Predicting Fraud within the Premium Tax Credit Filing Population

Travis Williams

Introduction

*Problem*

The Premium Tax Credit (PTC) is a tax credit that is associated with the Affordable Care Act legislation. Qualifying taxpayers who sign-up for insurance with the Federal Marketplace can have their healthcare premium partially subsidized (paid for) or choose to have the subsidy paid back to them in the form of a credit when they file their tax return. In either case, taxpayers are required to fill-out tax Form 8962 and attach it to their return to ‘reconcile’ this process. As with other recently created tax credits, a large number of tax returns are filed that seek to fraudulently claim the PTC credit. Hence, this project will sought to use IRS data to predict which tax returns claiming the credit are likely to be fraudulent.

*Customer*

IRS operational areas are tasked with the prevention of fraud for specific tax credits. Fraud prevention generally involves a collaborative effort between several IRS research divisions and these operational areas. Though the IRS risk assessment offices currently have fraud filters in place, filters need to be adjusted as new tax credits are offered to the taxpaying population. Working as a research consultant, I will provide an algorithm that can be used by the IRS risk assessment office to augment their current fraud filters to include fraud detection for tax returns that claim the PTC. Improving the robustness of the filters will allow IRS operational areas to do a more effective job at selecting PTC cases to audit, meaning audit cases selected will be more likely to be fraudulent cases. Preventing fraudulent tax credits from being processed is especially important because there is very slim chance money issued for fraudulent credits can be recovered.

Data

*Description and Wrangling*

As IRS tax data is not available to the public, I used two internal IRS sources to gather data for the project: Business Objects reports and the IRS data warehouse known as the Compliance Data Warehouse (CDW). I used Business Objects to query a report of tax returns that were filed claiming the PTC credit for Tax Year (TY) 2014. I exported this report into a series of 5 separate CSV files which I then imported into Jupyter Notebook for further wrangling. In addition, I used SAS to query data on audited tax returns from the CDW. I then exported this data from SAS as a CSV file and imported it into Jupyter Notebook. Once all datasets were in Jupyter, I stacked the 5 CSV files from Business Objects into one single data frame and merged that data frame with the CDW audit dataset to create one data frame with PTC and audit data.

In the existing dataset, several binary indicator variables contained missing values, which I replaced with 0’s. I chose to do so in this case based on past experience with missing IRS data fields, but had I not had such existing knowledge, I would have consulted with someone more familiar with the data to determine if missing values were truly indicative of a ‘no’ for the indicator variable.

*Important Fields and Limitations*

Using my existing knowledge of IRS program operations, I limited the dataset to only variables I felt may be relevant to fraud within PTC. This lowered the dataset from a 77 original features to 57 features. Had I not had such pre-existing knowledge, my approach would have been to consult with subject matter experts to determine fields to keep and eliminate. The resulting dataset contained a multitude of factors that could potentially impact the likelihood of a taxpayer filing a fraudulent PTC return, which I grouped into three main areas:

1. Behavioral factors- data that relates to specific actions by the taxpayer, such as including specific tax forms, calculation errors, and timeliness of return filing.

2. Monetary Factors- information such as taxpayer Adjusted Gross Income (AGI), amount of PTC claims, etc.

3. Demographic Factors- factors such as taxpayer age, location, etc.

In addition, the dataset contains a variable to show the result of the audit cases, whether the return was found to be a ‘good’ return (non-fraudulent) or a ‘bad’ return (fraudulent return). Certain values indicate cases were found to be fraudulent while others indicate the case was found to be non-fraudulent. I used this variable to create a binary indicator of fraud for all the cases in the dataset.

