STAT 5443 Final Project (#3): Exploring High-dimensional Classification Using Uninformed Feature Selection and Traditional Dimension Reduction

April Walker
adw027@uark.edu

Abstract

Often times, data scientists and statisticians must tackle classification projects by utilizing high-dimensional data with little no information regarding specific variables. In these circumstances, the individual must resort to strictly using computational algorithms rather than intuition to choose the most appropriate variables to include in your final model. For this project I chose to classify occurrences of Parkinson's Disease (PD) with dataset including features extracted from voice signals taken from 188 PD patients and 64 healthy patients as the control. In order to explore a variety of algorithms, I chose to use three methods of feature selection and dimension reduction methods: LASSO Regression, Principal Component Analysis (PCA), and a Gradient Boosted Machine (GBM). After reducing my data utilizing these 3 methods, I chose to contrasted my results using a common algorithm - a Random Forest (RF). The variables chosen LASSO slightly outperformed those by the GBM in the RF, while the PCA components had a severely low mean per-class error rate, rendering it unviable. Overall, I felt the LASSO algorithm was the best suited feature selection for this dataset, not only due to it's performance but because it was both the quickest to train and the easiest to tune.

Importance of Feature Selection

Once preprocessing has been finished, generally the most important task for a high-dimensional classification project is feature selection. On a technical level, feature selection is necessary to producing a model which accurately and efficiently develops its predictions. More generally, a model will be more interpretable if the number of features are kept low. Lastly, high-dimensional data increases computation time. On top of the the high-number of features your dataset has, in an industry setting you are likely to have a massive number of instances of your data. If you are wanting to train more computationally intensive models like deep learning algorithms or generalized additive models, you'll be waiting all day (literally) to get your results.

Generally, it's a good idea to get acquainted with your data before pursuing feature selection. However, if your data is extremely-high-dimensional or there's little information regarding your variables, you may have to do your feature selection "uninformed". In this paper, I will discuss some popular methods used for feature selection and dimension reduction.

The Data

For this particular project, I will be using a dataset developed to predict Parkinson's Disease (PD). speech signals were gathered using a tunable Q-factor wavelet transform and feature extraction methods were used to develop the 753 variables I will be reducing. The study included 188 PD patients (81 women and 107 men) and 87 healthy individuals (23 men and 41 women). Due to the imbalanced classification of the data, I chose to analyze a variety of performance metrics to develop my conclusions.

Because this dataset included 756 instances from 252 patients, rather than randomly sample instances I randomly sampled patients to include in the training and testing sets respectively. If I were to randomly sample from the entire dataset, I may unintentionally fit to the training set, giving my models deceptively high performance metrics. The train/test split chosen was 80%.

Technology and Machine Learning Model Overview

For this project all of my models were developed using H2o.ai. For those with a basic grasp of R just dipping their toes into machine learning, I strongly recommend this library. Additionally anyone with a background in math will likely appreciate that their documentation dives into the equations behind their learners. On the other hand, to someone with a background in software development, they might find their documentation clunky yet uninformative. To aid with this somewhat, I recommend the book *Practical Machine Learning with H2O* by Darren Cook.

For this project, I chose to compare the feature selection and dimension-reduction capabilities of LASSO Regression, Gradient Boosted Machines (GBM), and Principal Component Analysis (PCA).

My rational behind LASSO was rather straight forward - unlike Ridge Regression which may shrink many coefficients towards zero, LASSO regularization will likely reduce many variables to actually have zero effect in the model. Because of this, it is frequently used as a feature selection precursor to more complex machine learning models. Specific to H2o.ai, the generalized linear model function has a `lambda_search` boolean variable which allows for for the algorithm to automatically search for the optimal value of lambda, negating the importance of using a grid search to determine the optimal λ value.

The Gradient Boosted Machine was a unique choice based on my personal experience in machine learning. A more through discussion of this algorithm can be found around Chapter 10 within *The Elements of Statistical Learning* by Hastie, Tibshirani, and Friedman. The goal of boosting in general is to strengthen stereotypically weak classifiers (in this case a decision tree). The hyperparameters therefore highly mirror a Random Forest, however include a learning rate. New trees are applied to incrementally changed data in order to produce a ensembled model which much stronger prediction capabilities. While we didn't really cover GBM's explicitly in this course, I felt they were a logical next step in computational statistics. Furthermore, they're insanely popular in the industry. While significantly harder to fine tune than a Random Forest due to the learning rate, the parallels between the two algorithm's hyperparameters make them rather easy to implement. Lastly, their performance is quite strong.

Using a GBM for feature selection is perhaps a bit odd. It is much more computationally intensive than alternative methods, and the tuning aspect can drastically alter your results. Also, from previous experience I have found that LASSO more effectively reduced the number of features in my model at a similar performance level. My prediction was that a GBM would be overkill. However, I have seen GBM's be a popular choice for feature selection amongst data scientists so I was curious how they would perform.

