Customer Churn Prediction: Machine Learning and Neural Network Approaches

Aakash Sharma
aakash.student.phy21@iitbhu.ac.in
Phone: 9004189581

September 6, 2023

1 Introduction

Customer churn, the loss of clients or customers, is a significant concern for businesses in various sectors. Understanding and predicting churn is crucial for long-term business sustainability. This report focuses on building machine learning models to predict customer churn for a hypothetical service provider.

The dataset used for this project contains several features, including:

- Customer Demographics: Age, Gender, Location
- Service Usage: Total Usage in GB, Monthly Bill
- Subscription Details: Subscription Length in Months

Our approach followed a comprehensive machine learning pipeline, encompassing the following steps:

1. **Data Preprocessing:** Cleaning the dataset to handle missing values and outliers.
2. **Feature Engineering:** Creating new features to improve the predictive capability of the models.
3. **Model Training:** Using various machine learning algorithms to train models on the preprocessed dataset.

4. **Hyperparameter Tuning:** Adjusting model parameters for optimal performance.

5. **Model Evaluation:** Assessing the models' predictive accuracy using metrics such as F1 score, precision, and recall.

The project employed both traditional machine learning models, such as Logistic Regression, Random Forest, and ensemble methods like AdaBoost, XGBoost, and LightGBM, as well as a neural network model implemented using PyTorch.

Despite extensive efforts in feature engineering and model optimization, the models' performance remained suboptimal, prompting further investigation and providing opportunities for future work.

2 Data Preprocessing

Data preprocessing is a crucial step in any machine learning pipeline. The quality of the data and the amount of useful information that it contains can significantly affect the ability of the model to learn, hence affecting the model's performance. In this project, various preprocessing steps were applied to the initial dataset to make it suitable for training machine learning models. These steps include:

2.1 Handling Missing Values

Missing values in the dataset were identified and handled appropriately. Depending on the nature of the feature, missing values were either filled with a default value or estimated using statistical methods.

2.2 Encoding Categorical Variables

The dataset contained categorical variables such as Gender and Location, which needed to be converted into a format that could be provided to machine learning algorithms. Label Encoding was employed to convert these categorical variables into numerical form.

2.3 Outlier Treatment

Outliers can significantly impact the performance of machine learning models. A comprehensive analysis was conducted to identify and treat outliers in the dataset.

2.4 Train-Test Split

The dataset was divided into training and testing sets to evaluate the machine learning models' performance. A 70:30 split was used, where 70% of the data was used for training the model, and the remaining 30% was used for testing.

2.5 Feature Scaling

Although feature scaling was initially considered, it was later decided not to scale the features based on the nature of the data and the choice of models. Feature scaling is often not required for tree-based models, and given that a significant portion of our models were tree-based, this step was omitted.

These preprocessing steps enabled us to prepare a clean and informative dataset, ready to be fed into various machine learning algorithms for training.

3 Feature Engineering

Feature engineering is a vital step in the machine learning pipeline, as the right features can make or break a model. In this project, several new features were engineered to capture the intricacies of customer behavior and improve the models' predictive capabilities.

3.1 Polynomial Features

Polynomial features were created for variables like 'Subscription Length' and 'Monthly Bill' to capture their non-linear relationship with the target variable. For example, 'Subscription_Length_Square' and 'Monthly_Bill_Square' were computed.

3.2 Interaction Features

Interaction features were also considered, like 'Usage per Bill,' which was computed as the ratio of 'Total Usage in GB' to 'Monthly Bill.'

3.3 Flag Variables

Flag variables like 'High_Bill_Flag' and 'Low_Usage_Flag' were created to indicate whether a customer has an unusually high bill or low usage, respectively. These flags were set based on specific thresholds.

3.4 Customer Tenure Groups

Customers were segmented into different tenure groups like 'Short-Term' and 'Long-Term' based on their 'Subscription Length.' These groups aimed to capture behavioral patterns that are common to customers who have been subscribed for similar durations.

3.5 Variable Transformation

Various mathematical transformations were applied to some of the features to make them more suitable for certain algorithms or to highlight specific aspects of those features.

Despite these extensive feature engineering efforts, the machine learning models did not show a significant improvement in their performance, which suggests that further research and experimentation are needed in this area.

4 Traditional Machine Learning Models

Various traditional machine learning models were employed in this project to predict customer churn. Each model was chosen for its unique strengths in handling different types of data structures and relationships. The models were trained on the processed and engineered features from the dataset.

4.1 Logistic Regression

Logistic Regression was chosen as a baseline model due to its simplicity and efficiency. It is particularly useful for binary classification problems like

customer churn prediction. However, its linear nature limited its ability to capture more complex relationships in the data.

4.2 Random Forest

Random Forest, an ensemble of decision trees, was employed for its ability to capture non-linear relationships and interactions between features. The model also offers the advantage of feature importance evaluation.

4.3 AdaBoost

The AdaBoost algorithm was used to create a strong classifier based on multiple weak classifiers. It focuses on instances that are difficult to classify, making it a robust option for this problem.

