Data Engineer Report

Churn Predictor Web Application

Vishal Nigam
Department of Mathematics and Geosciences
Data Science and Artificial Intelligence
University of Trieste
vishal.nigam@studenti.units.it

Muhammad Mubashar Shahzad
Department of Mathematics and Geosciences
Scientific and Data Intensive Computing
University of Trieste
mshahzad@sissa.it

Sepehr Davanipur
Department of Mathematics and Geosciences
Data Science and Artificial Intelligence
University of Trieste
Sepehr.davanipur@studenti.units.it

***Abstract*—This report presents a comprehensive overview of the data engineering work involved in building a web-based churn prediction system for credit card customers. The project utilizes a highly rated open dataset from Kaggle, containing behavioral, transactional, and demographic information of banking clients. The aim is to enable proactive retention strategies by identifying customers likely to discontinue their accounts.**

As the Data Engineer, my responsibilities included cleaning and transforming raw data, engineering predictive features, evaluating model performance across several classifiers, and constructing a scalable pipeline to support model deployment. Multiple supervised learning algorithmsincluding logistic regression, random forest, and artificial neural networkswere explored to determine the most effective approach for the binary classification task.

This system forms the analytical core of an interactive web application, allowing real-time churn risk assessment for business decision support.

I. Problem Definition

Customer churn poses a significant challenge in the credit card industry, as retaining clients directly impacts profitability. This project addresses the problem by developing a machine learningbased web application to predict customer attrition using historical behavioral and demographic data.

The primary task is a binary classification: identifying whether a customer will churn. My role as a Data Engineer focused on preparing high-quality, structured data through cleaning, encoding, scaling, and building a preprocessing pipeline that supports model training, experimentation, and deployment.

II. Background

To support effective churn prediction, this project leverages a modern data science stack centered on Python due to its extensive ecosystem and community support. Key libraries included `pandas` and `NumPy` for data manipulation, and `scikit-learn` for preprocessing and classical machine learning models.

Baseline models such as Logistic Regression and Random Forest were developed using `scikit-learn`, which also enabled seamless integration of scaling, encoding, and cross-validation. For deep learning, a feedforward artificial neural network (ANN) was implemented using `TensorFlow/Keras`, chosen for its scalability and deployment readiness.

Version control and collaboration were managed via `Git` and GitHub. Jupyter Notebooks supported iterative experimentation and visualization, with `Seaborn` and `Matplotlib` used to explore data distributions and feature relationships.

This technology stack ensured modular, reproducible workflows that facilitated rapid model experimentation and set the foundation for streamlined deployment.

III. Dataset Information

The dataset, sourced from Kaggle, includes 10,127 customer records with 23 attributes capturing demographics, transaction patterns, and account activity. The target variable, `Attrition_Flag`, is binaryindicating whether a customer has exited the service.

Key preprocessing steps included:

- Removal of the identifier column (`CLIENTNUM`) and two precomputed Naive Bayes outputs.
- One-hot encoding of categorical variables and standardization of numerical features.
- Retention of all numerical fields to allow empirical model evaluation, even in the presence of strong correlations (e.g., between `Credit_Limit` and `Avg_Open_To_Buy`).

The dataset was clean, with no missing values, and well-suited for automated pipeline processing.

IV. Key Performance Indicators (KPIs)

To evaluate model effectiveness in a slightly imbalanced setting, the following key performance indicators (KPIs) were defined:

- **Accuracy:** Measures overall classification correctness.
- **Precision and Recall:** These metrics help assess model performance across the two classesparticularly important when the cost of false negatives is high (i.e., missed churners).
- **F1 Score (Weighted):** Balances precision and recall while accounting for class imbalance.
- **ROC-AUC Score:** Evaluates the models ability to distinguish between classes across various threshold settings.

