State Farm Classification Exercise1

Model 1: Logistic Regression

Logistic Regression is a classical modeling method that has many useful characteristics. It is very easy to implement and is not very difficult to train with very few parameters that require tuning. It is very useful to use a comparison to other models. I typically create a logit for all classification problems to serve as a comparison for more complex modeling. Depending on the project requirements, if a logit will satisfy, then there would be no need to develop more time consuming models.

However, what makes it good can also be a disadvantage, in that logistic regressions are linear models, and fitting nonlinear problems to linear models will often sacrifice accuracy. This was evident in my AUC of 0.78

Model 2: Random Forest Classifier

Random forest is an ensemble method that executes many iterations of a more simple technique called a decision tree. Each iteration takes a sample of the data and returns a result, which is then compiled into an overall result. A single decision tree tends to be biased and often over fit to the data. The random forest alleviates this problem but using many trees (typically hundreds) using subsets of the overall data. The resampling aspect of random forests is especially useful with class imbalance. This exercise did not have an extreme class imbalance but the training set only had ~20% positive classes.

Random Forests are computationally expensive and can take much longer to tune the parameters. I used a small instance of RandomSearchCV, which executed 60 versions of the model using a random parameter value from lists I provided. In all it took nearly 4 hours just to execute this portion. Model maintenance would therefore be very time consuming, having to execute this portion again and again. The additional time does grant an added bonus of increased predictive ability, in this case leading to 100% accuracy.

Model selection would depend heavily on the project for which it is being created. Embedding a model in a mobile application would need to be fast and responsive in order to satisfy an end user (which is typically not willing to wait more than 5 seconds for a device to respond). A model result that is more static and used in routine business intelligence would allow for a longer compute time. For example, I use a time series model that runs over the weekends and stores the results in a database which are retrieved at the beginning of each week and incorporated into standard reporting.