Artificial Intelligence Solution Implementation

by Maria Ingold

INTRODUCTION

Ethical Bank (EthiBank) would like to explore using machine learning (ML) to predict customer churn—exiting—while upholding transparency, ethics, and industry standards. Following the Cross Industry Standard Process for Data Mining (CRISP-DM) approach, this feasibility study uses a public bank dataset and WEKA software to demonstrate the transferable application, approach, and methods (Niakšu, 2015).

BUSINESS UNDERSTANDING

As an online-only fintech startup founded in 2019, EthiBank serves over 100,000 UK customers including individuals and businesses. It provides ethical online financial services including loans, investments, accounts, and software. However, EthiBank faces declining year-over-year revenue growth and competition from AI-driven rivals. EthiBank's churn rate of about 20% negatively impacts revenue and profitability, with customer acquisition costs about five times retention (de Lima Lemos et al., 2022). Although EthiBank lacks demographic data, it has rich usage, transactional, and engagement data, and research demonstrates ML can effectively predict churn using only core historical banking data (Rahman & Kumar, 2020; de Lima Lemos et al., 2022).

DATA UNDERSTANDING AND PREPARATION

Reflecting EthiBank's 20% churn rate, the imbalanced Akturk (2020) supervised learning dataset contains 10,000 banking clients with a 20% minority class of 2037 churners (*Exited* = 1) (Khoshgoftaar et al., 2007; Witten et al., 2017). This appears to be the same Kaggle dataset analysed by Rahman & Kumar (2020).

Table 1 shows the removal of five of the 14 original attributes following Duboue (2020): features are informative (for human and model), available (not missing), and discriminant (divides instances or correlates to target class). *Geography* was discarded because EthiBank is UK-based, *Gender* because EthiBank does not collect, and three had unique values or low variability (Google, N.D.). Of the remaining nine attributes, the numeric data was reasonable (Tables 2, 3 and 4), however, three attributes and the output class were converted using numericToBinary or numericToNominal (with class first requiring conversion to no-class) (Frank et al., 2016). Reducing features decreases dimensionality which helps shorten training time and simplifies the model (Neal, 2019; Rahman & Kumar, 2020).

Python randomly split the data 90/10 into train/test sets while maintaining the 20% churn ratio (Figures 1, 2 and 3). This allows comparison to Rahman & Kumar's (2020) published benchmarks on similar preprocessed data.

MODELLING

Supervised Learning

Supervised learning trains algorithms on labelled training data (output class) to make classification (finite) or regression (numeric) predictions on test data (Bell, 2020; Russell & Norvig, 2021). To generalise well, models balance simplicity to avoid underfitting (high bias) and complexity to avoid overfitting (high variance) (Neal, 2019). Simpler models are evaluated first (Russell & Norvig, 2021).

For bank churn prediction, a binary classification problem, past studies overlap on comparing *k*-nearest neighbours (KNN), decision tree (DT), support vector machine (SVM) and random forest (RF), with random forest outperforming (Rahman & Kumar, 2020; de Lima Lemos et al., 2022).

All four models solve both classification and regression problems, and are suitable for binary classification, however, while KNN and decision trees are interpretable, SVM and random forest are opaque (Belle & Papantonis, 2021).

Comparing these four algorithms (Table 5), a first run with defaults using single 10-fold cross-validation on the imbalanced training set (Table 6) shows KNN underperforming on accuracy, SVM close to baseline, and random forest ahead on AUC-ROC, the receiver operating characteristic area under the curve, the key measure for imbalanced datasets.

K-Nearest Neighbors

KNN is a simple classifier that stores training examples and classifies new inputs by identifying the k-most similar examples (measured by distance) and assigning the majority label (Aha et al., 1991; Laaksonen & Oja, 1996; Witten et al., 2017). Fast, flexible training is a key advantage over generalised models like decision tree, but testing is slower due to distance calculations (Aha et al., 1991; Witten et al., 2017).