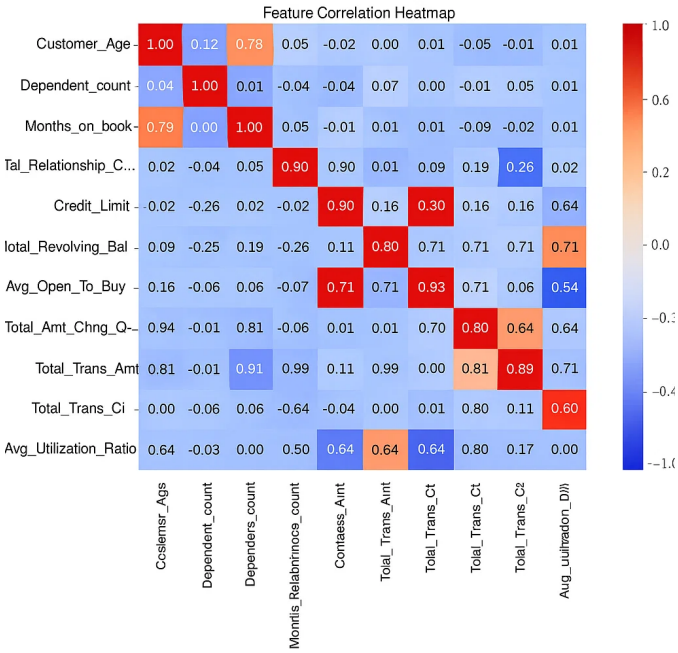

Fig. 1: Component Diagram of the MLOps Pipeline

Table 1:
Performance Metrics of Tuned Models

Model	Best Parameters	Balanced Accuracy	F1 Score (s)
Logistic Regression	C = 1, penalty = l2,	0.755	0.065
Random Forest	solver = lbfgs	0.854	0.855
XGBoost	learning.rate = 0.1, tanh, 3, act2 = relu	0.927	0.927
ANN	Units1 = 48, act1 = tanh, Units2 = 48, act2 = relu, lr = 0.001	0.862	13.785

Fig. 2: Sequence Diagram of Prediction Flow from UI to Model Output

- **Training Time:** Monitored to assess the feasibility of rapid re-training or updates in a production pipeline.

These metrics were used to compare multiple modelsLogistic Regression, Random Forest, and Artificial Neural Networks (ANN)ensuring that the selected model for deployment offered a balanced trade-off between performance and computational efficiency.

V. Data Exploration and Statistical Analysis

Descriptive statistics revealed that customers in the dataset span a broad range of ages, income brackets, and product usage patterns. Most customers have been with the institution for over 30 months, and features such as `Total_Trans_Ct` and `Total_Amt_Chng_Q4_Q1` provide insight into customer engagement and transaction activity.

The correlation matrix analysis highlighted several notable relationships:

- **Near-perfect correlation (0.99)** between `Credit_Limit` and `Avg_Open_To_Buy`, indicating redundancy.
- **Strong correlations** (> 0.80) among behavioral features such as `Total_Trans_Ct`, `Total_Trans_Amt`, and `Total_Ct_Chng_Q4_Q1`, suggesting they capture overlapping dimensions of activity.
- **Minimal correlation** among demographic variables, which were retained to allow the artificial neural network (ANN) to learn non-linear dependencies.

These findings guided feature engineering choices; however, no variables were excluded solely based on correlation strength. Instead, model experimentation was prioritized to evaluate which feature combinations produced optimal predictive performance.

VI. Methodology

A. Pipeline Overview

The goal of this project was to identify the most effective machine learning model for predicting customer churn based on structured credit card usage data. The predictive task is a binary classification problem, where the target variable indicates whether a customer has discontinued their account.

To ensure a robust evaluation, we designed an end-to-end pipeline that included:

- **Data preprocessing:** Cleaning, encoding, and scaling.
- **Feature engineering:** Evaluation and transformation of variables.
- **Model training and selection:** GridSearchCV with StratifiedKFold.
- **Performance evaluation:** Based on balanced accuracy, weighted F1 score, and training time.

We compared four supervised learning algorithms:

- **Logistic Regression (baseline)**
- **Random Forest (RF)**
- **XGBoost (gradient boosting)**
- **Artificial Neural Network (ANN)** chosen as the final model

Each model was trained on the same preprocessed dataset to ensure fair comparison.

1) Data Preprocessing: The dataset contained no missing or duplicate records, simplifying the cleaning process. The `Attrition_Flag` target column was binary-encoded with "Attrited Customer" mapped to 1 and "Existing Customer" to 0.

Categorical features such as `Gender`, `Education_Level`, and `Income_Category` were one-hot encoded, with the first category dropped to avoid multicollinearity. Numerical features were standardized using `StandardScaler` to ensure compatibility with ANN and other gradient-based models. All transformations were implemented via a `ColumnTransformer` pipeline.

After preprocessing, the dataset contained 32 features. It was then split into training (80%) and testing (20%) sets using stratified sampling to preserve class distribution.

2) Feature Engineering: The dataset already included rich behavioral and transactional features. Key indicators like `Total_Ct_Chng_Q4_Q1` and `Total_Amt_Chng_Q4_Q1` captured quarterly dynamics, while `Avg_Utilization_Ratio` described revolving credit usage.

- Redundant columns such as `CLIENTNUM` and two embedded Naive Bayes columns were removed.
- A near-perfect correlation ($\sim$0.99) between `Credit_Limit` and `Avg_Open_To_Buy` was observed, but both features were retained for empirical testing.
- One-hot encoding was applied to all categorical variables with first category drop.

Though no derived features were added in this phase, the pipeline remains modular to accommodate future additions (e.g., `Total_Revolving_Bal / Credit_Limit`). The goal was to prioritize model benchmarking on original features before introducing complexity.

3) Model Selection and Training: To identify the best model, four algorithms were trained and evaluated:

- **Logistic Regression** served as a baseline with minimal training time and modest performance.
- **Random Forest** improved all metrics and remained interpretable.
- **XGBoost** achieved highest accuracy, F1 score, and handled imbalance well.
- **ANN** competitive results with balanced recall and ROC-AUC, but longer training time.

