Machine Learning Engineer Nanodegree
Capstone Proposal

Syed Faiz Raza

April 9th, 2020

I. Domain Background

Predicting campaign response accurately will help companies target the most potential customers while saving marketing expense. To put it in numerical terms, if your overall response rate is 5% but you were able to predict the 10% most potential customers with a response rate of 80%, your return on investment would increase by 8 times compared to targeting everyone. Furthermore, targeting the wrong customer might backfire, making the product less appealing.

Targeting the right customers can be done with a supervised learning model, learning from the past responses/non-responses. This project aims to do that for a marketing campaign from a client of Arvato Financial Services. This type of modeling often face a big challenge of imbalanced class problem as the response rate is very low. Hence, the models will bias towards predicting non-response since a naive guess of everything being the majority class would be a pretty good guess. Tackling imbalanced class problem could be used using a combination of sampling methods and boosting models.

II. Problem Statement

We will predict who will respond to a mailout campaign. Our model will be evaluated using area under the receiver operating characteristics curve (AUC) as this is the metrics for the Kaggle competition. We will also discuss the model's performance with respect to other metrics such as sensitivity, recall, and area under precision-recall curve so that one can evaluate the model with different objectives.

This is a binary classification problem with highly imbalanced class, with only about 1.23% of our customers in the data responds positively to the campaign.

III. Datasets and Inputs

There are four datasets, all of which have identical demographics features

• Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns)

• Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns)

• Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).

• Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).

In addition to the above data, there are two additional meta-data:

• DIAS Information Levels — Attributes 2017.xlsx: a top-level list of attributes and descriptions, organized by informational category.

• DIAS Attributes — Values 2017.xlsx: a detailed mapping of data values for each feature in alphabetical order.

IV. Solution Statement

The goal is to optimize AUC, which measures the ability of the model to rank a random true positive higher than a random true negative.

Different sampling techniques and models will be employed.

V. Benchmark Model

The benchmark model is a Gradient Boosting machine with class weight balancing. I chose this algorithm because it has shown effectiveness in many problems and it doesn't require missing value handling. Since missing values might have different context and therefore different meaning, I want to see how later I could beat the benchmark with no missing value handling. The 10-time-repeated 5-fold validation AUC is 76.4%

VI. Evaluation Metrics

According to kaggle, the evaluation metric for this competition is AUC for the ROC curve, relative to the detection of customers from the mail campaign. A ROC, or receiver operating characteristic, is a graphic used to plot the true positive rate (TPR, proportion of actual customers that are labeled as so) against the false positive rate (FPR, proportion of non-customers labeled as customers).

VII. Project Design

The project will involve the following steps:

- *Run a baseline model with XGBoost with no tuning parameters and dataset as-is.* XGBoosts has proved itself in many problems so I want to see how much my extra effort in the following step will pay off compared to a default option
- *Handle missing data*
  Many algorithms requires non-missing data so this needs to be handle.
- *Feature engineering*
  The training dataset only includes nearly 43000 data points with over 300 features. Using all features would likely result in overfitting. In addition, some features needs preprocessing and new features could be created to increase predictive power for the model. We also run a sanity check on the dataset here

- *Model tuning*

Define validation procedure to evaluate each modeling experiment. Each experiment concerns with algorithm family, hyperparameter-tuning, and model ensembling. - *Compete on Kaggle*