Cpts437 Project Write-Up: Customer Churn Analysis and Prediction

Problem Definition

Customer churn, the phenomenon where customers discontinue their services with a company, poses a significant challenge to businesses in competitive industries. Understanding the factors influencing churn and predicting its likelihood are crucial for retaining customers and sustaining revenue. The dataset provided, *Telco Customer Churn*, offers a comprehensive view of customer demographics, service details, and account information, making it a valuable resource for developing predictive models to address this issue.

The dataset contains 7,043 rows, each representing a unique customer, and 21 features, including the target variable "Churn." Key data points include service subscriptions (e.g., internet, phone, streaming), demographic details (e.g., age, gender, family status), and billing information (e.g., contract type, payment method, charges). These features enable a detailed analysis of the factors contributing to customer attrition.

Objective

The primary goal is to analyze this dataset and build predictive models to classify customers as likely to churn or retain. By identifying high-risk customers, businesses can implement targeted retention strategies, such as personalized offers or improved services, to reduce churn rates effectively.

Challenges

1. Imbalanced Data: Customer churn datasets often have an uneven distribution of the target variable, requiring techniques like oversampling, undersampling, or cost-sensitive modeling.
2. Feature Engineering: Properly encoding categorical variables and handling missing data are crucial for model performance.
3. Model Selection: Choosing and tuning suitable algorithms like logistic regression, random forests, or gradient boosting for high accuracy and interpretability.

Inspiration

This project serves as a learning opportunity to explore machine learning techniques for solving real-world business problems and understanding customer behavior patterns.

Comprehensive Evaluation of Machine Learning Models: Random Forest, Gradient Boosting, and Logistic Regression

Handling Data Imbalance

To address data imbalance, various strategies were applied throughout the evaluation process. Data imbalance can lead to skewed predictions where the model becomes biased toward the majority class, resulting in poor performance for the minority class. To counteract this, techniques such as oversampling and undersampling were utilized. Oversampling the minority class increased its representation in the dataset, thereby providing the model with more balanced exposure to both classes. Conversely, undersampling the majority class involved reducing its representation to level the distribution, which helped to prevent the model from overfitting to the majority class.
In addition to resampling, class weights were adjusted in the model configurations to emphasize the importance of the minority class during training. By setting higher class weights for underrepresented classes, models were incentivized to prioritize their correct classification. This adjustment allowed for more equitable consideration of both classes in the learning process, improving metrics like recall and F1 score, which are crucial for understanding performance in imbalanced scenarios.
Evaluations were conducted with repeated runs to capture the consistency and robustness of the models' performance. This involved training and testing each model multiple times with different data splits and averaging the results to minimize the influence of any one data split or random initialization. The primary metrics used for performance assessment were accuracy, precision, recall, F1 score, and the Area Under the ROC Curve (AUC). Processing times were also recorded to evaluate the computational efficiency of each model configuration.

Evaluation of Random Forest

Random Forest was evaluated using three different parameter configurations, with multiple runs averaged for more reliable metrics. The first configuration used the parameters {'n_estimators': 100, 'max_depth': None, 'min_samples_split': 2}, which resulted in a mean accuracy of 0.9000, precision of 0.9485, recall of 0.8598, F1 score of 0.9020, and an AUC of 0.9379. The average processing time for this configuration was 0.8735 seconds. The second configuration, set at {'n_estimators': 200, 'max_depth': 10, 'min_samples_split': 5}, yielded a mean accuracy of 0.9050, precision of 0.9490, recall of 0.8692, F1 score of 0.9073, and an AUC of 0.9333. This configuration showed a slightly higher processing time at 0.9116 seconds. The third configuration, {'n_estimators': 50, 'max_depth': 5, 'min_samples_split': 2}, demonstrated a mean accuracy of 0.8700, precision of 0.9263, recall of 0.8224, F1 score of 0.8713, and an AUC of 0.9203, with a notably lower average processing time of 0.1551 seconds, making it the fastest.

Evaluation of Gradient Boosting

The Gradient Boosting model was tested with three parameter sets, running each configuration multiple times and averaging the results. The first configuration, {'n_estimators': 100, 'learning_rate': 0.1, 'max_depth': 3}, achieved a mean accuracy of 0.9150, precision of 0.9688, recall of 0.8692, F1 score of 0.9163, and an AUC of 0.9348, with an average processing time of 0.8316 seconds. The second configuration, {'n_estimators': 150, 'learning_rate': 0.05, 'max_depth': 5}, showed a mean accuracy of 0.8900, precision of 0.9570, recall of 0.8318, F1 score of 0.8900, and an AUC of 0.9407, but had a significantly higher average processing time of 2.1172 seconds. The third configuration, {'n_estimators': 100, 'learning_rate': 0.2, 'max_depth': 2}, demonstrated a mean accuracy of 0.8800, precision of 0.9368, recall of 0.8318, F1 score of 0.8812, and an AUC of 0.9220, with an average processing time of 0.6159 seconds.

Evaluation of Logistic Regression
Logistic Regression was tested with three different configurations, averaging results over multiple runs for more reliable comparisons. The first configuration, {'C': 1.0, 'solver': 'lbfgs'}, achieved a mean accuracy of 0.8550, precision of 0.9149, recall of 0.8037, F1 score of 0.8557, and an AUC of 0.9216, with an average processing time of 0.0128 seconds, the shortest among all models. The second configuration, {'C': 0.5, 'solver': 'liblinear'}, produced identical metrics to the first, with an accuracy of 0.8550, precision of 0.9149, recall of 0.8037, F1 score of 0.8557, and an AUC of 0.9214. The processing time was slightly longer at 0.0041 seconds. The third configuration, {'C': 1.5, 'solver': 'newton-cg'}, showed the same results as the first two, maintaining a mean accuracy of 0.8550, precision of 0.9149, recall of 0.8037, F1 score of 0.8557, and an AUC of 0.9218, with an average processing time of 0.0181 seconds.

Comparison of Models
The evaluations showed that Random Forest and Gradient Boosting consistently outperformed Logistic Regression in terms of accuracy, precision, and F1 scores, indicating their robustness in handling the given dataset. Random Forest proved to be a good balance between performance and processing time, while Gradient Boosting delivered slightly better performance metrics but at the cost of increased processing time. Logistic Regression, while computationally efficient and simpler to implement, did not match the performance levels of the tree-based models. The strategy of averaging results across multiple runs ensured that the findings were reliable, presenting a clear comparison of the models' performance. The use of resampling and class weight adjustments was crucial in managing the data imbalance and ensuring that the evaluation metrics were reflective of true model capability, not influenced by class distribution skew.

Conclusion

Random Forest and Gradient Boosting are recommended for customer churn prediction. Gradient Boosting is ideal when precision is paramount, while Random Forest is a better choice for balancing performance and efficiency. Logistic Regression, although simple, may be used as a baseline for further experimentation.

Code Demo:

https://drive.google.com/file/d/1686k5ef8fxy8iyxETcq8CX0Xa1dGAgza/view?usp=sharin
g

Project code:

https://colab.research.google.com/drive/1AWANTiQ0jKMv8HTZajFVb2YWyLmangyV?u
sp=sharing

Class presentation recording:

https://drive.google.com/file/d/1HUk8IMEDw1vUEz7ACpLXMbqDE5z-w4Lp/view?usp=s
haring