Master in Data Science

Module: Data Science Languages and Tools
Professor: Raffaele Miele
Assignment: Customer Churn Prediction
Team Members:

- Abelardo Garcia - abegar10@live.com
- Roberto Tirado - rtiraccelli@gmail.com
- Gianluca Pio - gianlupio@hotmail.it
- Athanasia Alexiadou - alexiadouathan@gmail.com
- Natalia Krynytska - nataliakrynicka209@gmail.com

Objective

The objective of this assignment was to improve the baseline model trained in class (a Decision Tree Classifier) to better identify customer churn. The dataset was highly imbalanced, and the goal was to use alternative models and preprocessing techniques to improve predictive performance and interpretability.

Dataset Overview

Source: Banking churn dataset with >370,000 records and a binary target variable `flag_request_closure`.
Target Classes: 0 = No Churn (~99.5%), 1 = Churn (~0.5%)
Challenge: Severe class imbalance and presence of high-missing-value features.

Methodology Summary

1. Data Cleaning & Preprocessing:

- Dropped columns with >60% missing values and high-cardinality categoricals.
- Encoded low-cardinality categorical variables.
- Imputed missing numeric fields with the median.

2. Class Imbalance Handling:

- Applied random undersampling to ensure a manageable training distribution (5:1 ratio).

3. Models Evaluated:

- Decision Tree (baseline)
- Random Forest (with balanced class weights)
- Logistic Regression (with balanced class weights)

4. Evaluation Metrics:

- Focused on Recall and AUC for identifying churners, with F1-score and Precision for context.

Performance Comparison

Model	Recall (Churn)	Precision (Churn)	F1-Score	AUC Score
Decision Tree	0.32	0.05	0.08	0.832
Random Forest	0.40	0.07	0.11	0.876
Logistic Regression	0.78	0.01	0.02	0.777

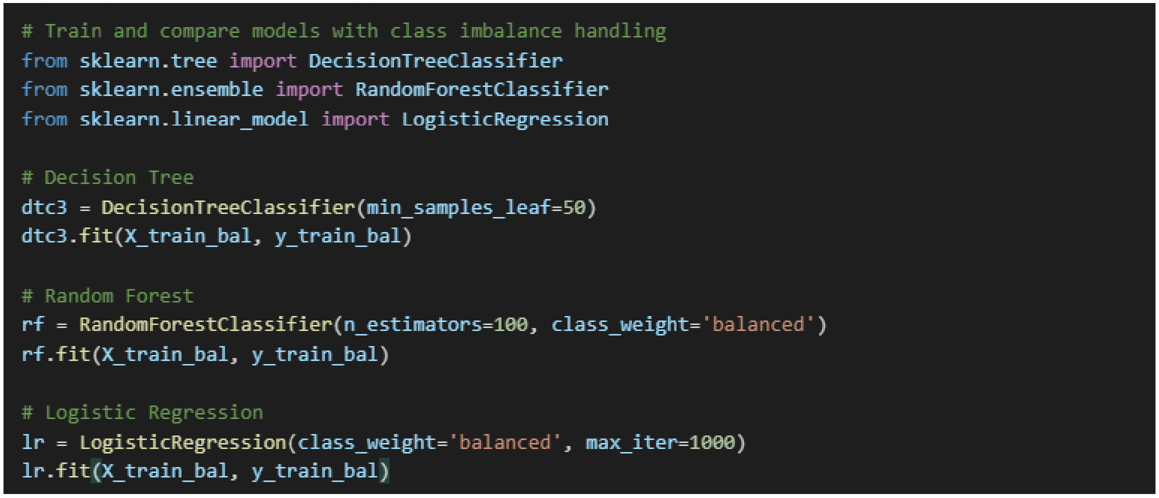
Insights:
Random Forest produced the best AUC score, indicating the strongest ranking ability.
Logistic Regression caught the most churners (recall = 0.78) but at the cost of very low precision.
Decision Tree offered a simple and interpretable baseline but lacked predictive strength.

Conclusion

The final recommendation depends on business goals:
- Use Random Forest when a balanced, high-performing model is needed for risk scoring.
- Use Logistic Regression in recall-critical situations where identifying most churners outweighs precision.

The project demonstrated the value of model comparison, threshold tuning, and the importance of resampling in imbalanced classification. Future improvements could include cost-sensitive learning or ensemble stacking.

Key Implementation Snippet



```
# Train and compare models with class imbalance handling
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

# Decision Tree
dtc3 = DecisionTreeClassifier(min_samples_leaf=50)
dtc3.fit(X_train_bal, y_train_bal)

# Random Forest
rf = RandomForestClassifier(n_estimators=100, class_weight='balanced')
rf.fit(X_train_bal, y_train_bal)

# Logistic Regression
lr = LogisticRegression(class_weight='balanced', max_iter=1000)
lr.fit(X_train_bal, y_train_bal)
```