4.4 XGBoost

XGBoost, an optimized gradient boosting algorithm, was also used. It is known for its high performance and speed and is widely used in machine learning competitions.

4.5 LightGBM

LightGBM, another gradient boosting framework, was employed for its efficiency and effectiveness, especially with large datasets. It also handles categorical features natively, which is an added advantage.

4.6 Hyperparameter Tuning

To optimize these models, hyperparameter tuning was conducted using techniques like Grid Search and Randomized Search. This tuning aimed to find the most effective set of parameters that could produce the best results for each model.

4.7 Evaluation Metrics

The models were evaluated based on several metrics, including Accuracy, F1 Score, Precision, and Recall. Despite the extensive feature engineering

and hyperparameter tuning, the traditional machine learning models did not yield significantly improved results.

5 Neural Network Model

To capture potentially complex relationships in the data, a neural network model was implemented using PyTorch. The neural network provides greater flexibility in modeling non-linear relationships compared to traditional machine learning models.

5.1 Architecture

The neural network consisted of multiple fully connected layers with varying numbers of neurons. Dropout layers were also included to prevent overfitting. The model was initialized with random weights and used the ReLU (Rectified Linear Unit) activation function.

5.2 Optimization and Regularization

The Adam optimizer was employed for its efficiency and adaptive learning rate. To address the class imbalance, class weighting was applied during the loss computation. Early stopping and learning rate scheduling were also implemented to improve training efficiency and model generalization.

5.3 Training on GPU

The model was trained on a GPU to speed up the training process, making it feasible to experiment with various architectures and hyperparameters within a reasonable time frame.

5.4 Evaluation Metrics

The model's performance was evaluated using metrics such as Accuracy, F1 Score, Precision, and Recall. The neural network did not show a significant improvement in these metrics, indicating that it struggled to learn the underlying patterns in the data effectively.

5.5 Challenges and Observations

One major challenge was the model's inability to move past a certain accuracy threshold. Despite experimenting with various architectures and regularization techniques, the model's performance did not improve significantly. The high recall and low precision suggest that the model was biased towards predicting a specific class.

5.6 Future Improvements

Given the limitations in the current architecture and performance, future work could include exploring more advanced neural network architectures, such as those employing self-attention mechanisms, and further fine-tuning of hyperparameters.

6 Observations and Results

This section summarizes the key observations and results obtained from both traditional machine learning models and the neural network model.

6.1 Traditional Machine Learning Models

Despite extensive feature engineering and hyperparameter tuning, the performance metrics for traditional machine learning models like Random Forest, AdaBoost, XGBoost, and LightGBM did not show significant improvement. While these models are generally robust for classification tasks, they were unable to capture the nuances of the dataset effectively.

6.2 Neural Network Model

The neural network model also faced similar challenges. Despite experimenting with various architectures, regularization techniques, and hyperparameter settings, the model's performance did not surpass a certain threshold. The model seemed to be biased towards predicting a specific class, as indicated by the high recall but low precision and accuracy.

6.3 Challenges

The primary challenges faced during this project include:

- Class imbalance, making the model biased towards the majority class.
- Insufficiently expressive features, limiting the models' ability to generalize.
- Overfitting, despite regularization and early stopping techniques.

6.4 Key Takeaways

- Advanced feature engineering techniques are needed to capture the complexities of customer churn.
- More sophisticated model architectures may be required for accurate predictions.
- Further research and experimentation are essential for improving model performance.

7 Future Work and Recommendations

The challenges and observations from this project provide several avenues for future work and improvement. Below are some recommendations that could be considered for future iterations of this project.

7.1 Advanced Feature Engineering

Given the limitations of the existing feature set in capturing the complexities of customer behavior, there is a need for more advanced feature engineering techniques. This could include more sophisticated interaction terms, polynomial features, or even domain-specific metrics that could better capture customer churn.

7.2 Alternative Model Architectures

The performance of both traditional machine learning models and the neural network model suggests that alternative, more complex model architectures may be worth exploring. This could include the use of self-attention mechanisms, recurrent neural networks, or even ensemble methods that combine different model types.

7.3 Handling Class Imbalance

Further methods to address class imbalance could be employed, such as more advanced resampling techniques or cost-sensitive learning. This could help the model learn the minority class better and thus improve overall performance.

7.4 Hyperparameter Optimization

More extensive hyperparameter optimization techniques could be applied, potentially using automated methods like Bayesian Optimization, to find the optimal set of parameters for each model type.

7.5 Utilizing Additional Data

If available, incorporating additional data or features could provide the models with more information, improving their predictive power. This could include customer interaction data, more detailed usage statistics, or even external data sources.

7.6 Model Interpretability

Given that the end goal is not just to predict but also to understand customer churn, future work could also focus on model interpretability. Techniques like SHAP (SHapley Additive exPlanations) could be employed to understand the impact of each feature on the model's predictions.

By addressing these areas, there is a strong potential for significant improvements in the model's performance, making it more useful and reliable for predicting customer churn.