State Farm Classification Exercise Executive Summary

Kevin O'Connor

2022-06-03

Abstract

For this project, I was tasked with performing some fundamental model building on a set of nondescript and messy data consisting of 100 unlabeled variables and a binary target variable each with 40,000 observations. After pre-processing and cleaning the data, I was instructed to make two models, and generate predictions on a validation data-set of the same variables over 10,000 additional observations.

The models utilized are simple logit and gradient boosting machine models. The logit model is computationally inexpensive, flexible, and consistent. The gradient boosting model is a tree-based ensemble technique that typically will provide greater predictive power that a linear model, at a cost of computation time and the time spent tuning hyper-parameters. I will make the case that the GBM model is superior.

Data Cleaning and Feature Engineering (1)

The data-set has quite a few "NA" values across columns. Most columns with missing data appear to have missing values at random (MCAR), with a few exceptions. Typical approaches are to replace "NA" values with the mean of the feature in question, or with the most common value. I do not recommend the latter without intuition to back up the imputation, as this can easily introduce bias. The former is a simple, safe, and computationally inexpensive way to impute data, but we can do better.

	colSums(is.na(base_df))
y	0
x1	0
x2	0
x3	0
x4	0
x5	2428

For the majority of the missing values, we will impute utilizing the "mice" package. Multiple imputation using fully conditional specification implemented by the MICE algorithm as described in Van Buuren and Groothuis-Oudshoorn (2011).The mice package allows us to perform sophisticated imputation methods with very simple code. Initially, I tried to run the models by imputing the mean for numeric variables, and the most common for categorical variables, with AUC maxing out ~.77. The mice package will run regressions on each variable, and impute data based on what it predicts the value to be.

First, we address some low hanging fruit. Some variables are so sparse that they likely won't add higher fidelity to our models. In particular "x30", "x44", and "x57" have majority missing values. "x39" has only one level, I.E, no variation. We will exclude these variables entirely. There are some inconsistencies with "x3", where days are sometimes abbreviated. We need to make these day labels consistent. Also, there are two variables that have white space or symbols preceding the value. So, we substitute out the non-numeric characters.