Hyperparameter *k* controls model complexity—too low overfits and too high underfits—and is usually odd to avoid tied votes (Russell & Norvig, 2021). With default *k*=1 underperforming, 10 repetitions of 10-fold cross-validation on the imbalanced training set (Figures 12 and 13) found k=17 optimised 84% accuracy (compared to 78.7%), and *k*=15+ returned AUC-ROC 0.81 (compared to 0.66). The higher *k*=17 generalised better than *k*=5 found by Rahman & Kumar (2020), likely due to preprocessing variations (one less feature) and cross-validation method (10% hold out).

KNN should suit eight features and 9000 training samples, because it is nonparametric (no assumptions about data distribution), and performs well with abundant data in low dimensions (distance becomes less meaningful with the curse of dimensionality) (Laaksonen & Oja, 1996; Russell & Norvig, 2021).

Decision Tree

Decision trees recursively split data on highest information gain—indicating the most important attribute—until classification at a leaf node (Witten et al., 2017; Bell, 2020; Charbuty & Abdulazeez, 2021; Russell & Norvig, 2021). Attribute importance transparency is a key reason for using decision tree. Age was the most important, with NumOfProducts second-most informative (Figure 15).

While decision trees handle large data sets well, unpruned trees overfit, therefore, WEKA's J48 (C4.5) algorithm prunes automatically (Witten et al., 2017; Bell, 2020). The pruned 79-node tree with 42 leaves achieved 85% accuracy on the imbalanced training set (Figure 14), indicating that older customers and those with more products tend to stay, although Figures 8 and 9 show this is taken from relatively small sample sizes. Pruning hyperparameter tests show increased pruning (0.25 to 0.1) improved AUC-ROC slightly (0.793 to 0.806), while reducing to a 53-node tree with 29 leaves that maintained Age and then NumOfProducts' importance (Figures 16-19).

Support Vector Machine

SVM classifies data by finding the maximum margin hyperplane that separates classes (Russell & Norvig, 2021). The hyperplane is equidistant between margins defined by the support vectors—training points nearest the boundary—with a wider margin providing more confidence in generalisation (Bell, 2020). By using support vectors rather than storing all training points (like KNN) SVM resists overfitting (Russell & Norvig, 2021).

Non-linear classification uses the “kernel trick” to map to higher dimensions (Bell, 2020). WEKA’s SVM, Platt’s (1999) Sequential Minimal Optimization (SMO), defaults to PolyKernel.

The linear PolyKernel was faster than non-linear RBFKernel, with 9 million versus 457 million evaluations, but at equal 81.9% accuracy (Figures 22 and 23). However, SVM underperformed, potentially struggling with imbalanced data (Figure 24) (Rahman & Kumar, 2020).

Random Forest

Random forest is an ensemble of decision trees that extend decision tree bagging (bootstrap aggregating) (Russell & Norvig, 2021). Bagging aggregates predictions from *k* trees of *N* random examples, but because information gain often selects the same root, random forest decreases correlation by randomly sampling attributes at each split. They then predict by taking the majority vote (for classification) and averaging (for regression).

While single decision trees require pruning to avoid overfitting, random forest does not, and resists overfitting as more randomised trees are added, although performance plateaus beyond a point (Breiman, 2001).

The default hyperparameters of *k*=100 trees and N=100% bag size (full 9000 training set) achieved 0.83 AUC-ROC using 10x10-fold cross-validation across the imbalanced training set (Figures 25, 26 and 27). Changing *k* and N did not improve performance, so defaults were retained. While slower and less interpretable, random forest has so far performed best.

EVALUATION

Training, Validation and Test

The dataset is assumed to be stationary, and independent and identically distributed (i.i.d.), and the 10,000 was randomly split 90/10 into train and test sets from the same distribution while maintaining the 20% churn ratio (Russell & Norvig, 2021; de Lima Lemos et al., 2022). With 10% held out for testing, *k*-fold cross-validation (*k*=10) was used on the training set for model selection and hyperparameter tuning (Rahman & Kumar, 2020). This allowed each data point to be validated on a different fold, while retaining the full 9000 examples for training. WEKA's stratified cross-validation ensured the churn ratio was consistent across folds (Witten et al., 2017).