The dataset has three major limitations. First, the dataset is based on data from TY 2014, which are several tax seasons behind the most current IRS tax year data. TY 2014 was chosen because though PTC tax data was available was subsequent tax years, complete audit data was not. Had full TY 2017 audit data been in place, the ideal method would have been to use TY 2016 tax data to run exploratory analysis and train a model, then test the model on TY 2017 tax data. In this case, as no other data option exists to show fraudulent returns past TY 2015, I will test the model I create on TY 2015 tax data. As a consulting analyst, this may be a good opportunity for me to suggest a pilot study to have IRS operations collect more current information for a sample of PTC fraud cases for TY 2016 and TY 2017, then use that information to create a better model. As fraud schemes tend to change annually, modeling from the most current data should in theory provide the most value for operations.

The second limitation involves the mindset behind PTC tax fraud. Quantitative data will show us the factors that may make it likely for a return to be fraudulent. What is will not provide is in depth information on the mindset behind fraud schemes as well as the systemic gaps in the process that allow fraud to perpetuate. As a consulting analyst, I’d suggest qualitative research and process auditing/mapping to augment the results of the quantitative analysis.

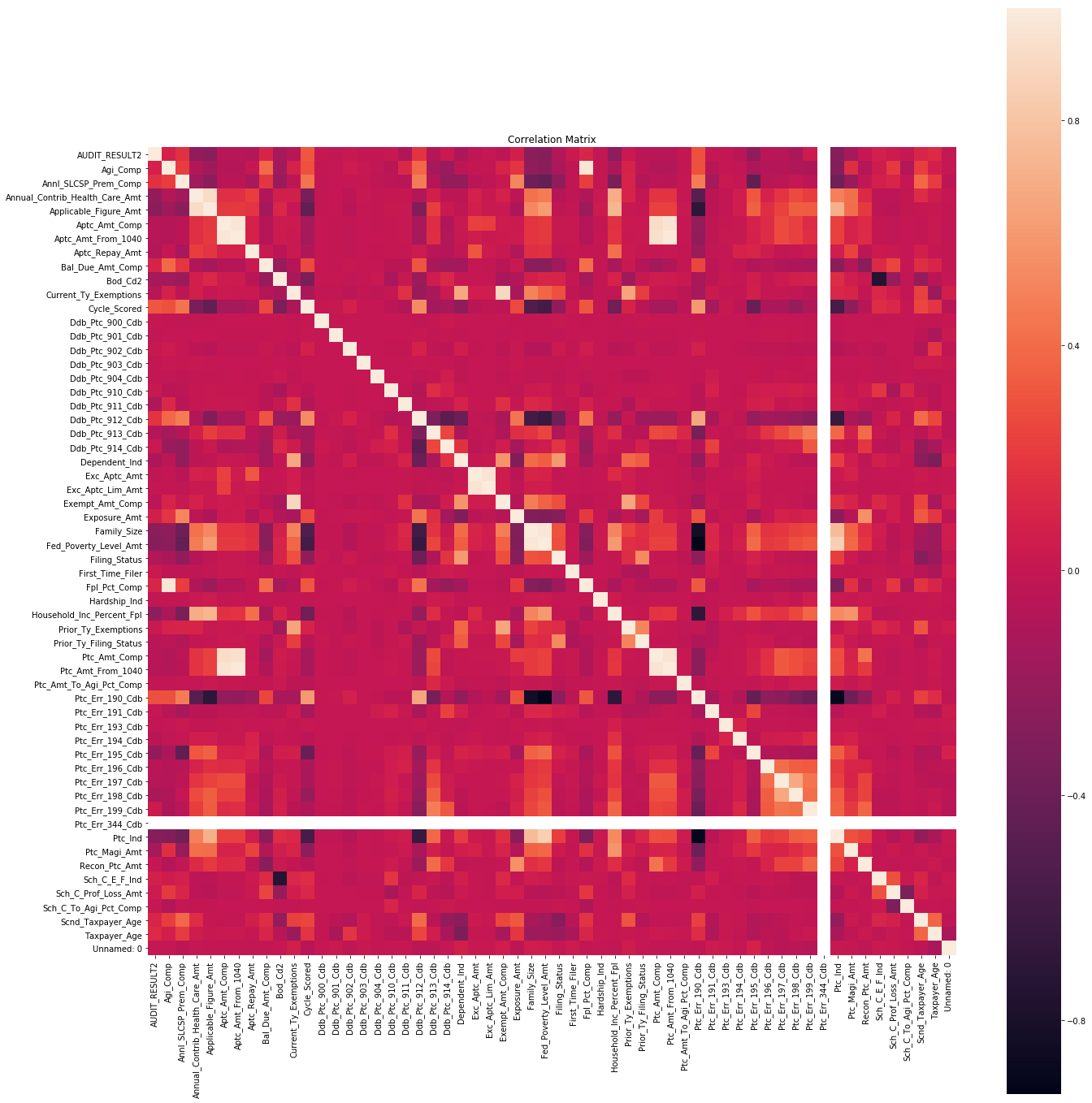
Third, the dataset contains only returns that were audited and such, is a biased dataset. For instance, most cases selected for audit already undergo filtering, which means they are most likely not a representative sample of all PTC cases. We will miss characteristics of PTC cases that were audited and were not fraudulent, which can both skew our model results and make it much harder to improve the performance of the model.

