Machine Learning to Predict Customer Retention

Jonayet Lavin

March 2024

1 Models and Techniques Overview

Random Forest classifier was tested, chosen for its adeptness at handling categorical data and complex non-linear feature interactions, which are prevalent in the dataset. This model also mitigates overfitting through its ensemble approach. To improve upon this, AdaBoost was utilized, which combines weak learners (decision stumps) for enhanced robustness and reduced hyperparameter tuning, and reduced susceptibility to overfitting by weighting learners based on their accuracy. Additionally, neural networks were explored to compare performance, finding them slightly less effective than Random Forest for the dataset, likely due to its simplicity and size.

2 Approach

2.1 Data exploration, processing and manipulation

For this project, notable data exploration, processing and manipulation was performed. In order to compare different models and avoid encoding and data-filling (for missing values) discrepancies, uniform data processing was performed.

The customerID column was dropped as it was irrelevant to the learning task since it is only an artificial column to enumerate each data point. Missing value imputation was then performed by filling the missing values of the TotalCharges column with the median of the column in order to maintain data integrity of a column that will be a significant factor for this learning task. For categorical encoding, LabelEncoder was used from sklearn.prepossessing to transform categorical features into a numerical format. This was necessary for the Random Forest model which takes numerical input.

Feature scaling was also performed with StandardScaler from sklearn. preprocessing to prevent feature bias where one or more features dominates due to their large scales. This prevented a feature with larger numerical values to dominate the model's decision process.

Whether the dataset classes were balanced was checked, since training a model on an unbalanced dataset can lead to a bias towards the majority class leading to potential misclassification for the minority class. As shown in Figure 1, the dataset is unbalanced with a ratio of classes as 0.357. Balancing the dataset was done with sklearn.utils.resample on the minority class, which was done after train-validation split. Synthetic Minority Over-sampling technique (SMOTE) was also attempted for the AdaBoost Classifier using SMOTE from imblearn.over_sampling. Techniques of balancing the training set proved ineffective.

In terms of train-validation split, an 80-20 training-validation set split was used to evaluate the model's performance on unseen data without risking too much overfitting.

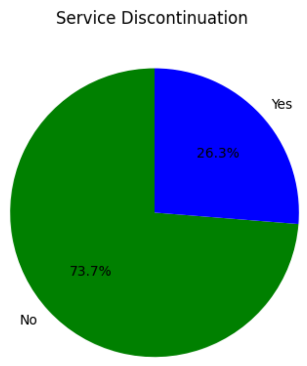

Figure 1: Proportions of service discontinuations vs service continuations.

2.2 Details of models and techniques

For the Random Forest, it was aimed to optimize the model's performance through parameter grid search. This involved defining a comprehensive grid of parameters, which includes the number of trees in the forest, the maximum depth of trees, and the minimum number of samples required to split an internal node. So by searching this grid, the aim was to identify the combination of parameters that maximized the model's accuracy.

AdaBoost was also tried with the entire original feature vector (excluding customerID column). The model was found to reach an ROC AUC score of 0.84 when trained with 80% of the provided training data on the default parameters. To optimize this model, a 2D grid search was performed on the number of estimators and the learning rate. Note that by default, the estimator for Adaboost in sklearn is a Decision Tree Classifier. The result of this optimization are discussed in Section 3.3.

Predictions were also conducted with a variety of neural networks. Networks with fully connected layers was experimented with, of different depths and layer sizes, and also design choices such as having a bottleneck in the network or not were considered. Convolutional layers on the network was tested as well. The experiments were followed with grid search to find the best option. On the optimization side, SGD and Adam optimizers were tested with different learning rates, learning rate schedulers, different choices of momentum and weight decay. "Binary Cross Entropy Loss" was used, which is ideal for classification tasks. Feature engineering was used by determining the most important features with the Random Forest model and training the neural network only with these features, which proved ineffective.

3 Model Selection

3.1 Scoring

For the neural network models, accuracy and ROC-AUC were used as scoring metrics. Initially, accuracy was used as the scoring metric, to get a general idea of how well the models were performing. This provided a good starting point for hyperparameter tuning. To further refine the models and make comparisons between the RandomForest and AdaBoost models, ROC-AUC was used as the scoring metric. This allowed the comparison of the models on a more granular level, and allowed for a more informed decision on which model to use for the final predictions. With the accuracy metric, it was not possible to make a clear decision on which model to use, as the accuracy scores were very similar for all models. However, with the ROC-AUC metric, it was possible to differentiate between the models and make a more informed decision. ROC-AUC is a better metric for this dataset, as it is imbalanced, and ROC-AUC takes into account the true positive rate and the false positive rate, which is important for imbalanced datasets.