To reduce variability, 10 runs of 10-fold cross-validation were averaged when tuning KNN, decision tree and random forest in WEKA Experimenter (Witten et al., 2017; de Lima Lemos et al., 2022). However, single 10-fold cross-validation was used for the final model evaluation in WEKA Explorer as Experimenter does not support separate test sets (Bouckaert et al., 2022).

Evaluation Metrics

Detailed in A.3.1, key evaluation metrics included build speed, accuracy, true positive rate (recall), false positive rate, precision, F-measure, and AUC-ROC (University of Essex Online, N.D.; Witten et al., 2017; Bouckaert et al., 2022). For imbalanced classification, accuracy (higher is better) is less useful than AUC-ROC, with 0.5 random, 0.8 good, and 1.0 perfect.

Additionally, recall (true positive rate) is the percentage of correctly found positive cases, with higher better. False positive rate is the percentage incorrectly predicted as positive, with lower better. Precision is the percentage of correct positive predictions, with higher better. F-measure balances precision and recall, with higher better, and a maximum of one. Confusion matrices visualise model success through high values in the diagonal.

Imbalanced Training Evaluation

On imbalanced data, the baseline of 79.6% just predicts the majority class (Table 9).

After previously tuning KNN's hyperparameter k from 1 to 17, its AUC-ROC exceeded the 0.8 boundary (from 0.659 to 0.808), overtaking decision tree's tuned pruning hyperparameter (which had increased AUC-ROC from 0.793 to 0.806). SVM struggled at 0.565, excelling only at having the lowest minority class false positive rate (Figure 20). Random forest, however, achieved the best AUC-ROC of 0.827 (Figure 25 and Table 9).

Tuned models generalised better, including improving KNN's minority class recall. However, SVM failed to fit the imbalanced boundary. Ultimately, random forest was most robust.

Oversampled Training Evaluation

The "class imbalance problem" shows recall and precision, and thus F-measure, generally performed better for majority than minority in imbalanced data (Table 9) (Mohammed et al., 2020). Synthetic minority oversampling technique (SMOTE) improved majority bias by oversampling training (not test) data to balance the classes (Chawla et al., 2002). The data was then randomised (Figure 29).

After oversampling, accuracy dropped for KNN and SVM, but the minority and majority metrics were more closely balanced, with minority typically performing much better, although the majority class values sometimes decreased (Table 10).

Possibly due to dense synthetic samples, KNN performed slightly better at k=9 versus k=17 on oversampled data. Tuning decision tree pruning made minimal difference, although 0.05 built a simpler tree than 0.1 (Tables 11 and 12). Crucially, however, oversampling made numOfProducts a more important feature than Age (Figure 37).

Random forest outperformed with 0.948 AUC-ROC and 88.24% accuracy. Decision tree was second best, followed by KNN and SVM (Table 13). With AUC-ROC significantly increased for all four models, balancing the training data improved generalisation and minority class performance.

Test Set Results

Comparing test (Table 14) with training (Table 15) results, random forest maintained robust performance on test with AUC-ROC of 0.828 (imbalanced) and 0.843 (oversampled) versus training at 0.827 (imbalanced) and 0.948 (oversampled). This indicates good generalisation and avoidance of overfitting.

After oversampling, decision tree had the highest test accuracy at 85%, however, the oversampled minority class recall improved at the cost of lower precision. The oversampled training data increased the tree size, possibly resulting in overfitting, although it was pruned more aggressively, which could have reduced test precision.

Oversampling reduced KNN’s test AUC-ROC and accuracy, but increased majority class precision and minority class recall and F-measure. Oversampling made it easier to detect minority examples, but also led to more false positives.

SVM consistently failed to generalise well in both training and test with the lowest AUC-ROC for both imbalanced and oversampled data, and for imbalanced data it favoured precision over recall for the minority class.

In summary, random forest generalised best with test performance matching strong training results. With high accuracy, decision tree provided the second-best results.