Method

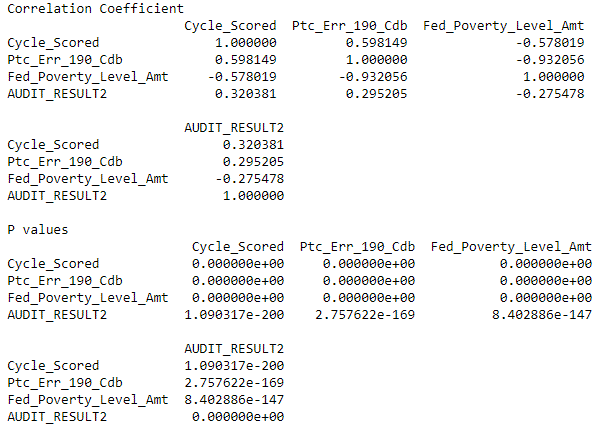
*Exploratory Analysis*

Before conducting modeling, I wanted to get a sense of the variables that would likely have a large impact on whether a return was fraudulent. To accomplish this, I initiated a Pearson correlation and printed a correlation matrix (see Graph 1). In addition, I printed a table of the variables with the highest correlation coefficients as well as their corresponding p-values (to test for significance) (see Table 1).

**Graph 1: PTC Fraud Correlation Matrix**

**

**Table 1: PTC Fraud, Pearson Correlation with P Values**

****

According to the Pearson correlation, the input variables with the largest correlation with filing a fraudulent PTC return are the date the tax return was processed by the IRS (Cycle\_Scored), whether or not a taxpayer failed to attach a Form 8962 to their tax return (Ptc\_Err\_190\_Cdb), and the calculated amount for the taxpayer’s Federal Poverty Level (FPL) (Fed\_Poverty\_Level\_Amt). The FPL amount is calculated based on several factors like the taxpayer’s family size and influences the amount of PTC they qualify for. Both processing date and failing to attach a Form 8962 have positive correlations while the FPL amount is a negative correlation. All of these correlations are significant.

In addition to examining input variable correlation with PTC fraud, I examined the breakout of the target variable, AUDIT\_RESULT2 (see Table2). As you can see, 71% of the tax returns in the audit dataset were determined to be fraudulent.

**Table 2: PTC Fraudulent Return Breakout**



*Machine Learning*

As my PTC audit dataset contained input variables on specific actions by the taxpayer related to their return and other demographic factors about the taxpayer and because have it had a target variable that showed me which cases were fraudulent, I applied supervised learning algorithms to the dataset. I chose to apply two algorithms commonly applied to case classification problems such as this: logistic regression and random forest. Both of these techniques work well when the dataset has a large amount of cases compared to input variables. In this case, I had 8,438 cases and 57 original input variables, so it was an appropriate choice.

