Customer Churn Prediction

Udacity ML Nanodegree Project Proposal

Jin Kwon (jinkwon0425@gmail.com)

Jan. 3rd, 2020

1. Domain Background

Maintaining strong customer base is crucial for a company to thrive. Due to its importance, companies invest a lot in conducting customer analytics, such as accurately calculating customer lifetime value and identifying customers' most common needs.

One important topic under the same umbrella is understanding and predicting customer churn. For example, a high customer churn rate can undermine a company's profit and return on investment[1]. Many CRM firms such as Salesforce[2] and other SMB are investing in applying AI and machine learning to tackle customer churn. This signifies how lots of industries are interested in analyzing and predicting customer churn.

Machine learning model that can predict and bring insights into contributing factors to customer churn can benefit a company in two ways: First, the model can function as an early warning signal for a company to make timely intervention at an individual customer level. An intervention can include a one-on-one session to understand a customer's pain points and offering a discount. Second, if the machine learning algorithm can pinpoint common causes of customer churn, the company can optimize its resources to maintain a healthy customer base.

Additional benefit from a robust out of this project would be that it can benefit in other similar domains such as predicting and understanding employee attrition from a firm.

2. Problem Statement

The problem that will be tackled in this project can be thought of as two parts.

First is predicting customer churn. This will involve building a machine learning model that, given a set of features of a customer, predicts (classifies) whether the customer did or did not churn at the given timepoint. The result can either be a binary output (churned or did not churn) or a probability (90% certain that the analyst churned).

Second is understanding contributing factors to customer churn. This can be done by analyzing feature importance from machine learning models. If multiple different models with high accuracies point to similar factors as major contributors to customer churn, then these businesses can utilize the insight to take actions to mediate customer churn.

3. Datasets and Inputs

Telco customer churn data[3] provided by the sponsor shared through Kaggle will be used. This data contains static features (non-time series) including customers' demographic features (gender, whether he/she is a senior citizen, has a partner, etc.) and service-related features (payment method, has an online security, has phone service, etc.). Lastly, and most importantly, the data contains a churn label, which indicates whether the customer left the service within the last one month from the timepoint of data collection.

Frequent problem with data that entails modeling rare event such as customer churn or machine failure is that there are few data points with the positive label. In the Telco customer churn data, we have roughly 2k churned data points and 5k not churned data points. Although this is not an extreme case of class imbalance, machine learning models that were reported to work well in such scenario will be explored (e.g., weighted random forest).

4. Solution Statement

Solution to this problem will be two-fold. First, a machine learning model that predicts customer churn with acceptable accuracy will answer predicting customer churn. Second, features that are significantly predictive of the customer churn from the accurate set of models will assist businesses with making optimal decisions on customer support.

5. Benchmark Model

Luckily with this particular data and project, we have initial and mature machine learning models shared by Kaggle users. The main benchmark model will be the simplest regression-like model. Given classifying churn is a binary classification task, my benchmark model will use logistic regression with ROC AUC as a metric. Following is a benchmark logistic regression model that I quickly mocked up using the reference noted in the notebook: https://www.kaggle.com/jinkwon0425/logistic-regression-benchmark-model. With the logistic regression model, we are achieving ROC AUC of 0.857.

6. Evaluation Metric

As this project involves imbalanced dataset, with data points labeled as churned half the size data points labelled as not-churned, evaluation metrics that take this into consideration will be adopted. A few examples are AUC-ROC, False Positive, and False Negative.

7. Project Design

Project will follow the following steps:

a. Data exploration and manipulation:
   - **i.** Explore feature correlation → features that are highly correlated will be removed or reduced using dimensionality reduction methods (e.g., PCA)
   - **ii.** Identify N/As → If there are negligibly few rows with N/As, then these rows will be removed. If there is a column with significantly large number of N/As (e.g., more than 40%), then this column will be dropped assuming it does not have predictive power. Otherwise, missing NAs will be imputed using one or more methods (https://www.theanalysisfactor.com/seven-ways-to-make-up-data-common-methods-to-imputing-missing-data/)
   - **iii.** Identify outliers using simple statistics (boxplot, histogram, etc.) or statistical methods (Cook's Distance) → outliers will either be removed or be treated like an NA where imputation method from ii will be used to replace with values that make the most sense.
   - **iv.** Data augmentation for class imbalance (optional). Up-sampling minority using SMOTE will be explored to check if this process increases model performance. In this step, we will make sure to apply SMOTE only to the training data, since we want to avoid testing the model performance against artificially generated data points.

b. Model-specific data manipulation:
   - i. When using models that are impacted by the scales of feature values, data will be scaled accordingly to accommodate the model's need.
   - ii. Encoding categorical features for models that require numerical features.

c. Train/test data split. Here, we will be wary of the label class imbalance and use stratification by the `Churn` label. I may need to do use train/test/validation split to check accuracy of stacked models (step e).
d. Modeling:
    i. Algorithms for binary classification will be considered including logistic regression, xgboost, random forest, etc. Given the class imbalance, models that were proven to work well such as weighted random forest will also be considered.
    ii. Model tuning: The last ML modeling I only did manual hyperparameter tuning. However, during this project I am excited to explore sklearn's gridsearch and other automatic hyperparameter tuning methods[5].
e. More Modeling using Stacking: Often times in the industry we want to push the model performance as much as possible. Also personally a new topic that I was eager to explore, stacking of multiple models will be conducted[6]. Hyperparameters (weights of each model) will also be tuned using methods similar to the ones discussed in d.ii.

Reference:

[1] Customer Churn and Revenue: https://onix-systems.com/blog/customer-churn-rate-and-its-impact-on-business-performance
[2] Salesforce "Predicting Churn and Keeping Customers: AI and Customer Success": https://www.salesforce.com/video/3620925/
[3] Telco Customer Churn data on Kaggle: https://www.kaggle.com/blastchar/telco-customer-churn
[4] Accuracy metrics for minority class: https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/:
[5] https://scikit-learn.org/stable/modules/grid_search.html
[6] Model Stacking https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/#:~:text=Stacked%20Generalization%20or%20%E2%80%9CStacking%E2%80%9D%20for,dataset%2C%20like%20bagging%20and%20boosting.