Lab #5 - FMRI Imaging

Todd Faulkenberry

11/24/2018

Introduction

This paper builds on a study conducted by Kendrick N. Kay et. al., and published in the magazine Nature in March 2008. The paper, entitled "Identifying natural images from human brain activity," explains the authors' use of a decoding method that analyzes functional Magnetic Resonance Imaging (fMRI) data of a test subject in order to identify a specific image seen by that subject while the fMRI was collected. The implications of a such a finding are stunning: With a well-constructed model, scientists may be able use data measuring human brain activity to recreate the human visual experience.

In this paper, I construct a LASSO regression model for each of the twenty units (voxels) under consideration. The Rsquared of my final models lack the type of predictive power for which I hoped; this fact is also reflected in the correlation between my fitted values and the observed values. Unfortunately, my models do not have much success in being able to predict voxel activity based on the predictors most highly correlated with each voxel. As the ensuing analysis of two specific voxels show, almost all of my models responded similarly to the L1 penalty that defines LASSO. The actual predictors themselves, however, vary quite significantly, with some voxels having their predictors consolidated within certain ranges and others having them spread out across the entire index of predictors. The differences in these two voxels suggest some paths forward were I to revisit this data.

The Data

The fMRI data used in the study was collected from the Primary Visual Cortex (V1), the earliest, simplest, and most studied visual area of the brain. In humans, it is located at the back of brain. Because of its foundational use in the brain and the role it play its pattern recognition, the primary visual cortex should be an area with a lot of activity when an observer is shown an image.

The data provided looks at 20 voxels - three-dimensional cube-like units - in the V1. For each voxel, we have its fMRI reading for each of the 1750 images distilled into a single value (meaning a 1750 X 20 data frame.) Additionally, for each of those 1750 images, we have a measurement of 10921 variables called gabor features. Gabor features represent the initial image by breaking down its feature into different parts and measures the tuning and frequency of these different parts across space and orientation. In this project, we are tasked with using the gabor features to predict the brain's response for future images at the aforementioned 20 voxels.

This data presents a handful of interesting problems that must be determine before we begin modeling. First, and simplest, is loading the data so that it is ready for transformation and analysis. In order to do this, I converted the four files provided in our .RData file into data frames. I then joined the aforementioned two data frames into one data frame so we can compare the relationships between the gabor features and the fMRI readings at each voxel.

Preprocessing

Preprocessing is a necessity when dealing with predictor-heavy data because it also us to do some standardization and elimination of predictors that will ultimately not be useful for creating a robust model. First, given the large amount of predictors, we must ask ourselves if there are predictors worth eliminating. How should we determine what to eliminate? First, we look at near zero variance predictors. It is important to eliminate zero and near zero variance predictors because their presence will actually keep us from constructing a functional model. This method found 512 near zero variance predictors out of the 10921 provided. We

eliminate these, bringing down our number of total predictors to 10409. Ideally, we would not have to eliminate these predictors, but it's a necessity given the role variance plays in the regressions we run below.

Even after eliminating the near-zero variance predictors, however, I was still unable to run LASSO regression. In order to compensate for this - and in order to make running these tests on my computer computationally feasible - I further filtered predictors so that the remaining predictors had a correlation of plus-or-minus 0.2 with the voxel in question. For most voxels, this pre-determined threshold filtered out many of the predictors, leaving each individual voxel with roughly 1,000 predictors. For certain voxels (10, 11, 13, 14, 16, 17, 19), this threshold eliminated too many predictors. For those, I lowered the threshold to 0.1, and for one voxel (20) I lowered the threshold to 0.05. In addition to mitigating the predictors that were preventing regressions from running, I also centered and scaled the remaining variables.

Subsetting the Data

Our next problem is subsetting the data into training, test, and validation set. Luckily, we were provided a natural validation set of 120 images. These images were used by the original study as a validation set for their modeling, and we will do the same here. This data will only be used at the end to assess our final model(s).

Next, we must consider how to divide the rest of the data into training and test sets. Because we're using these two data sets to refine our model selection, we have multiple considerations. First, we must consider the state of our data. We have relative few observations, and given the amount of data collected from the study, we have many, many more predictors than observations. Second, we must consider what we're measuring: The fMRI at different voxels. Given that some voxels may be wildly different from others, it may not make sense to fit a general model over the entire data set.

