Springboard Data Science - Career Track
Capstone Three - Final Report

Automating Deferral Discrepancy Report

Problem Statement

Can true discrepancies in participant deferrals be accurately identified using a supervised learning classification model?

Background

I am currently a Business Analyst at Company B. Company B is a record keeper and third party administrator for retirement savings plans. Our job is to manage the retirement savings plans for the employees of other companies. An employee who participates in their company's retirement plan is known as a "participant" of that plan. We receive payroll files from our clients that tell us how much money we should buy into a participant's retirement account (known as the participant's "deferrals"). The money that we buy was withheld from that participant's paycheck by the employer. The participant makes an election for the amount (or percentage) of deferrals they want withheld from each paycheck on our website, which we then feed to the client.

It is important that the employer is withholding the correct amount of deferrals for each participant. My job is to generate, scrub and analyze deferral discrepancy reports. I scrub thousands of records each week to see if the discrepancies that show up on the report need to be sent out to the client (aka true discrepancies), or if they are due to other factors such as timing issues, internal errors in payroll processing, rounding differences between computer programs, etc. My goal with this project was to take the first step in fully automating the deferral discrepancy reports by focusing on the scrubbing step, as that is the step that is the most time consuming.

Approach

I decided to use a tree-based classification model to predict whether or not a record should be sent out to the client (Send: Yes=1, No=0). My dataset included scrubbed Deferral Discrepancy reports for the first 6 months of 2021, totalling approximately 30,000 records. The following features were included in the final dataframe used for modeling (see Capstone3_DV_ntb1_DataWrangling_EDA for feature descriptions):

Features		
send	pr_def_pct	rate_issue_pr%_low
plan_cd	internal_pct_calc	rate_issue_pr%_high
part_cd	diff_pr_internal_pct	autoenroll_autoescalate_missed
pr_comp	rate_req_if_pct	no_salary_reported
pr_def_amt	rate_req_if_amt	comp_issue_pr%_low
ytd_def_amt	annual_irs_limit	comp_issue_pr%_high
	rate_type_pretax	over_402g

Data Wrangling and Analysis

Missing Data

In the raw dataset there were 4 columns with missing data: rate_eff_date, rate_req_date, rate_req_origin_cd, and plan_change_schedule_cd. For the 3 "rate_" columns, the missing values make sense. They are all associated with a discrepancy type labeled "No internal request record". I decided to delete these records from the dataset, because they are always true discrepancies that need to be sent out to the clients (2507 records total). At that point, I was left with 33 missing values in the plan_change_schedule_cd column. Based on past experience, I know that these are due to a historical issue that has since been corrected. As there is no good way to fill the missing values, I deleted these records as well.

Outliers

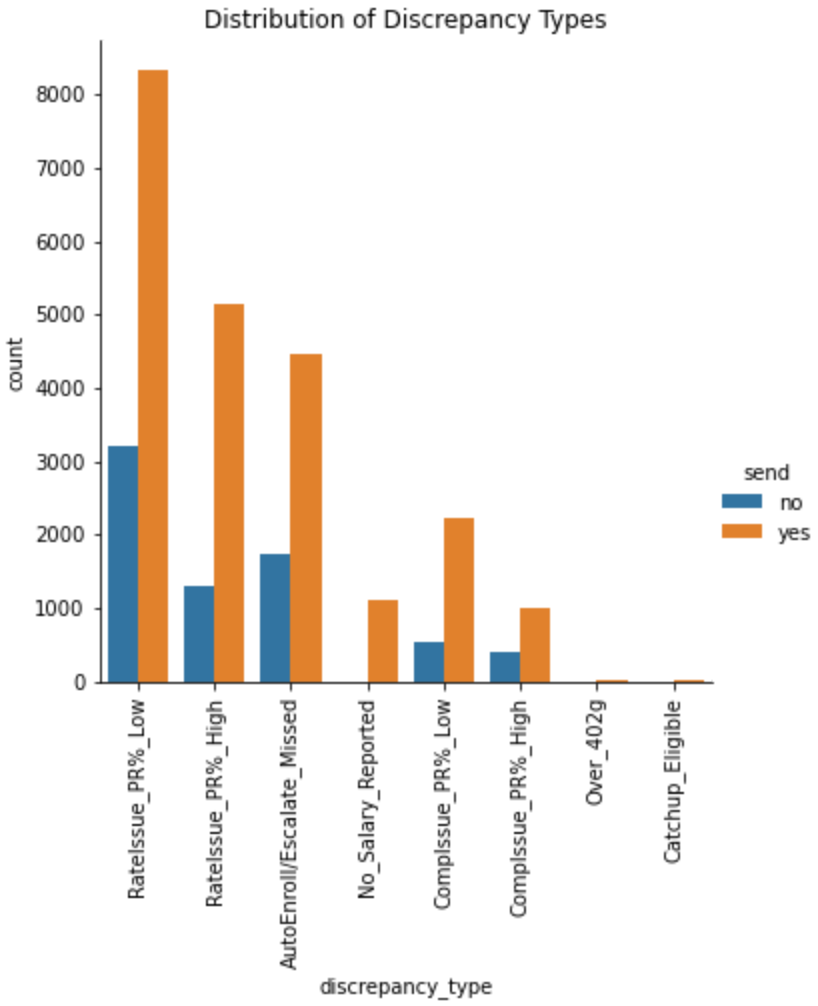
During exploratory data analysis it was evident that there were outliers in the "pr_def_pct", "rate_req_if_pct", and "rate_req_if_amt" columns. I made the following decisions:

- pr_def_pct:
    - Deleted all records with payroll rates over 100%, as these should have been identified and addressed prior to scrubbing the report.
- rate_req_if_pct:
    - Deleted all records with contribution rate percentage requests over 100%, as these are internal input errors that need to be corrected in our system.

- rate_req_if_amt:
    - Deleted all records with contribution rate dollar amount requests greater than or equal to $19500 (irs limit for 2021), as these are internal input errors that need to be corrected in our system.

Exploratory Data Analysis

The most interesting insight that I gained from exploring the data was the distribution of discrepancy types that we send to clients. An in depth analysis of this information has never been conducted before. It's nice to see that the majority of real issues are rate problems, because those are the easiest discrepancy types to fix. Correcting compensation issues are much more challenging because they are usually very difficult to explain to the client.



One-Hot Encoding

I decided to One-Hot Encode two of the columns. The first column was "discrepancy_type" which resulted in the addition of 7 columns to my dataframe identifying each record as having one of 8 discrepancy types:

- RateIssue_PR%_Low
- RateIssue_PR%_High
- CompIssue_PR%_Low
- CompIssue_PR%_High
- AutoEnroll/AutoEscalate_Missed
- No_Salary_Reported
- Over_402g
- Catchup_Eligible

Then, I encoded the "pretax_roth_cd" column which resulted in a single column labelled "rate_type_pretax" with a "1" value denoting a pretax deferral type and a "0" denoting a roth deferral type.

Modeling

Model Testing and Selection

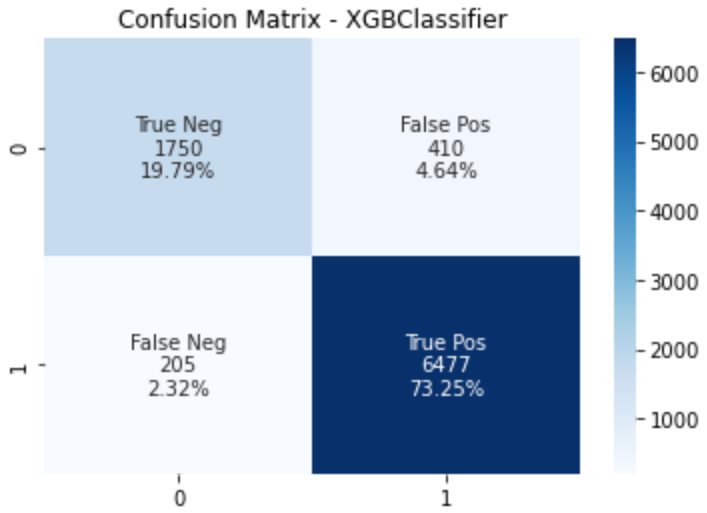
I split the data into training and test sets, and then tested five separate tree-based classification models. As you can see in the table below, the model with the best overall performance was the XGBClassifer. Based on these results, I proceeded with tuning the hyperparameters for the XGBClassifier model.

Models (Default - No Tuning)	ROC AUC Score	Accuracy	False Pos Rate (%)	False Neg Rate (%)
DecisionTree	0.840	0.888	6.19	4.98
RandomForest	0.832	0.897	7.20	3.10
AdaBoost	0.690	0.825	14.04	3.42
GradientBoost	0.753	0.864	11.35	2.21
XGBoost	**0.884**	**0.927**	**4.91**	**2.40**

Hyperparameter Tuning

The results after hyperparameter tuning of the XGBClassifier model are below.