DEPLOYMENT

Following CRISP-DM, this analysis demonstrated the feasibility of using machine learning for ethical predictive churn modelling (Niakšu, 2015). Like EthiBank data, the public dataset lacked some demographics, but demonstrated ethically evaluating churn with only core banking data (Akturk, 2020). EthiBank's 100,000+ customers with 20% churn could also evaluate undersampling (randomly balancing churners and non-churners) in addition to oversampling (Rahman & Kumar, 2020; de Lima Lemos et al., 2022).

Both this analysis and published research confirm random forest's top performance for bank churn modelling (Rahman & Kumar, 2020; de Lima Lemos et al., 2022). With cloud infrastructure, 100,000+ trees can train quickly using parallel processing (Russell & Norvig, 2021).

An Application Programming Interface (API) can integrate predictions with EthiBank's customer management system to automate identifying high churn risk customers for targeted retention campaigns (Bloch, 2006). Local Interpretable Model-Agnostic Explanations (LIME) techniques can increase trust and transparency, and corroborating random forest with highly transparent decision tree results may help assure stakeholders (Belle & Papantonis, 2021).

Before full deployment, further real-world testing on EthiBank data with a pilot group is advised. Ongoing monitoring tracks performance metrics to maintain effectiveness (IBM, N.D.). Periodic retraining on new customer data maintains accuracy, although if the focus is on high-value customers first, tweaks may be needed once a churn threshold is achieved.

In summary, this study demonstrated using industry standards and transparent, ethical machine learning to minimise churn and increase EthiBank’s revenue. Next steps include further planning, costing and validation—but the concept shows strong potential.

REFERENCES

Aha, D., Kibler, D., Albert M., & Quinian J. (1991) Instance-Based Learning Algorithms 6: 37–66.

Akturk, M. (2020) *Churn for Bank Customers*. Available from: https://www.kaggle.com/datasets/mathchi/churn-for-bank-customers [Accessed 20 October 2023].

Bell, J. (2020) *Machine Learning: Hands-On for Developers and Technical Professionals*. Wiley. DOI: https://doi.org/10.1002/9781119642183.

Belle, V. & Papantonis, I. (2021) Principles and Practice of Explainable Machine Learning, *Frontiers in Big Data*. Frontiers Media S.A. DOI: https://doi.org/10.3389/fdata.2021.688969.

Bloch, J. (2006) How to design a good API and why it matters 506–507. DOI: https://doi.org/10.1145/1176617.1176622.

Bouckaert, R., Frank, E., Hall, M., Kirkby, R., Reutemann, P., Seewald, A. & Scuse, D. (2022) *WEKA Manual for Version 3-8-6*. Available from: https://waikato.github.io/weka-wiki/documentation/ [Accessed 24 October 2023].

Breiman, L. (2001) Random Forests, *Machine Learning* 45: 5–32. DOI: https://doi.org/10.1023/A:1010933404324.

Charbuty, B. & Abdulazeez, A. (2021) Classification Based on Decision Tree Algorithm for Machine Learning, *Journal of Applied Science and Technology Trends* 2(01): 20–28. DOI: https://doi.org/10.38094/jastt20165.

Chawla, N., Bowyer, K. and Kegelmeyer, W. (2002) SMOTE: Synthetic Minority Over-sampling Technique, *Journal of Artificial Intelligence Research* 16: 321–357. DOI: https://doi.org/10.1613/JAIR.953.

Duboue, P. (2020) *The Art of Feature Engineering*, *The Art of Feature Engineering*. Cambridge University Press. DOI: https://doi.org/10.1017/9781108671682.003.

Frank, E., Hall, M., Witten, I. & Kaufmann, M. (2016) *WEKA Workbench Online Appendix for 'Data Mining: Practical Machine Learning Tools and Techniques'*. Available from: https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf [Accessed 5 September 2023].

Google (N.D.) *Representation: Qualities of Good Features*. Available from: https://developers.google.com/machine-learning/crash-course/representation/qualities-of-good-features [Accessed 23 October 2023].

IBM (N.D.) *Planning Monitoring and Maintenance - IBM Documentation*. Available from: https://www.ibm.com/docs/en/spss-modeler/18.2.0?topic=deployment-planning-monitoring-maintenance [Accessed 29 October 2023].