Given the limited amount of observations are the need to predict for each voxel, I decided to split my remaining data 80/20 between training and test data. Additionally, to maximize our training set, I then conducted 10-fold cross validation for each of the 20 voxels. CV presents a handful of benefits. Primarily, it allows us to construct abetter model. By dividing into 10 folds, we're maximizing our available data and stabilizing our results by taking an average over multiple fits. Averaging our results across the cross-validations should provide us with a less biased estimator (though it may introduce variance.) Additionally, running CV for each individual voxel should lead to more accurate results for each voxel instead of a more general model that may not capture each voxel's nuance.

LASSO Regression and Results

Now we have done the preprocessing necessary to analyze the data, the next step is thinking about which model to apply and considering the costs and benefits to each model. The pressing problem with the data from the fMRI imaging is that is our p is much greater than our n - in other words, we have many more predictors than we do observations. This is problematic as the amount of predictors drives our variance to astronomical levels. Additionally, the fact that many of our predictors appear to be highly correlated should drive variance up even further. In order to keep our variance in check, we can use regularization. Regularizing our model means we will take steps to reduce variance, namely by introducing some bias. Given the trade off between variance and bias, introducing bias directly leads to a reduction in variance. In the end, this creates a model with stronger predictive ability than we would have otherwise.

For these reasons, I used LASSO regression to develop a model for this data. LASSO stands for Least Absolute Shrinkage and Selection Operator. LASSO decreases model complexity through the parameter Lambda. Lambda functions as a penalization for model complexity: As lambda increases, the more penalty we face for having variance, i.e. the more penalty we face for having more predictors. One of the benefits of LASSO compared to similar regression methods (e.g. Ridge Regression) is that LASSO pushes most coefficients to zero because of how its penalization technique works. This makes the model more interpretable by driving the vast majority of coefficients to zero, while ridge regression will rarely if ever push the vast majority of coefficients to zero.

To run LASSO, I used the caret package. I trained each model using the train() function on the training set. These computations were rather cumbersome on my own computer, so I ran them once and saved their results (which is in my local data folder but which I did not push to Github) and then manually added them to the report.

Voxel	Lambda	RMSE	Rsquared	MAE
1	0.1	0.9688	0.1426	0.7551
2	0.1	0.9195	0.2332	0.7297
3	0.1	0.9510	0.1782	0.7536
4	0.1	0.9599	0.1879	0.7615
5	0.1	0.9426	0.1922	0.7527
6	0.1	0.9408	0.1716	0.7462
7	0.1	0.9050	0.2519	0.7239
8	0.1	0.9526	0.1701	0.7548
9	0.1	0.9054	0.2648	0.7198
10	0.1	0.9963	0.0327	0.7864
11	0.1	0.9934	0.0191	0.7943
12	0.1	0.9594	0.1604	0.7642
13	0.1	0.9991	NaN	0.7867
14	0.1	0.9971	NaN	0.8014
15	0.1	0.9459	0.1879	0.7535
16	0.1	0.9865	NaN	0.7793
17	0.1	0.9909	0.0326	0.7871
18	0.0032	0.9272	0.1950	0.7336
19	0.1	0.9897	0.0571	0.7856
20	0.1	0.9953	NaN	0.7932

As you can see, these models performed rather poorly, with the highest Rsquared being voxel 9 with 0.2648. Four models didn't even report an Rsquared, showing that my process was faulty. Unfortunately, I did not have the time or the technical ability to diagnose and correct the problems evident from my results. To be honest, I'm not sure why the models performed so poorly even though the analysis was limited to the predictors with the highest correlation. I suspect I did something wrong with my preprocessing.

Correlation between obsevered and fitted values

Below is a table that shows the correlation between my fitted model and the observed values of my validation set. Because I incorrectly set up my initial setup and ran out of time, some correlations below are not reported. The same voxels are not reported below are the same voxels for which I had to adjust my initial correlation measures. As I mentioned above, this suggests that my preprocessing earlier had a mistake that inhibited the correct calculation of my initial LASSO. Naturally, this problem also extends to my ability to report correlation as well.