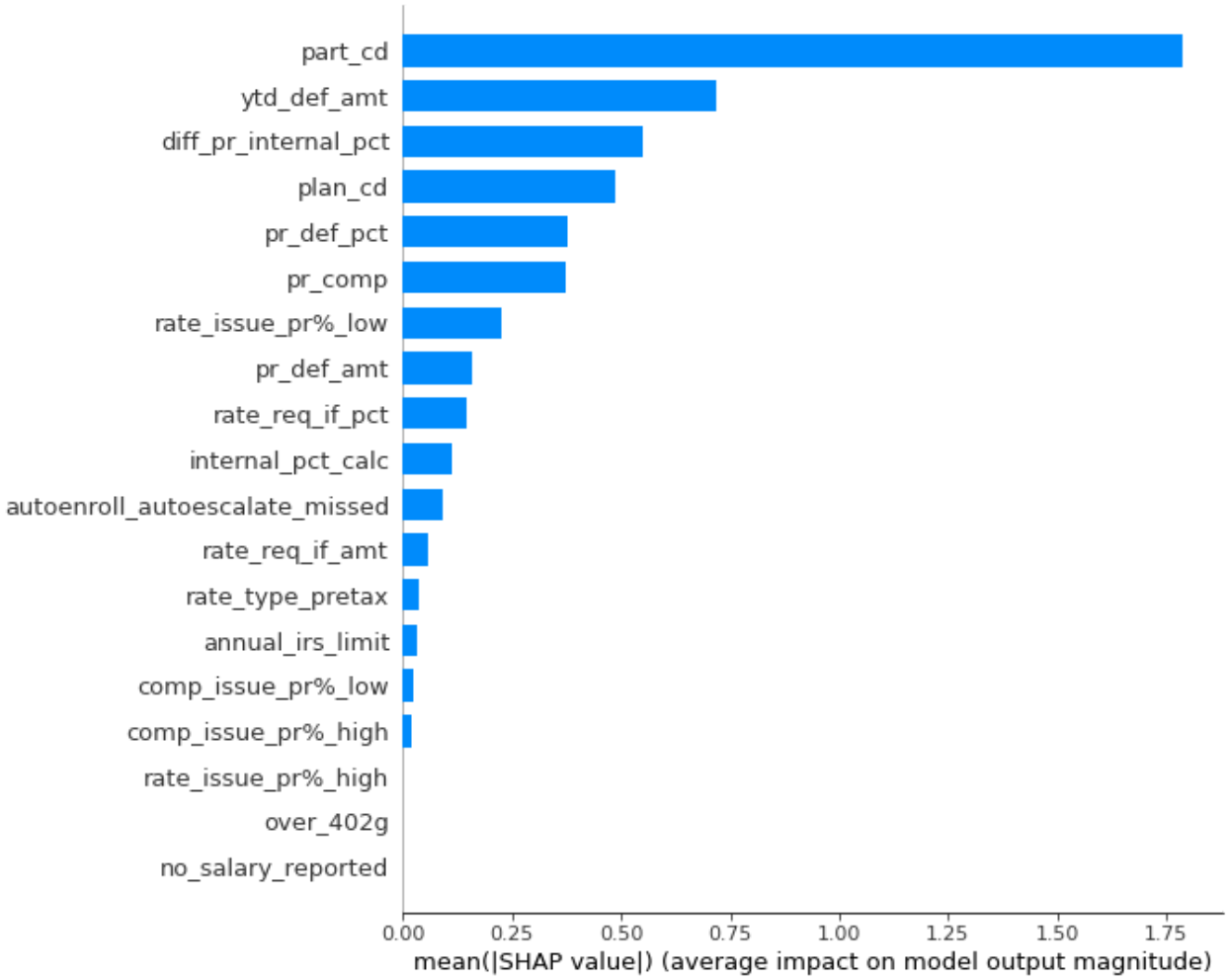
Model (after tuning)	Best Parameters	ROC AUC Score
XGBClassifier	eta: 0.15 max_depth: 13	0.89



Based on my past experience, a client is more likely to get upset when they receive a false positive than if they discover that a discrepancy was missed (false negative). Even though the false positive rate is almost double the false negative rate, I am still happy with these results. My model has selected the correct class approximately 90% of the time, which is very good. While I still feel that there are improvements to be made, this model is a great start.

Feature Importance

As you can see in the figure below, the participant identifier, year-to-date deferral amount, and the difference between the payroll and internal records rate were the top 3 features in terms of impact on my model's classifications. To be completely honest, I'm having trouble figuring out why these are the top 3 features and I think this requires further analysis. The "diff_pr_internal_pct" makes a little bit of sense to me, because the top three discrepancy types that we generally send out to clients ("true discrepancies") are all rate issues, meaning that there is a substantial difference between the rate being submitted through payroll and the requested rate in our system.



Future Recommendations

I started this project knowing that it was only the first step in a long process. My end goal is to fully automate Company B's deferral discrepancy reporting process and I have many ideas of next steps to take. I would first like to try to expand my approach to a multi-class classification problem. The issue that I have with the binary classification is that it is difficult to tell why a decision was made to send or not send a discrepancy and I think that it is important when dealing with clients to have a clear reason. Second, I would like to go further back with my data collection. Due to time constraints I was only able to acquire data for the first 6 months of 2021, but I do have access to data for all of 2020. Having a more robust dataset can only help me to make a better model. Finally, I would like to combine my revamped model with the report generation procedure to create a fully automated process from start to finish. The time that this would save for me or future employees would be invaluable.