rfb-vibe / telecom-customer-churn Public

0 stars 0 forks

Star Unwatch

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

rfb-vibe add presentation pdf ... 2 hours ago 12

View code

Predicting Telecommunications Customer Churn

Authors: Rebecca Frost-Brewer

According to Profitwell, the average churn rate within the telecommunications industry is 22% (churn rate referring to the rate customers close their accounts or end their business relationship).

Business Understanding

The telecommunications company, SyriaTel, is faced with the problem of better predicting when its customers will soon churn. They need a solution that will predict whether a customer will ("soon") stop doing business with SyriaTel. This will be valuable to SyriaTel, so that they may better understand their churn rate and identify areas they may address to improve its churn rate.

Finding predictable patterns using a classification model will benefit SyriaTel's business practices to minimize customer churn.

To determine which classification model best predicts potential customer churn, I will evaluate models' performance using the F1 score. The F1 score is the harmonic average of two other metrics, precision and recall, and is suited well to evaluate imbalanced datasets.

Precision summarizes the fraction of examples assigned the positive class that belong to the positive class whereas the recall summarizes how well the positive class was predicted and is the same calculation as sensitivity. Both precision and recall values fall in the range [0,1], with 0 indicating no precision/recall and 1 perfect precision/recall. These values can be combined into one metric, the F1 score, which is the harmonic average of the precision and recall scores. The F1 score also ranges [0,1].

The closer to 1 the F1 score, the more perfect the model is classifying samples.

Data Understanding

The data source for this project comes from SyriaTel's churn data. This data is suitable for the project because it includes key performance indicators and data points from SyriaTel related to its customers and their accounts as well as whether the customer churned or not.

The data consists of 3,333 observations with 21 features and no missing values.

Explanation of Features

- `state` : the state the user lives in
- `account length` : the number of days the user has this account
- `area code` : the code of the area the user lives in
- `phone number` : the phone number of the user
- `international plan` : true if the user has the international plan, otherwise false
- `voice mail plan` : true if the user has the voice mail plan, otherwise false
- `number vmail messages` : the number of voice mail messages the user has sent
- `total day minutes` : total number of minutes the user has been in calls during the day
- `total day calls` : total number of calls the user has done during the day
- `total day charge` : total amount of money the user was charged by the Telecom company for calls during the day
- `total eve minutes` : total number of minutes the user has been in calls during the evening
- `total eve calls` : total number of calls the user has done during the evening
- `total eve charge` : total amount of money the user was charged by the Telecom company for calls during the evening
- `total night minutes` : total number of minutes the user has been in calls during the night
- `total night calls` : total number of calls the user has done during the night
- `total night charge` : total amount of money the user was charged by the Telecom company for calls during the night
- `total intl minutes` : total number of minutes the user has been in international calls
- `total intl calls` : total number of international calls the user has done
- `total intl charge` : total amount of money the user was charged by the Telecom company for international calls
- `customer service calls` : number of customer service calls the user has done
- `churn` : true if the user terminated the contract, otherwise false

Modeling

Baseline Model

For the baseline model, I use a decision tree stump:

```
baseline = DecisionTreeClassifier(max_depth = 3, class_weight = 'balanced', random_state = SEED)
```

Vanilla Modeling

I create vanilla models using a decision tree classifier, logistic regression, k-Nearest Neighbors classifier, random forest classifier, and eXtreme Gradient Boost (XGBoost) classifier.

Each model has its own advantages and disadvantages, which is why I will include each to best determine the strongest predictive model for the stakeholder.

Decision Tree Classifier

Decision Trees are a non-parametric supervised learning method used for classification and regression with the goal of creating a model that predicts the value of a target variable by learning simple decision rules inferred from the data features; in the case of this project, the decision tree classifier will attempt to predict customer churn.

Logistic Regression

Logistic regression is a linear model that is used for classification and models the probability of one event or class (out of two alternatives) taking place and that the target variable is categorical, e.g., a customer churns (1) or does not churn (0). Thus, logistic regression is an applicable model to our business problem and may be of value.