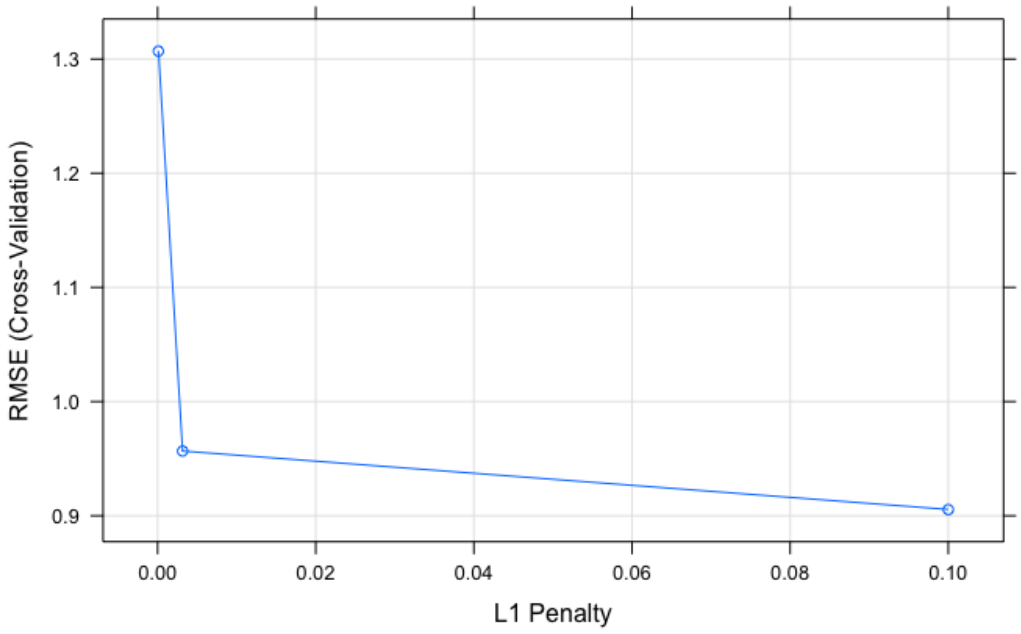
Voxel	Observed/Fitted Correlation
1	0.4379427
2	0.5053798
3	0.4596547
4	0.4526482
5	0.4480828
6	0.4480164
7	0.5310728
8	0.4623049

Voxel	Observed/Fitted Correlation
9	0.5361995
10	-
11	-
12	0.4161911
13	-
14	-
15	0.4802229
16	0.1859258
17	-
18	0.5338153
19	0.2823724
20	-

Notwithstanding the unreported correlations, many of the correlations seen here are above 0.4, with a handful above 0.5. Given the relationship between correlation and Rsquared, these results reflect more or less the same relationship we witnessed with our Rsquared above. If I had not ran out of time, it would have been wise for me to restart with new models and try to figure out what was limiting these models' predictive power.

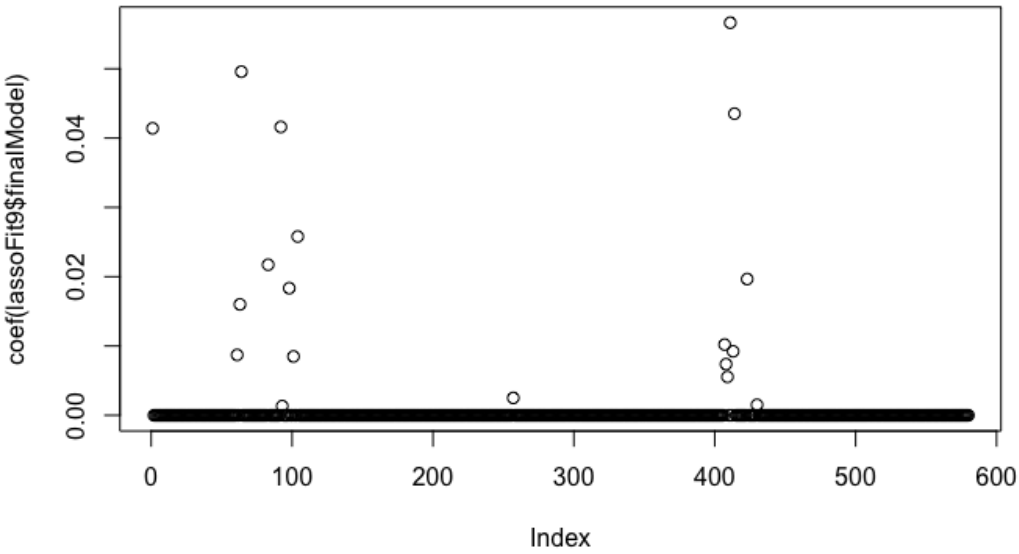
Diagnostics of Voxel 9

For my diagnostics section, I chose to focus on Voxel 9 because it had the highest Rsquared of the models I constructed and considered. First, I analyzed the L1 penalty, or alpha, to see how it reacted during cross-validation.