Because these were classification models, I used accuracy to assess performance. Accuracy is a measure of the percent of cases correctly classified by the model. In this case, the baseline model for comparison is one in which all returns are predicted to be fraudulent. The model would be 71% accurate if predicting every return was fraudulent.

The data is labeled, so though I know the outcome for all cases, I wanted to train a model and then assess its accuracy on predicting case outcomes. To do this, I divided the data into training data (to build the model based on the relationship between the input variables and the target variable) and test data (to examine the percent of cases the model could correctly classify without knowing the case outcome ahead of time). This process is called cross-validation. However, simple cross-validation can impact performance of the model depending on which section of the dataset is used as the training and which is used as the test dataset. Since it is very difficult to know which cases should be withheld, I used a commonly employed process known as K-Fold cross validation. In this case, I divided the data into five equal parts (known as K folds). I run five separate cross-validations, which each of the five K folds acting as the test set during one run and the other sets acting as the training set.

To improve model performance, I employed several methods. First, to reduce dimensionality of the dataset, I initiated a procedure known as Regularization, which chooses only the most important features (input variables) to use to train the model with. I used L1 regularization penalty to select features for my Logistic Regression model. Feature selection is automatically applied during a Random Forest, so I did not need to initiate that step before running that model. In addition, I used another Sci-Kit learn tool GridSearchCV to determine values of the input parameters that resulted in the best model accuracy and then used these values to improve my models (known as parameter tuning).

For both models, I assessed performance through the following steps:

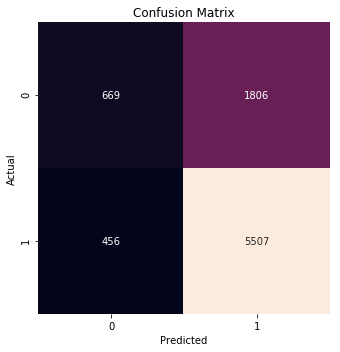
* printed a confusion matrix (true positives, false positives, true negatives, false negatives)
* plotted the true positives rate vs the false positives rate, aka receiver operating characteristic (ROC) curve
* plotted the precision-recall (PR) curve; precision is the true positives over all predicted positives, and recall is the true positives over all real positives

Results

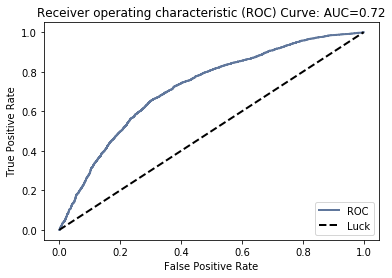
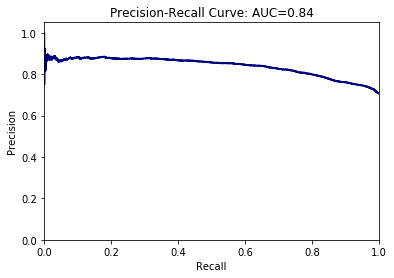
*Logistic Regression*

In a logistic regression model, outputs are categorical. In this case, I had two values; a 0 for a non-fraud case and a 1 for a fraud case. Since it is a regression model it a linear model. After applying L1 regularization to reduce dimensionality, parameter tuning (using GridSearch), and K-fold cross-validation, the accuracy of my model was assessed at 73%. The majority of actual positives were predicted to be positive; hence the true positive rate (recall) was an exceptional 92%. Three fourths of the cases predicted to be positive were actually positive, hence the precision is 75% (100% is ideal) (see Graph 2 below). The Receiver Operating Characteristic (ROC) curve shows the model is better than random at predicting fraud, with the AUC at .72 (1.0 is ideal). The Precision-Recall Curve (PRC) reflects the exceptional recall and good precision of the model, with an AUC of .84 (1.0 is ideal) (see Graph 3 below).