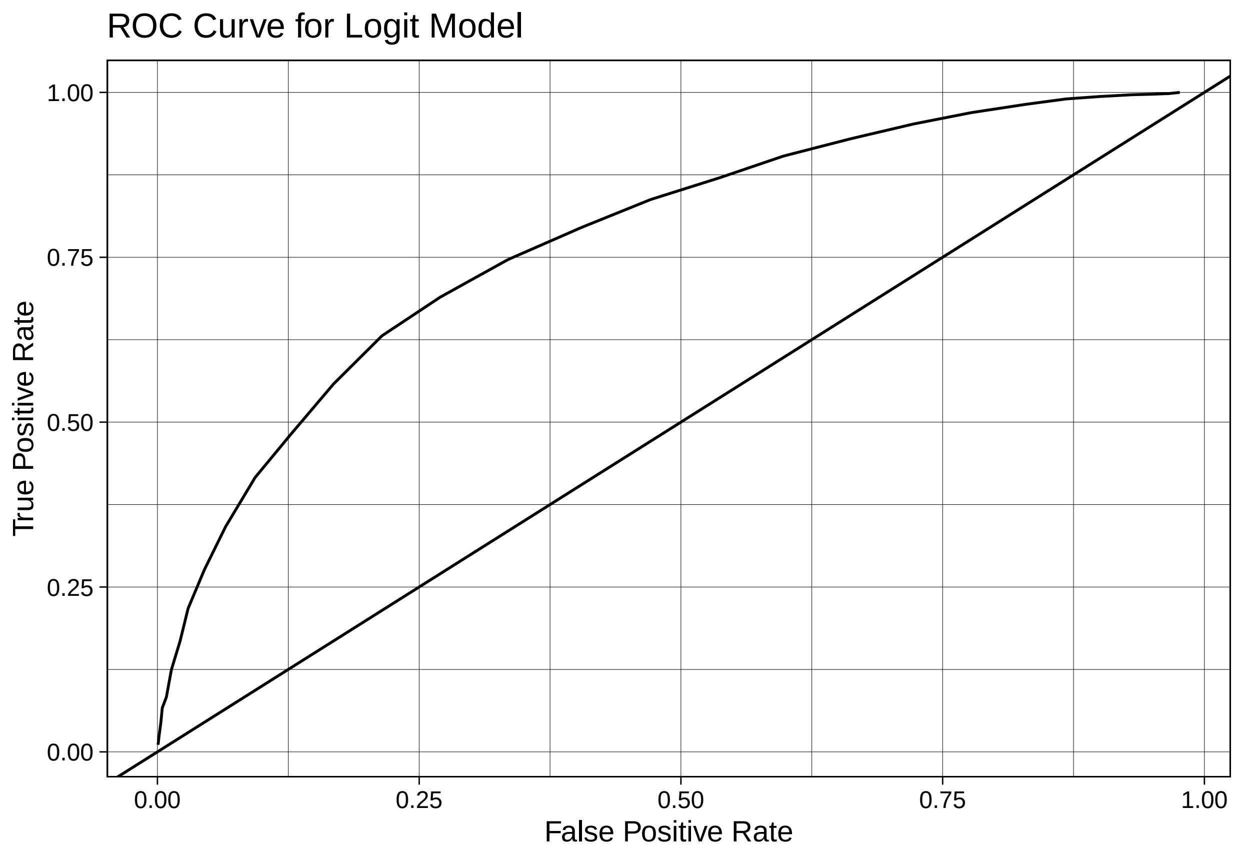
```
##
##  iter imp variable
##   1   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   2   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   3   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   4   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   5   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6

##
##  iter imp variable
##   1   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   2   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   3   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   4   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
##   5   1  x5  x11  x14  x16  x22  x26  x38  x41  x42  x45  x49  x52  x54  x55  x61  x63  x64  x67  x6
```

Next, are categorical variables, such as gender, auto-maker, and state. We will code these "as.factor", assigning numeric levels to the categories. This is quite a bit cleaner than one-hot encoding. Lastly, there is a great deal of variation between variables in each column. So, I z-score the numeric variables such that the models don't incorrectly weigh the value of one variable over another. The last step before we begin modeling, is to split the testing data into model training and model testing sets.

Logistic Regression Model (2)

Model Specification

For the logit model, I choose the most simple implementation. The target binary indicator is modeled against every covariate we selected in the cleaning stage.



Gradient Boosting Machine Model (3)