Khoshgoftaar, T., Seiffert, C., Hulse, J., Napolitano, A., & Folleco, A. (2007) Learning with Limited Minority Class Data. DOI: https://doi.org/10.1109/ICMLA.2007.76.

Laaksonen, J. & Oja, E. (1996) Classification with Learning k-Nearest Neighbors, *Proceedings of International Conference on Neural Networks (ICNN'96)* 3: 1480–1483. DOI: https://doi.org/10.1109/ICNN.1996.549118.

de Lima Lemos, R.A., Silva, T.C. & Tabak, B.M. (2022) Propension to customer churn in a financial institution: a machine learning approach, *Neural Computing and Applications* 34: 11751–11768. DOI: https://doi.org/10.1007/s00521-022-07067-x.

Mohammed, R., Rawashdeh, J. & Abdullah, M. (2020) Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results, *2020 11th International Conference on Information and Communication Systems, ICICS 2020* 243–248. DOI: https://doi.org/10.1109/ICICS49469.2020.239556.

Neal, B. (2019) *On the Bias-Variance Tradeoff : Textbooks Need an Update*. MSc thesis. Université de Montréal. DOI: https://doi.org/https://doi.org/10.48550/arXiv.1912.08286.

Niakšu, O. (2015) CRISP Data Mining Methodology Extension for Medical Domain, *Baltic J Modern Computing* 3(2): 92–109.

Platt, J.C. (1999) Fast Training of SVMs Using Sequential Minimal Optimization, in *Advances in Kernel Methods* 185–208.

Rahman, M. & Kumar, V. (2020) Machine Learning Based Customer Churn Prediction In Banking, *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)* 1196–1201. DOI: https://doi.org/10.1109/ICECA49313.2020.9297529.

Russell, S. & Norvig, P. (2021) *Artificial Intelligence: A Modern Approach, Global Edition*. 4th ed. Pearson Education, Limited.

University of Essex Online (N.D.) *Evaluation of Learning Systems*. Available from: https://www.mycourse.co.uk/Computing/AI/UAI/UAI%20Lecturecast%205/content/index.html#/ [Accessed 7 October 2023].

Witten, I.H., Frank, E. & Pal, C. (2017) *Data Mining: Practical Machine Learning Tools and Techniques*. 4th ed. Elsevier.

APPENDIX

A.1 Dataset

https://www.kaggle.com/datasets/mathchi/churn-for-bank-customers

TABLE 1 | Raw dataset attributes and preprocessed type and action. (14 original attributes, with 8 retained features and 1 output class)

Attribute	Description	Type	Preprocessing
RowNumber	1 to 10000	Numeric	Removed
CustomerId	Unique random ID	Numeric	Removed
Surname	Last name	String	Removed
CreditScore	Credit score	Numeric	
Geography	Germany or France	Nominal	Removed
Gender	Male or Female	Nominal	Removed
Age	Customer age	Numeric	
Tenure	Years with bank	Numeric	
Balance	Customer bank balance	Numeric	
NumOfProducts	Number bank products	Nominal	numericToNominal
HasCrCard	Credit card (1 = yes)	Nominal (Binary)	numericToBinary
IsActiveMember	Active (1 = yes)	Nominal (Binary)	numericToBinary
EstimatedSalary	Salary estimate (USD)	Numeric	
Exited	Class (1 = churned)	Nominal (Binary)	numericToBinary

TABLE 2 | WEKA's min, max, mean, and standard deviation of 10,000 full dataset

Feature Name	Min	Max	Mean	Standard Deviation
CreditScore	350	850	650.529	96.653
Age	18	92	38.922	10.488
Tenure	0	10	5.013	2.892
Balance	0	250,898.09	76,485.889	62,397.405
NumOfProducts	1	4	1.53	0.582
HasCrCard	0	1	0.706	0.456
IsActiveMember	0	1	0.515	0.5
EstimatedSalary	11.58	199,992.48	100,090.24	57,510.493
Exited	**0**	**1**	**0.204**	**0.403**

