AWS Machine Learning Engineer Nanodegree
Capstone Proposal

Churn classification

Domain Background

Businesses face a significant challenge in dealing with customer churn, which occurs when customers stop using a company's products or services. High rates of customer churn can negatively impact a company's revenue and profitability [1]. To address this issue, businesses can leverage machine learning algorithms to identify the factors that contribute to churn. These algorithms are designed to detect early warning signs and identify customers who are more likely to leave voluntarily [2] and [3]. In this project, I will explore three different algorithms: logistic regression, decision tree, and random forest, to develop an accurate classifier that predicts which customers are likely to churn and which will not. By applying these powerful tools, I aim to create a robust solution that can effectively address the problem of customer churn.

Problem Statement

In today's highly competitive industry, customer churn is a critical challenge that many businesses face. When a customer switches to a rival company that offers a better solution or product, churn occurs. Identifying customers who are likely to churn is essential for any business to implement effective retention strategies for long-term success, as acquiring new customers is more costly than retaining current ones. However, manually identifying such customers or using conventional techniques can be time-consuming and difficult. Therefore, developing machine learning techniques that can accurately identify customers at risk of churning based on their past behavior and interactions can significantly improve the ability to prevent future churn. The goal of this project is to create a model that can accurately predict churn and improve customer retention.

Datasets and Inputs

To conduct this project, I will use the churn dataset available at Kaggle Bank Customer Churn Dataset. This dataset consists of 10,000 observations, 12 features, and is divided into two classes: 0 (not churn) and 1 (churn). The features included in this dataset are described below:

Column	Description
customer_id	**The first feature in the dataset is a unique identifier assigned to each customer. This column is not used as an input for the model and is solely for identification purposes.**
credit_score	**The second feature in the dataset is a numerical value that represents the creditworthiness of a customer. This feature is used as an input to the model to predict churn, where higher scores indicate lower risk and vice versa.**
country	**The third feature in the dataset is the country where the customer resides. This feature is used as an input to the model to predict churn.**
gender	**The gender of the customer. It is used as an input to the model to predict churn.**
age	**the gender of the customer. This feature is utilized as an input to the model to predict churn.**
tenure	**the number of years that the customer has been with the bank. This feature is utilized as an input to the model to predict churn.**
balance	**the amount of money that the customer has in their account as another feature. This feature is utilized as an input to the model to predict churn.**
products_number	**The number of banking products that the customer has with the bank. It is used as an input to the model to predict churn.**
credit_card	**whether the customer has a credit card with the bank or not. This feature is utilized as an input to the model to predict churn.**
active_member	**Whether the customer is an active member of the bank or not. It is used as an input to the model to predict churn.**
estimated_salary	**The estimated salary of the customer. It is used as an input to the model to predict churn.**
churn	**The target variable is a binary feature that indicates whether the customer has left the bank or not during a certain period. This variable takes a value of 1 if the customer has left and 0 if they haven't. The target variable is used to train and evaluate the machine learning model for predicting customer churn**

In this project, the dataset will be divided into three subsets: **Training (70%), Validation (20%),** and **Test (10%).** This division allows us to train the model on the training set, tune its hyperparameters using the validation set, and evaluate its performance on the test set. The use of three separate subsets helps to prevent overfitting and ensures that the model generalizes well to new, unseen data.

Solution Statement

in order to address the challenge of customer churn, I will implement three machine learning models: Logistic Regression, Decision Tree, and Random Forest. These models will be trained using the dataset, and their performance will be evaluated based on various metrics such as accuracy, recall, and precision. The AUC-ROC metric will be utilized to determine the most effective model.

Benchmark Model

To set a benchmark, I will use the mentioned models along with the final model, and compare their performance using the AUC-ROC metric. I will use Scikit-learn library and its features, such as Pipelines for building pipelines, OneHotEncoder for creating dummy variables, RobustScaler for normalizing data, and SimpleImputer for imputing missing data to develop the models. For data manipulation, I will utilize Polars, a DataFrame library that is fully written in Rust and provides an API for Python. To visualize data, I will rely on Matplotlib, Folium, and Seaborn libraries. Additionally, due to the small number of positive examples in the dataset, I will apply Synthetic Minority Oversampling Technique (SMOTE) from the imblearn library to address the issue of imbalance [5], [6] and [7].

Evaluation Metrics

- For evaluation, I will use several metrics such as: accuracy, confusion matrix and AUC-ROC.

Project Design

Data Download

- The dataset will be downloaded using the Kaggle API provided by Kaggle, and then the zip file will be extracted using the bash unzip command.

Data Preprocessing

- I will use Polars to load the entire dataset and exclude unnecessary features such as customer_id. Then, I will compute statistics such as class distribution, mean, median, minimum, and maximum values. Additionally, I will use box plots to identify outliers and calculate correlations between some features

Subset Data Splitting

Then, I will split the selected dataset into 70% training, 20% validation, 10% test datasets using scikit-learn help functions.

Model Training and Evaluation

I will train and evaluate the models using the scikit-learn classes: LogisticRegression, DecisionTree, and RandomForest, with their default hyperparameters, and measure the improvement obtained by applying the SMOTE technique to address the class imbalance issue. Subsequently, I will perform hyperparameter optimization of the models and evaluate them using k-fold cross-validation to measure accuracy [8]. I will also plot the confusion matrix of each model. Finally, I will compare the models using the AUC-ROC metric to select the best one