Tuning

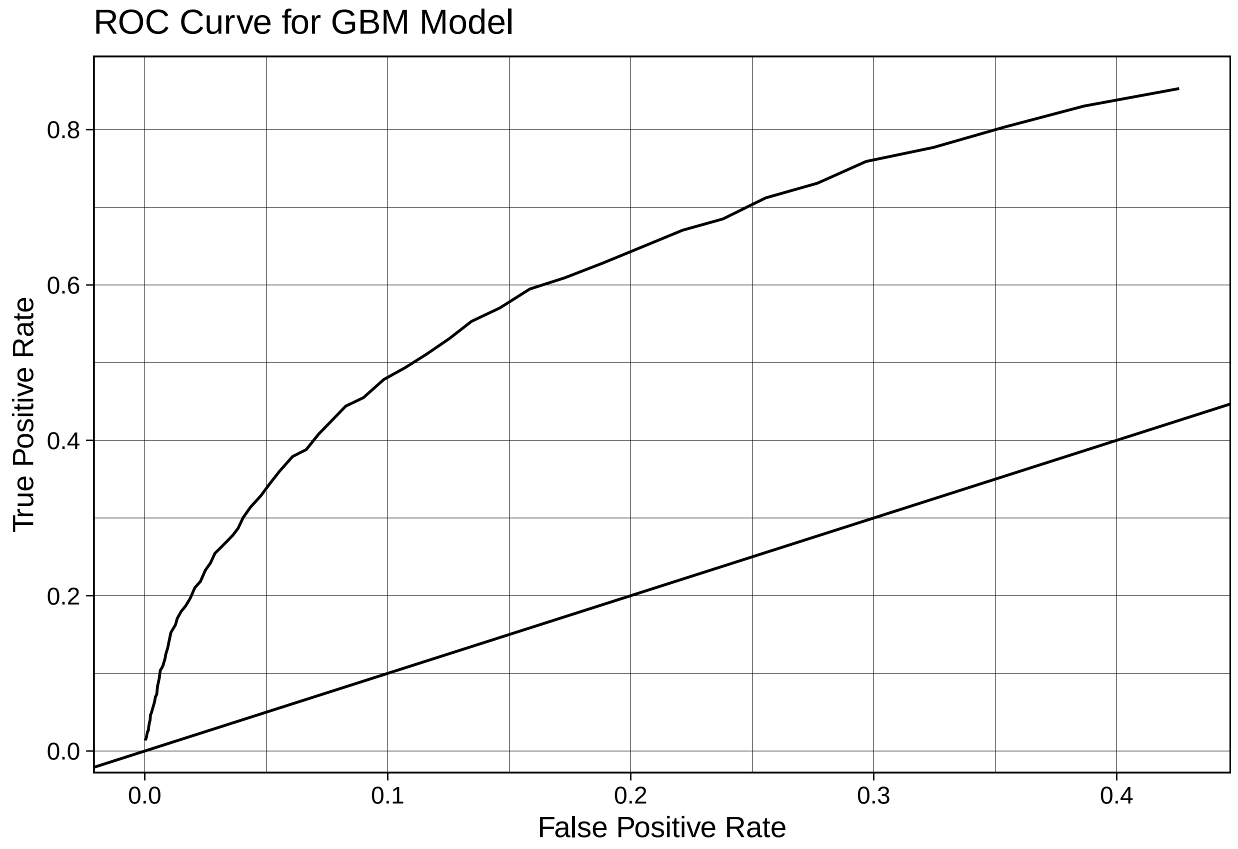
To get the most out of the GBM model, I need to tune hyper-parameters “shrinkage factor”, “interaction depth”, and “number of trees”. This is implemented by looping different levels of the parameters through a model performance function, which generates a table of AUCs across different tuning levels. From these tables, I select the levels that generate the highest AUC.

Model Specification

From the tuning steps, I implement a GBM model with the number of trees set to 1250, the shrinkage rate of .035, the interaction depth of 2, with 3 cross validation folds. The binary target is modeled against every covariate we selected from the cleaning stage.

Table 1: Model Performance by AUC

Model	AUC
Logit	.7701
GBM	.8005



Comparing Model Performance (4)

With an AUC of .7701, the baseline logit model performs fairly on the testing data we withheld. Logit models are very useful, as they are easy to train, prove to be strong for classification problems out of the box, and robust in feature spaces where the distributions of variables are unclear. However, logit models are sensitive to multicollinearity. As the feature space grows, the likelihood of present multicollinearity increases, as such, they are best applied to smaller sets of variables.

With an AUC of .8005, the GBM model outperforms the logit model by 3.9%.This is typical of modern ensemble techniques. The logit model is a single classification model, whereas, the GBM uses 1250 weak classification trees and iteratively combines results to create a strong classifier. The main costs of implementing GBM models is the greater use of compute and time modeling, in addition to the time and effort it takes to tune the model.

Recommendation and Summary (5)

If I was to make a recommendation on which model to use, it would generally depend on the intended use. However, logistic regressions have been used for binary classification for over 50 years. Ensemble methods are becoming more accessible, fast, and powerful over time. GBM will almost always outpeform single classifiers, and without even looking at specific scores, I would advocate for using more modern data science methods.