For the Random Forest and AdaBoost classifiers, ROC-AUC was used as the scoring metric. ROC-AUC was used for these classifiers because predicting whether a customer will continue to purchase the services or not is a binary classification problem, which ROC-AUC is suited for. ROC-AUC also has low sensitivity to class imbalance compared to metrics like accuracy. Therefore, given that the dataset had a class imbalance, ROC-AUC allowed to have a reliable performance assessment as it measured the model's ability to distinguish between the two classes without being biased by the class distribution. Moreover, Random Forest and AdaBoost are both ensemble methods, and ROC-AUC can effectively capture the performance improvement these methods provide, considering the complex dataset where the relationship between features and the target variable were non-linear.

3.2 Validation and test

The AdaBoost classifier was found to score the best.

3.3 AdaBoost

It was found that using a number of 86 estimators with a learning of 0.4 gives the best ROC AUC score on the validation set. This combination was obtained by performing a grid search. In fact, performing this grid search is crucial for monitoring over-fitting. Figure 2 shows the mean score (as measured by ROC AUC) on the training and testing set as the number of estimators used to fit the model was increased. It can be seen that for each learning, the overfitting (due to a larger variance) becomes noticeable after using large numbers of estimators. To see this more clearly, Figure 3 shows the same plot but for a fixed learning rate (in this case equal to 0.4). In this case it is clear that the distance between the two train and test curves becomes larger after using more than 86 estimators.

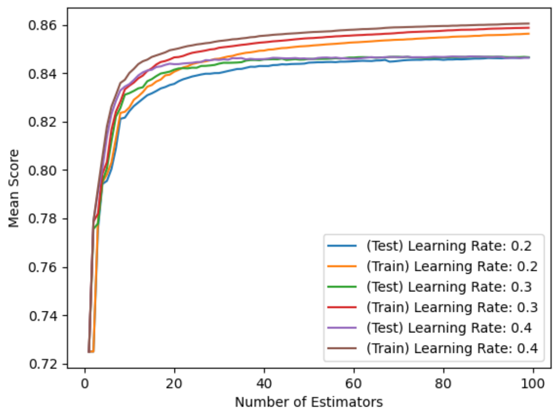

Figure 2: Train vs test mean ROC AUC score for the AdaBoost model

3.4 Neural Networks

Exploratory experiments followed by a grid search (GridSearchCV class from scikit learn and the skorch library was used) showed that a neural network with only fully connected layers and 1 hidden layer, with each layer having 64 nodes performed the best. It was found that using Adam optimizer with 0.001 learning rate, no momentum and weight decay, trained for 10 epochs performed the best. It was observed that learning curves became smoother when a large batch size like 4274 was used (Figure 4) however a batch size of 64 performed similarly. The ROC AUC score for the best neural network was 0.8416 which allowed for abandoning this model to continue to work on other models.

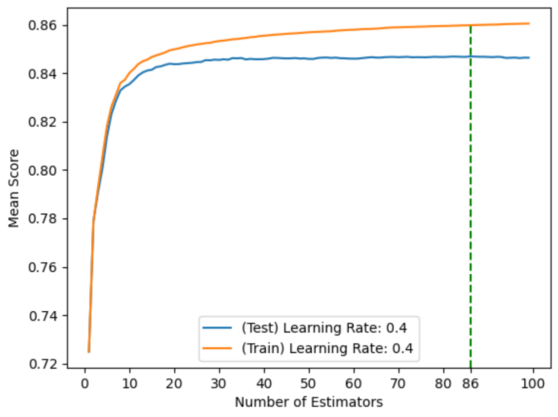

Figure 3: Train vs test mean ROC AUC score for the AdaBoost model and the best learning rate