All models were trained on the same transformed data and evaluated on a stratified test set. Metrics included:

- Balanced Accuracy
- Weighted F1 Score
- Precision and Recall
- True/False Positive and Negative Rates
- Specificity and ROC-AUC Score
- Training Time

Performance comparisons informed selection and further tuning phases.

4) Hyperparameter Tuning: Fine-tuning was conducted via:

- **GridSearchCV** for Logistic Regression, Random Forest, and XGBoost.
- **KerasTuner** for ANN.

The objective metric was weighted F1 score, validated via 5-fold stratified cross-validation. The optimized models demonstrated enhanced generalization and accuracy, with selected hyperparameters carried into the final evaluation phase.

VII. Discussion

The entire pipeline was developed in Python using a modular design that ensured reproducibility and flexibility. Data preprocessing was handled with `scikit-learn`'s `ColumnTransformer`, including scaling and encoding. The target variable was binarized, and the dataset was split using stratified sampling to preserve class balance.

Four modelsLogistic Regression, Random Forest, XGBoost, and ANNwere trained and evaluated on the same processed data. Classical models used `StandardScaler` and `joblib` for persistence, while ANN was implemented in `Keras` and saved using the `.keras` format.

Evaluation metrics included Balanced Accuracy, F1 Score, ROC AUC, and Training Time. XGBoost emerged as the top performer, combining high accuracy with efficient training. ANN showed strong results with non-linear data but required more computational resources. Random Forest provided interpretability with competitive performance, and Logistic Regression acted as a lightweight benchmark.

The structured pipeline supports rapid experimentation and lays the groundwork for future MLOps integration.

A. Performance Evaluation

Model performance was evaluated using a diverse set of metrics to ensure robustness, particularly under mild class imbalance conditions. The primary metrics included:

- **Balanced Accuracy:** Accounts for skewed class distribution.

- **Weighted F1 Score:** Balances precision and recall weighted by class frequency.
- **Precision and Recall (TPR):** Class-specific predictive power.
- **Specificity (TNR):** Ability to correctly identify retained customers.
- **False Positive/Negative Rates:** Quantifies classification errors.
- **ROC AUC Score:** Measures separability of churn vs. non-churn cases.
- **Training Time:** Indicates computational efficiency and retraining feasibility.

All models were trained under the same preprocessing pipeline and evaluated on an identical stratified train/test split.

1) Model Comparison:

- **XGBoost** achieved the highest overall performance across all key metrics. It demonstrated strong generalization and managed class imbalance effectively.
- **Artificial Neural Network (ANN)** showed competitive performance, outperforming Random Forest in recall and accuracy but required longer training time.
- **Random Forest** delivered stable and interpretable results, with strong performance across most metrics.
- **Logistic Regression**, while fast and lightweight, performed lowest in predictive metrics but served as a useful baseline for benchmarking.

All final results reflect post-tuning performance, with model configurations optimized using cross-validation.

VIII. Results

This project aimed to predict customer churn using four supervised learning models: Logistic Regression, Random Forest, XGBoost, and an Artificial Neural Network (ANN). After hyperparameter tuning and evaluation, **XGBoost** demonstrated the best overall performance, achieving:

- **Balanced Accuracy:** 92.7%
- **Weighted F1 Score:** 96.7%
- **ROC AUC Score:** 92.7%

XGBoost effectively balanced precision and recall while maintaining computational efficiency, making it the most scalable and robust option for deployment.

The **ANN** model followed closely with:

- **Balanced Accuracy:** 86.2%
- **Weighted F1 Score:** 93.2%

It showed strong capacity to learn complex, non-linear relationships, albeit with higher training time due to its deep architecture.

Random Forest achieved:

- **Balanced Accuracy:** 85.4%

It offered quick training and interpretability, making it a reliable choice in environments favoring explainable models.

Logistic Regression served as a lightweight baseline with minimal training time but showed the lowest predictive power among all models.

All models were trained using stratified splits and evaluated through 5-fold cross-validation to ensure statistical robustness. Final trained models were saved using `joblib` and `.keras` formats for reproducibility and deployment.

A. Future Improvements

Several avenues for future improvement include:

- Integrating model explainability tools such as `SHAP` to improve transparency and trust in predictions.
- Automating periodic retraining with fresh customer data to maintain performance over time.
- Exploring advanced ensemble or meta-learning strategies to further boost predictive accuracy and generalization.

Overall, the results validate XGBoost as the most effective and deployable model in the current architecture, combining accuracy, robustness, and scalability.

References

[1] Kaggle. *Credit Card Customer Churn Prediction Dataset*, 2020. [Online]. Available: https://www.kaggle.com/datasets/sakshigoyal7/credit-card-customers

[2] A. Gron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., OReilly Media, 2019.

[3] C. Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, 2020. [Online]. Available: https://christophm.github.io/interpretable-ml-book/

[4] F. Chollet, *Deep Learning with Python*, Manning Publications, 2018.

[5] XGBoost Developers. *XGBoost Documentation*. [Online]. Available: https://xgboost.readthedocs.io

[6] Keras Developers. *Keras Documentation*. [Online]. Available: https://keras.io/