Customer Churn in the Telecommunications Sector

Business Problem

The primary objective of this project was to predict customer churn in a telecom dataset using various machine learning techniques. Churn refers to customers who discontinue their service, and accurately predicting this outcome is vital for telecom providers looking to retain users and minimize revenue loss. Given that retaining an existing customer is often significantly cheaper than acquiring a new one, an effective churn prediction model can offer substantial business value.

In this context, the classification task was binary, where the target variable indicated whether a customer had churned (1) or remained (0). Importantly, this problem falls into the category of imbalanced classification, as the proportion of churners in the dataset was around **27%**, while non-churners made up the remaining **73%.** This imbalance required careful handling during model training to avoid biased performance.

From a business standpoint, **recall** (sensitivity) for churners was the most important metric. In other words, we aimed to correctly identify as many actual churners as possible, even if it meant occasionally misclassifying some non-churners (false positives). Missing an actual churner could mean a lost customer and revenue, whereas a false positive might only result in unnecessary outreach.

Modelling Approach

I built and trained each model in a separate notebook to explore them in more depth. This approach allowed me to test different techniques for each model (like SMOTE, SMOTEENN, or different sampling strategies) based on what best suits that algorithm. While some basic steps like importing libraries and splitting data may be repeated, doing so helped me stay focused on one model at a time and improved readability.
In the end, this structure also makes it easier to compare the models and select the best-performing one.
To approach this problem, I chose to build five separate classification models:

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost
- Support Vector Machine (SVM)

Each model was developed independently to ensure full understanding of its behavior, performance, and limitations. This allowed for custom preprocessing, such as applying SMOTE only where needed, using feature scaling selectively, and tuning hyperparameters specifically for each model's strengths. This approach also allowed me to document and interpret each model’s results in greater depth, instead of running all models in a loop and selecting the best by score alone.

Across all models, the dataset was consistently split into training (70%), validation (15%), and test (15%) subsets. I used stratified sampling to ensure that the class distribution was maintained across all splits. The imbalance was addressed using SMOTE, but only on the training data to avoid information leakage.

For evaluation, I used a combination of metrics:
• Recall (For churn class)
• ROC AUC
• F1-Score
• Confusion Matrix

Each model was tuned using GridSearchCV or RandomizedSearchCV with 5-fold cross-validation to identify optimal parameters. Once the best parameters were selected, I evaluated each model on the untouched test set to assess real-world generalization.

In addition, I used feature importance analysis (where applicable) to gain interpretability, and experimented with threshold tuning (in XGBoost) to further boost recall without overly sacrificing precision. **For the XGBoost model, I used both GridSearchCV and RandomizedSearchCV to improve performance, and I performed threshold adjustment for better recall.**

Comparison Before and After Tuning

Model	Before Tuning	After Tuning	Improvement
Logistic Regression	0.8445	0.8448	+0.0003 (Very slight)
Decision Tree	0.6634	0.907	+0.2436 (Huge)
Random Forest	0.8176	0.9262	+0.1086 (Strong)
XGBoost	0.8204	0.9068	+0.0864 (Strong)
SVM (SVC)	0.7863	0.8227	+0.0364 (Moderate)

To get the most out of each algorithm, I decided to build and tune all five models separately instead of putting them in one big loop. This gave me more control to try different preprocessing steps (like SMOTE, scaling, or tuning methods) depending on what worked best for each model.

The table above shows how much each model improved after tuning. As you can see:

- The Decision Tree improved the most, going from a ROC AUC of 0.66 to 0.91 a huge jump.
- Random Forest and XGBoost also performed really well, both crossing 0.90 after tuning.
- SVM showed moderate improvement, while Logistic Regression stayed more or less the same.

By working on each model in its own notebook, I was able to go deeper into testing and tuning. This helped me get better results and understand how each algorithm behaves with this

dataset. That's why models like Random Forest, XGBoost, and Decision Tree ended up performing the best after optimization.

Comparison of All Models

Model	Test Recall (Churn)	Test ROC AUC	Test F1 Score (Churn)	Test Accuracy	Interpretability
Logistic Regression	0.8	0.75	0.76	0.75	High
Decision Tree	0.6	0.76	0.56	0.76	Medium
Random Forest	0.66	0.83	0.6	0.76	Medium
XGBoost	0.83	0.8	0.58	0.68	Low
Supper Vector Machine	0.45	0.81	0.53	0.79	Low

To evaluate the performance of each model, I compared their test set results across key metrics relevant to the business goal of predicting customer churn. The table above summarizes the **recall**, **ROC AUC**, **F1-score**, and **accuracy** for each of the five models. Among these, **Logistic Regression** and **XGBoost** emerged as top contenders.

Logistic Regression achieved the highest F1-score (0.76) and strong recall (0.80), making it both accurate and interpretable. On the other hand, **XGBoost** demonstrated the **highest recall (0.83)**, which is critical in churn prediction where the goal is to correctly identify as many potential churners as possible. However, this came at the cost of a lower accuracy (0.68) and F1-score (0.58), indicating more false positives. In contrast, **Random Forest** provided a balanced trade-off with high AUC (0.83) and decent recall (0.66).

Given the business need to minimize lost customers, **recall was prioritized** in this evaluation. While Logistic Regression offered a more balanced performance, **XGBoost was ultimately selected as the best model** due to its ability to detect a higher proportion of actual churners, despite a drop in overall precision. In high-stakes business decisions like churn prevention, such trade-offs are often acceptable.

It is important to note that performance was assessed using a relatively small test set 15% of total data). While stratified sampling was applied, future work could benefit from larger evaluation samples or repeated validation to confirm stability of results.

Real World And Business Impact

The selected model, XGBoost, could be highly valuable for telecom companies aiming to improve customer retention. By predicting customers likely to churn with over 83% recall, the model enables proactive engagement strategies such as targeted discounts, loyalty offers, or personalized outreach before the customer decides to leave.

Although the model produces a higher number of false positives, this is acceptable in the business context, as the cost of unnecessary outreach is typically much lower than the cost of

losing a customer. Moreover, features like contract type, tenure, and service calls were identified as strong predictors, which can guide further customer service improvements.

The model can be embedded into an internal CRM system or used as part of a scheduled churn-monitoring dashboard, allowing marketing or customer success teams to act on alerts. Regular retraining with updated data would ensure continued accuracy as customer behavior evolves.

Limitations and Future Work

While the project achieved strong predictive results, there are some limitations. First, the dataset was static and relatively small 7,000 rows, which may limit generalizability. The test set was just 15% of the full data; performance could fluctuate with a different test split. Additionally, the model does not explain why customers are churning, which limits the actionability of predictions.

In future iterations, more features could be engineered (e.g., historical trends, complaint logs), and more advanced techniques like ensemble voting, SHAP explainability, or time-aware validation could be explored. Testing on real-time data or across customer segments could also help validate the model's robustness in practical use.

Conclusion

This project aimed to build a machine learning model to predict customer churn in a telecom dataset. After evaluating five models, XGBoost was selected as the best-performing model with a recall of 83% on the test set, making it suitable for identifying potential churners. The model can support proactive retention strategies, improving customer engagement. While results were promising, future improvements such as deeper feature engineering and real-time deployment could further enhance its effectiveness.

References

- Scikit-learn documentation. https:/
- https://colab.research.google.com/drive/16w3TDn_tAku17mum98EWTmjaLHAJcsk0?usp=sharing
- https://xgboost.readthedocs.io/en/stable/