**PDAX – Predicting Customer Churn**

By: Li Sun

**Abstract**

The goal of this project is to use machine learning classification algorithms to generate predictions on PDAX’s customer churn. Working with PDAX’s actual data, I created three classification models: logistic regression, random forest, and XGboost. All three models performed well given the data.

**Design**

PDAX is a Cryptocurrency Trading Exchange in the Philippines. They’ve been successful and have grown to 500k users in the country. Despite PDAX’s growth, they are not aware of the amount of customer churn in their customer base. Every month a significant amount of their customer who traded on PDAX’s platform last quarter will churn and not trade at all. Thus, the problem we are trying to solve is: How can we predict which of PDAX’s customers will churn next month?

**Data**

Using SQL, I pulled 10 features of data, broken down into two sections – static data and dynamic data. At a per customer level (each row is a customer), Static Data includes features such as age, gender, address, phone number, etc. Dynamic data consists of a ratio: # of trades in the last two weeks / # of trades in the 8 weeks (before last two weeks). That way I use 10 weeks of trailing data to predict churn. Additionally, I included another dynamic data which is the monthly net deposit / per customer. Dataset is small with ~26k rows.

**Algorithms / Solution Paths**

Data Cleaning & Feature Engineering

* + Cleaning the data – Data is cleaned. All users who hasn’t traded in the last 10 weeks was deleted. This narrowed the total rows from 150k+ to 26k.
  + Feature Engineering – I was able to engineer the two dynamic features, which is taking a ratio of trades vs. no trade + netting last month’s of deposits and withdrawals.
  + Class imbalance – to correct the data imbalance, I tried both oversampling and using class weights, which either adversely affected or did not affect performance.

Machine Learning Algorithms:

* **Logistic Regression** – Running Logistic Regression as my initial model, I was able to get a good performance of 73% accuracy, 74% Precision, and 98% Recall with overall F1 at 84%
* **Random Forrest** – Running the Random Forest model, I was initially able to get the model at higher precision and recall than Logistic Regression. However, the model was overfit. I then tuned the RF model, limiting the tree depth to only 3 maximum. This got rid of the overfit.
* **XGBoost** – Using AUC as the evaluation metric, this model ran very well. Final precision is at 82% and recall at 91%.

**Tools**

Sklearn, Pandas, Numpy, Matplot, SQL

**Communication**

Slides and visuals are in PPT, submitted via PDF.