As we see very clearly, the RMSE for Voxel 9 decreases as we increase our L1 up until a value of 0.1. Interestingly, most of the drop in RMSE happens almost immediately, with the difference between an L1 of roughly .003 and .01 less than .01 RMSE. It's unclear to me why the drop was so sudden and then marginal

the rest of the way. To investigate why this may be the case, I then looked at the values of the coefficients to see if I could discern any meaning from their values.

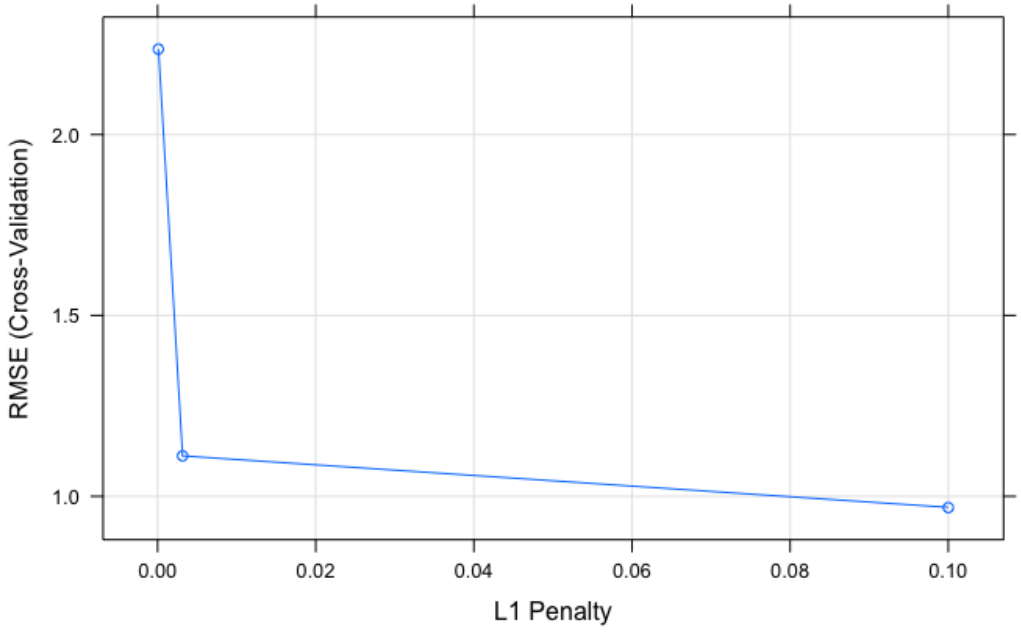


Looking at the values for the coefficients from LASSO, we see that the predictors for Voxel 9 are mostly clustered around two index ranges (~100 and ~400.) While some the values are clearly larger than others, I would not consider any of them outliers as there are a handful of predictors within both ranges that have significant values. In total, there appears to be roughly 20 predictors that have values greater than zero. While this could seem like very few given the large amount of predictors with which we started, I would consider it a decent amount given the incentive inherent in LASSO to drive as many predictors as possible to exactly zero.