K-Nearest Neighbors Classifier

The neighbors-based classification is a type of instance-based learning that computes classification from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point. For this modeling, k-Nearest Neighbors implements learning based on the nearest neighbors of each query point, where k is an integer value specified by the user.

Random Forest Classifier

Random Forest Classifiers are a type of ensemble method, which means a diverse set of classifiers is created by introducing randomness in the classifier construction; in this case, he prediction of the ensemble is given as the averaged prediction of the individual decision tree classifiers. In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.

XGBoost

From its documentation, XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework.

Gradient Boosting alogrithms are a more advanced boosting algorithm that makes use of Gradient Descent. It starts with a weak learner that makes predictions on the dataset. The algorithm then checks this learner's performance, identifying examples that it got right and wrong. The model then calculates the Residuals for each data point, to determine how far off the mark each prediction was. The model then combines these residuals with a Loss Function to calculate the overall loss.

XGBoost, or eXtreme Gradient Boosting, provides a parallel tree boosting that solve many data science problems in a fast and accurate way. It is a stand-alone library that implements popular gradient boosting algorithms in the fastest, most performant way possible. In fact, XGBoost provides best-in-class performance compared to other classification algorithms.

Tuned Models

Following the first set of vanilla model scores, there is certainly room for improvement in our predictive modeling. While the vanilla XGBoost model had an F1 score of 0.8 on the testing data, it may be possible with further tuning of all five models that we can improve that F1 score.

To best solve our business problem of predicting customer churn, these iterative models will attempt to improve results.

Feature Importances refer to techniques that calculate a score for all the input features for a given model; the scores simply represent the "importance" of each feature with a higher score indicating that the specific feature will have a larger effect on the model that is being used to predict a certain variable.

The **Classification Report** is a printed report showing the main classification metrics: precision, recall, and the F1 score, along with the sample size (indicated as 'support').

Remember, precision is the ability of the classifier not to label as positive a sample that is negative (what percent of the predictions were correct?), and recall is the ability of the classifier to find all the positive samples (what percent of the positive cases were caught?). The F1 score can be interpreted as a weighted harmonic mean of the precision and recall that reaches its best value at 1 and its worst score at 0.

The report also includes the macro average (averaging the unweighted mean per label) and weighted average (averaging the support-weighted mean per label).

The classification report is meaningful to this business problem because the model eventually chosen to predict customer churn should correctly determine a given class.

The Confusion Matrix is a function evaluates classification accuracy by computing the confusion matrix with each row corresponding to the true class.

The higher the diagonal values of the confusion matrix the better, indicating many correct predictions of True Positives and True Negatives.

For each of the tuned models, I create a parameters grid that is passed into `GridSearchCV` with scoring set as `F1_weighted` due to the imbalanced dataset in order to determine the best performing parameters for each classifier. Here is the final results table of the tuned models:

	accuracy	precision	recall	f1
Logistic Regression	0.863	0.569	0.240	0.337
KNN	0.881	0.739	0.281	0.407
Decision Tree	0.934	0.851	0.661	0.744
Random Forest	0.941	0.909	0.661	0.766
XGBoost	0.946	0.888	0.719	0.795

The final model selected to address the business problem of predicting customer churn is the tuned XGBoost classifier. Remember, the XGBoost classifier provides parallel tree boosting that can solve many data science problems in a fast and accurate way.