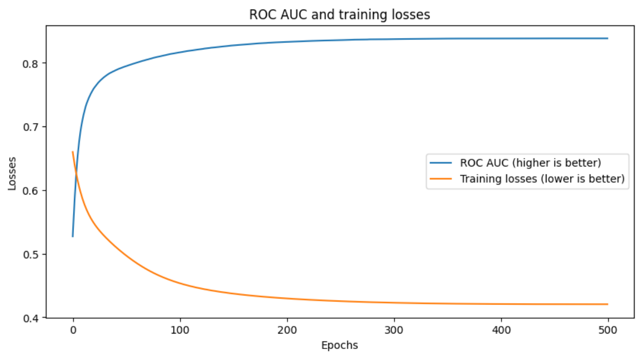

Figure 4: ROC AUC and training losses plotted together for a neural network trained with a batch size of 4274

4 Conclusion

4.1 Insights

Regarding the process of determining which features have the most influence on the prediction target, feature importance within the context of a given algorithm (like Random Forest or AdaBoost) is internally consistent and can be used to interpret the specific model, but comparing importance scores across different algorithms is substandard as they may not align due to different methodologies for calculating importance. Notably, AdaBoost, neural networks, and Random Forest were used, but Random Forest is best for assessing feature importance due to its straightforward and interpretable metric, averaged impurity decrease, across all trees. For instance, the importance of features in AdaBoost can be influenced by the order in which features are used as the model is sequential. A feature used early in the sequence can have a larger impact on improving model performance compared to later features. Therefore, the feature_importances_ attribute of RandomForestClassifier model was used to obtain the relative importance of each feature for the prediction. The top 10 features (with their importance scores) were obtained to be tenure (0.1658), Contract (0.1634), TotalCharges (0.1387), MonthlyCharges (0.1184), OnlineSecurity (0.0867), TechSupport (0.0804), InternetService (0.0500), OnlineBackup (0.0361), PaymentMethod (0.0313), and DeviceProtection (0.0232). (This RandmomForestClassifier received a public score of 0.84567). Overall, the project proved to be a learning opportunity about the significance of data preprocessing and how it can significantly impact the performance of various models. Through experimentation with Random Forests, neural networks, and AdaBoost classifiers, and employing techniques like grid search for hyperparameter tuning, it was possible to learn about the tradeoffs between model complexity and overfitting. The use of ROC-AUC as the scoring metric showed the importance of choosing appropriate metrics, especially in imbalanced datasets.

4.2 Challenges

Performing grid search over 4 hyperparameters, each having 6 possible values for the Random Forest was extremely time consuming and required long waiting hours whenever trying to improve upon the best performing model of the time. This was only a constraint in terms of time, since computations were performed on Google Colab and personal machines were not unavailable during the training time.

Similarly, time was also a challenge for the neural networks. For the neural networks the grid search involved not only switching through hyperparameters but also different model architectures. In the future, it would be better get a baseline for each of the different model architectures and then apply a unique grid search for each of them. This will hopefully result in running more grid searches that each take up less time.

5 Miscellaneous

The AUC is a performance measurement for classification problems at various threshold settings and represents the degree or measure of separability, so how well a model is able to distinguish between classes. AUC was used as for the metric because it provides a balance between sensitivity and specificity. This is because AUC accounts for both true positive rate and false poisitive rate, allowing for a single measure of the overall model performance regardless of the classification threshold. AUC is also applicable to imbalanced datasets as it is not biased towards the majority class. Moreover, AUC measures the quality of the model's prediction regardless of the classification threshold, making it valid for problems where the threshold is not initially known. Since the dataset was relatively imbalanced with more customers not discontinuing the service, it may be better to use Precision-Recall AUC, as precision and recall are directly impacted by the model's ability to correctly identiy the minority class, so this metric would be a more direct assessment of model performance.

There are methods and pipelines that can be parallelized among the machine learning methods that was used. Algorithms used like Random Forest and gradient bossing can be parallelized as they build multiple trees that can be constructed simultaneously. Sci-kit learn has the n_jobs parameter for this parallelization. Hyperparameter tuning methods used like grid search can be parallelized since each parameter combination is independent of others. Also, cross validation can be parallelized since each fold is independent.

Additional data that would be useful to collect are customer feedback, customer interaction data, and service usage patterns. Customer feedback such as surveys can offer direct insights into the customer satisfaction and potential reasons for discontinuing the service. Customer interaction data such as call logs and chat transcripts can provide insight into customer satisfaction as well. Usage patterns such as service usage frequency and peak usage can identify engagement levels, also helping in predicting probability of discontinuing.