Comparison and Interpretation of Voxel 1 and Voxel 9

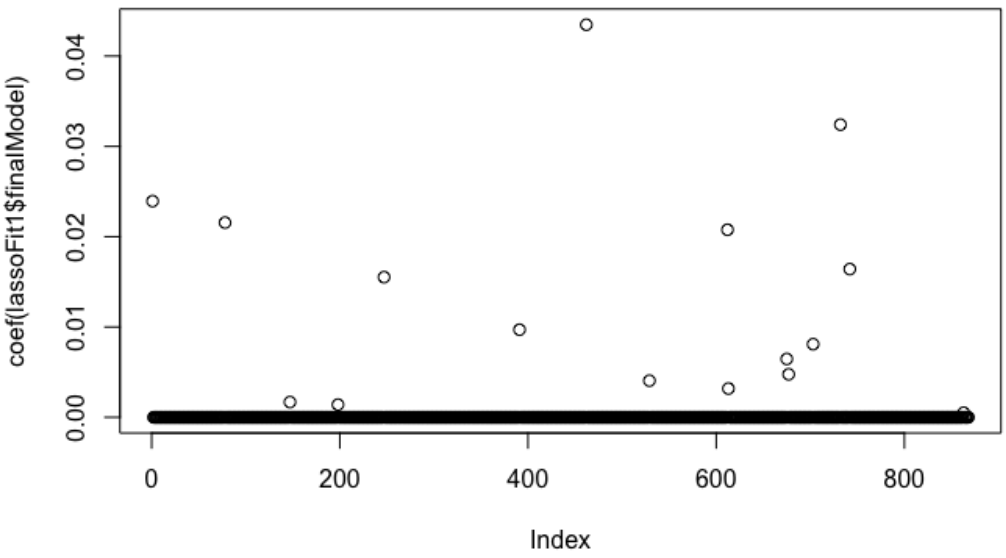
To extend my analysis, I now bring in Voxel 1 to compare its performance to Voxel 9. While I chose Voxel 9 specifically because it had performed the best of my models, I chose Voxel 1 because of its mediocre performance. While a handful of voxel models performed worse than Voxel 1's model, there were also plenty of models whose performance exceeded that of the model. Moreover, the poorly performing models I believe may have been performing poorly due to human error. As such, by comparing an average model with Voxel 9 instead of one my best or one of my worst models, I hoped to better understand what was going on with an "average" model that was performing but not up to the best standards I could meet. Through this, I hoped to see if I could uncover why better-performing models were outperforming it.

In order to do this analysis, I looked at the same two graphs I looked at with regard to Voxel 9. First, I again looked at the L1 penalty:



As is immediately obvious, The RMSE for Voxel 1 acts very similarly to the RMSE for Voxel 9. The only major difference is the amount of RMSE, with the RMSE for Voxel 1 slightly above 1, while the RMSE for Voxel 9 was slightly below 1. Though I did not include the graphs here, this exact same pattern – with a step drop and then a marginal decrease in RMSE as the L1 get closer to 0.1 – in fact happened across many different voxels, not just Voxels 1 and 9. Had I had additional time, I would have first tried to better understand this L1 and why it acts so homogeneously across all different types of voxels and voxel models.

Even though I did not discern why the L1s acted so similarly, I continuted the analysis by looking at the same distribution of coefficients:



Unlike Voxel 9, Voxel 1's predictors have no discernible concentration at at any index range. They are more evenly distributed across the entire range, from zero all the way to above 800. Additionally, I think you could say that some of the predictors for Voxel 1 are outliers (specifically the one value above 0.04) because the vast majority of non-zero predictor values are much lower. Voxel 1 does appear, however, to have roughly the same number of predictors as Voxel 9. This dichotomy between the clustering of values for Voxel 9 and the lack of clustering for Voxel 1 suggest that the worst-performing models are the ones that didn't find any connection or relationship between these groups of predictors. In models like those for Voxel 9, we see that the training found groups of predictors and weighted them similarly, leading to more accurate predictions. With the model for Voxel 1, we see no recognition of similar predictors that deserve similar weights. If I were building a model from the beginning, I would try to incorporate the correlation of certain predictors with other predictors and/or voxels with other voxels into my model. This would have allowed me to build upon pre-existing relationships to construct models with better predictive ability.

Conclusion

In this paper, I used the LASSO to build a model to predict voxel response from a large group of predictors. While my models were ultimately pretty ineffective, I learned a lot about machine learning and the model-building process in R that will help me as I begin to apply machine learning in public policy. First, I learned the importance of preprocessing to create a data set that is properly centered, scaled, and cleaned. Such a data set will be ready for robust analysis and will help make the rest of the process down the line easier. Second, I learned the importance of trying to incorporate pre-existing relationships into your analysis. By understanding and applying the correlation of different voxels, for example, I could have both saved myself some time and created an even more data-driven model that made full use of all the information we have to better predict voxel reaction. Third, I learned the importance of taking your time in building models. With a process as complex as machine learning, and a package as large and useful but as unwieldy and opaque as caret, problems can arise at each step. It's important to set yourself up for success by getting started earlier, so that you ultimately have the time to address the problems you encounter.