Final Model Metrics

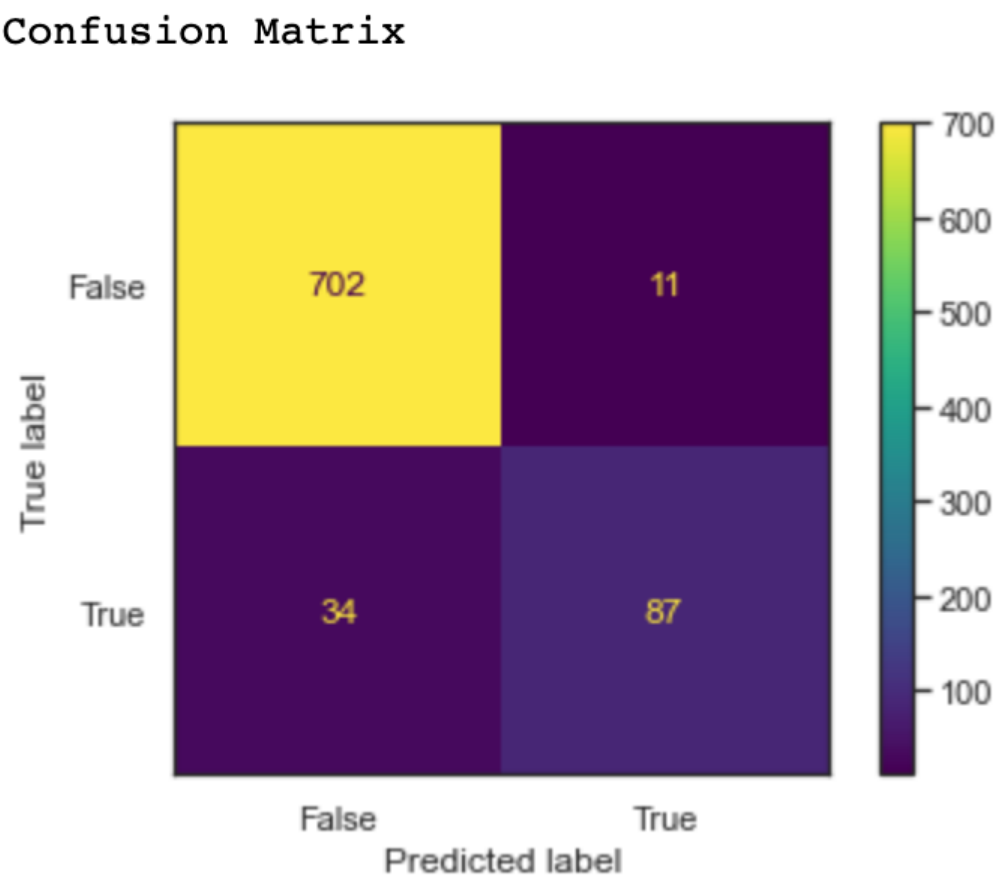
- Training F1-Score: 0.93
- Validation F1-Score: 0.79
- F1-Score Delta: 0.14

Testing Classification Report precision recall f1-score support

```
    False       0.95      0.98      0.97       713
     True       0.89      0.72      0.79       121

 accuracy                           0.95       834
```

macro avg 0.92 0.85 0.88 834 weighted avg 0.94 0.95 0.94 834



Conclusion and Recommendations

The final tuned XGBoost model achieved an F1-score of 0.79, the highest and closest to 1.0 of all the evaluated models. I

README.md

Further analysis indicated that the most important features influencing the model are:

1. **the number of calls the customer made to customer service**
2. whether the customer has a voice mail plan
3. the total number of minutes used per day
4. **whether the customer has an international plan**
5. the total number of international calls made

Both of the bolded features were also influential in other models evaluated, thus I recommend conducting further customer analysis to identify trends regarding calls to customer service – issues, volume, customer service rating. Additionally, based on exploratory analysis, customers with an international plan are more likely to churn than those without an international plan – what about the international plan may be influencing churn?!

For More Information

Please review our full analysis in our Jupyter Notebook or our presentation.

For any additional questions, please contact **Rebecca Frost-Brewer (frostbrewerr@gmail.com)**

Repository Structure

```
├── README.md                    <- The top-level README for reviewers of this project
├── index.ipynb                  <- Narrative documentation of analysis in Jupyter notebook
├── ppt-churn-presentation.pdf   <- PDF version of project presentation
├── results-table.png
├── confusion-matrix.png
├── rfb-headshot.png
└── telecom.csv                  <- Sourced externally
```

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- **Jupyter Notebook** 100.0%