**Graph 2: Logistic Regression Confusion Matrix**



**Graph 3: Logistic Regression ROC & PRC Curves**

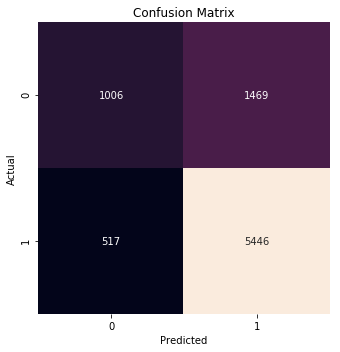


*Random Forest*

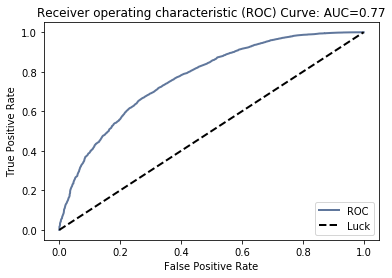
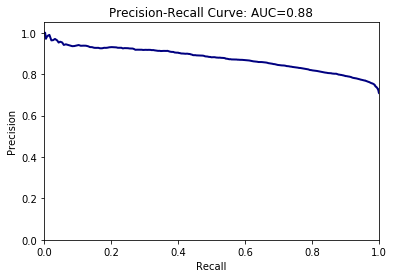
Decision trees are great for nonlinear and conditional relationships. They run fast, but are prone to overfitting. To combat the lack of generalization, best practices include the use of a random forest instead of a single decision tree. Random forest runs a number of decision trees on a subset of the data, and averages to improve prediction and counteract overfitting. In addition, RF automatically does feature selection.

After applying parameter tuning (using Grid Search), and K-fold cross-validation, the accuracy of my model was assessed at 76%. The majority of actual positives were predicted to be positive; hence the true positive rate (recall) was an exceptional 91%. Nearly four out of five cases predicted to be positive were actually positive, hence the precision is 79% (100% is ideal) (see Graph 5 below). The Receiver Operating Characteristic (ROC) curve shows the model is better than random at predicting fraud, with the AUC at .77 (1.0 is ideal). The Precision-Recall Curve (PRC) reflects the exceptional recall and good precision of the model, with an AUC of .88 (1.0 is ideal) (see Graph 6 below).

**Graph 5: Random Forest Confusion Matrix**



**Graph 6: Random Forest ROC & PRC Curves**



Discussion

Random Forest was the higher performing model on my data, with 76% accuracy on the test data. The Precision-Recall Curve (PRC) reflected the exceptional recall and good precision of the model, with an AUC of .88. This outperformed the Logistic Regression, which had an accuracy of .73 and an AUC of .84 on the PRC (see Table 3 below).

**Table 3: Model Performance Metrics**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Accuracy Score** | **AUC-ROC** | **AUC-PR** |
| Baseline | 0.71 | N/A | N/A |
| Logistic Regression (linear) | 0.73 | 0.72 | 0.84 |
| Random Forest | 0.76 | 0.77 | 0.88 |

Recommendations

I have the following recommendations based on current research and data limitations:

1. Encourage the IRS to collect audit pilot data on current PTC cases and make case determinations of fraud vs. non-fraud on these cases. This will allow researchers to determine if current cases match past fraud trends (since data used in these models was from several tax years ago due to limitations).
2. As one of the largest indicators of fraud was whether or not a taxpayer attached a Form 8962 to their tax return, find a more systemic method for verifying if the taxpayer actually received Marketplace insurance throughout the year.
3. As audit cases used to build the models were filtered from the original population of all returns claiming the PTC, conduct a study that takes a sample from the entire PTC return population and examine fraud trends. It is possible certain important features were minimized due to a non-random selection of cases from the original population.
4. Since the cycle the return was processed in had a significant impact on whether a PTC return was fraudulent or not, conduct additional research to determine the cycles with the highest fraud rates and prioritize audit caseload to conduct more audits during cycles with higher fraud rates and fewer audits during cycles with lower fraud rates.
5. Incorporate real-time predictive analytics into audit case selection during the filing season. This will allow the IRS to determine fraud schemes on the fly and adjust audit business rules to effectively stop more fraudulent returns.