Lastly, I wanted to incorporate a more classic dimension reduction method. PCA allows one to highly reduce the number of dimensions in a dataset by choosing only orthogonal dimensions most responsible for the variance in the data. This method is unsupervised, and these dimensions may not actually be the ones most responsible for the classification of an instance. I chose to only

keep the 29 components responsible for 70% of the variance in the data. While this seems rather low, higher percentages didn't seem to impact performance.

Finally, all models were run through a Random Forest (RF) and metrics were compared to determine overall performance. While the RF isn't the strongest learner, it was chosen simply to provide a speedy comparison. In a real world application, the variables chosen through feature selection may be run through a better performing but more computationally intensive model than an RF.

Methods and Results

For the GBM model, I decided to run a grid search letting the `learning_rate` vary as 0.2, 0.4, and 0.5. The `max_depth` was varied as 2, 3, 5, 7, and 10. Lastly, the `min_rows` was tested as either 1 or 2. I chose to use my test set for validation rather than use k-fold validation. Lastly, I built 100 trees for this machine.

Because I used `lambda_search`, a grid search was not necessary for LASSO. The only other variable of interest is the `alpha = 1`, which indicates we are running a LASSO regression in H2o.ai. Now, I will discuss the results of the GBM and LASSO Regression on their own. Metrics from the testing dataset and variable reduction for these models can be found on Table 1.

The predictive performance of the GBM and LASSO regression are deceptively similar based on the accuracy. LASSO Regression almost flawlessly identified cases of PD in the dataset, incorrectly classifying 1/111 as healthy. On the other-hand, the model severely suffered from false positives, incorrectly predicting 18/42 of the healthy individuals had PD. Meanwhile, the GBM model had a 5/111 false negatives and 10/42 false positives.

At this point in the project, I think it was hard to determine a true winner in terms of feature selection. While the LASSO Regression model was clearly under-performing as a model, I was uncertain whether it's pitfalls would show up in later algorithms. Additionally, LASSO reduced the variables about twice as much as the GBM.

Table 1: Feature Selection Results for LASSO and GBM

	Variables	**Mean per Class Error**	**Log Loss**	**RMSE**	**Accuracy**
GBM	149	0.14157	0.32272	0.31130	90.196%
LASSO	77	0.21879	0.34633	0.32745	87.858%

From here, the reduced datasets with only the variables given by the LASSO and GBM models and the components given by PCA were included. Lastly, as a baseline I ran an RF including all 753 variables given by the original dataset. The results for this are given by Table 2.

Since RF's are very speedy, I ran a rather large grid search with `max_depth` varying from 2, 3, 4, ..., 15 and `min_rows` varying between 1, 2, and 3. My forest had 200 trees.

Table 2: Feature Selection Results for LASSO and GBM

	Variables	**Mean per Class Error**	**Log Loss**	**RMSE**	**Accuracy**
Baseline	753	0.29021	0.41213	0.36234	83.660%
GBM	149	0.20849	0.39465	0.35037	86.928%
LASSO	77	0.17278	0.39275	0.34686	88.889%
PCA	29	0.5	0.64088	0.47352	72.549%

At this point, I would declare LASSO the clear winning. Not only did the 77 variables chosen by LASSO outperform GBM's 149 in terms of accuracy, both LASSO and GBM had the same number of false negatives (4/111) but the GBM had 16 false positives contrasting with LASSO's 13. Both methods provided considerable lift when compared to metrics from the baseline. The baseline model

with all 753 variables severely suffered from false positives, classifying 24/42 healthy individuals with PD.

PCA was an utter failure. The results from the training model were concerning, but not altogether horrible. While models had a tendency to guess positive, the mean per class error was still around 0.3. I had considered leaving the PCA data out, but felt it was at least a learning experience.

Conclusions

For a high dimensional classification project, I discovered that the GBM is an incredibly powerful tool for making predictions. Without any prior feature selection and an $n \approx p$ it was still able to produce an accuracy of 90.196%. However, as a tool for feature selection I felt that LASSO regression was the clear winner. It reduced the number variables twice as much as the GBM and these variables outperformed the model in my RF. Lastly, it was incredibly speedy. While my GBM grid took about 5.5 minutes to train, LASSO took 2.5 seconds.

One reflection is that the performance in the RF is rather arbitrary. I don't actually have statistical confidence that my LASSO variables are better than my GBM variables. It's entirely possibly if I chose an alternative model, the results may favor GBM instead.

Another major area to improve on is my models tendency to misclassification healthy individuals. My choice in stopping metric, log loss, did not concern me. More than likely, I need to take a step back and work on more adequately including under-represented class. For PCA, in particular, I may have benefited from using weights in my dataset such that the ratio is closer to 50/50.

While choosing so many models reduced my ability to completely refine any one method, I believe this project allowed me to take-away a lot more for projects down the line.

Final Comments

I'm sorry I didn't include more pretty graphs. This paper's a bit dull looking, isn't it? I hope the massive amount of text didn't make this too much of a bore!