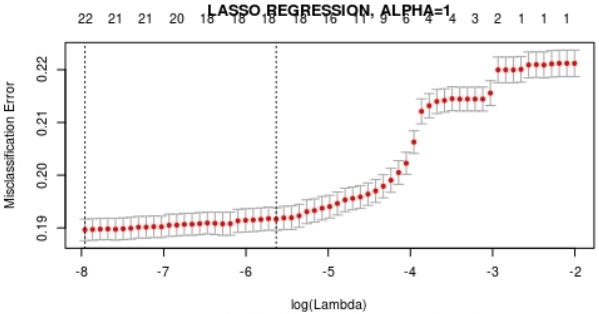
Min. error lam. = 0.0007296 at misclassif. error = 0.1894 , # var = 22



Min. error lam. = 0.001114 at misclassif. error = 0.1899 , # var = 21

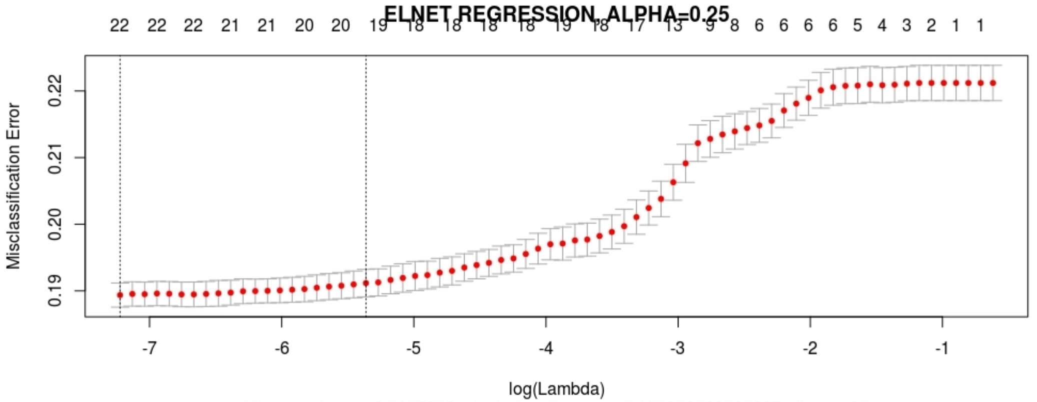


Min. error lam. = 0.0005119 at misclassif. error = 0.1894 , # var = 22

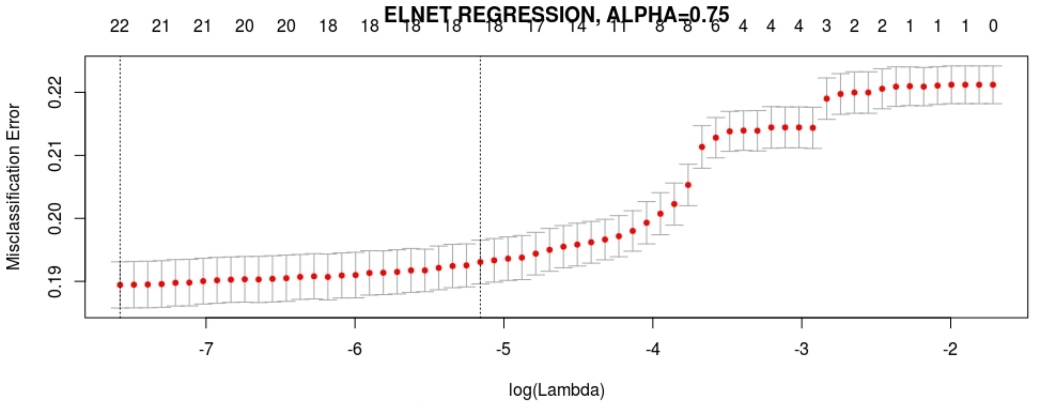


Min. error lam. = 0.0003498 at misclassif. error = 0.1896 , # var = 22

The number of variables is practically the same, so it cannot be used to decide which regression is better. The misclassification error is better for the regularized regressions, being the least for alpha=0.25 and alpha=0.75 in the Elastic Net regression. The numbers on the plots above are rounded. Let us have a closer look at the two plots in question with unrounded values for the misclassification error:



Min. error lam. = 0.0007296 at misclassif. error = 0.189366666666667 , # var = 22



Min. error lam. = 0.0005119 at misclassif. error = 0.189433333333333 , # var = 22

The winner is the Elastic Net regression with alpha=0.25 as it has the smallest misclassification error. Thus, the best lambda is 0.0007296.

My code also contains two other measures of the regularized regressions which can be obtained by using parameters type.measure = “deviance” for binomial deviance and type.measure = “auc” for AUC (area under the curve). I did not review those for the interest of time and because the information already available is sufficient to make a decision. But they can be certainly used for model selection as additional measures in a real-world situation when many things may be at stake.