TABLE 3 | WEKA's min, max, mean, and standard deviation of 9000 training set

Feature Name	Min	Max	Mean	Standard Deviation
CreditScore	350	850	651.326	96.35
Age	18	92	38.875	10.449
Tenure	0	10	5.014	2.893
Balance	0	250,898.09	76,591.219	62,404.091
NumOfProducts	1	4	1.53	0.58
HasCrCard	0	1	0.705	0.456
IsActiveMember	0	1	0.514	0.5
EstimatedSalary	11.58	199,992.48	99,988.897	57,495.602
Exited	**0**	**1**	**0.204**	**0.403**

TABLE 4 | WEKA's min, max, mean, and standard deviation of 1000 test set

Feature Name	Min	Max	Mean	Standard Deviation
CreditScore	358	850	643.355	99.105
Age	18	79	39.343	10.83
Tenure	0	10	5.003	2.887
Balance	0	238,387.56	75,537.918	62,360.382
NumOfProducts	1	4	1.534	0.594
HasCrCard	0	1	0.707	0.455
IsActiveMember	0	1	0.528	0.499
EstimatedSalary	91.75	199,454.37	101,002.324	57,665.143
Exited	**0**	**1**	**0.203**	**0.402**

```python
import pandas as pd

df = pd.read_csv('churn_preprocess_2_split.csv')

total_pos = df[df['Exited']==1].shape[0]
total_neg = df[df['Exited']==0].shape[0]
total = total_pos + total_neg

pos_ratio = total_pos / total
neg_ratio = total_neg / total

test_size = 1000

num_pos = int(pos_ratio * test_size)
num_neg = test_size - num_pos

pos_sample = df[df['Exited']==1].sample(n=num_pos)
neg_sample = df[df['Exited']==0].sample(n=num_neg)

test = pd.concat([pos_sample, neg_sample])
train = df.drop(test.index)

print("Total positives:", total_pos)
print("Total negatives:", total_neg)
print("Test set size:", len(test))
print("Positives in test:", num_pos)
print("Negatives in test:", num_neg)

train.to_csv('churn_train.csv', index=False)
test.to_csv('churn_test.csv', index=False)
```

FIGURE 1 | Train / Test Split (Python and pandas)

```
Total positives: 2037
Total negatives: 7963
Test set size: 1000
Positives in test: 203
Negatives in test: 797
```

FIGURE 2 | Churn and test imbalance ratios

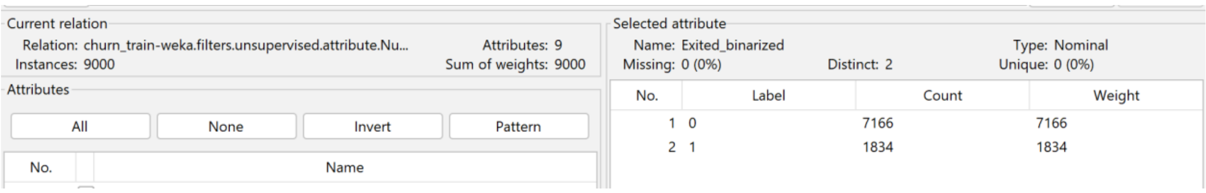

FIGURE 3 | 9000 training instances with 80/20 majority/minority class ratio

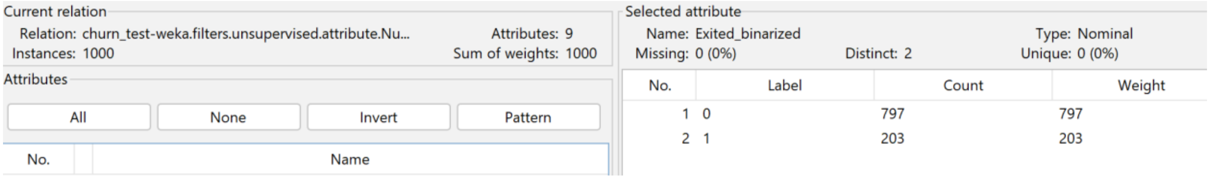

FIGURE 4 | 1000 test instances with 80/20 majority/minority class ratio

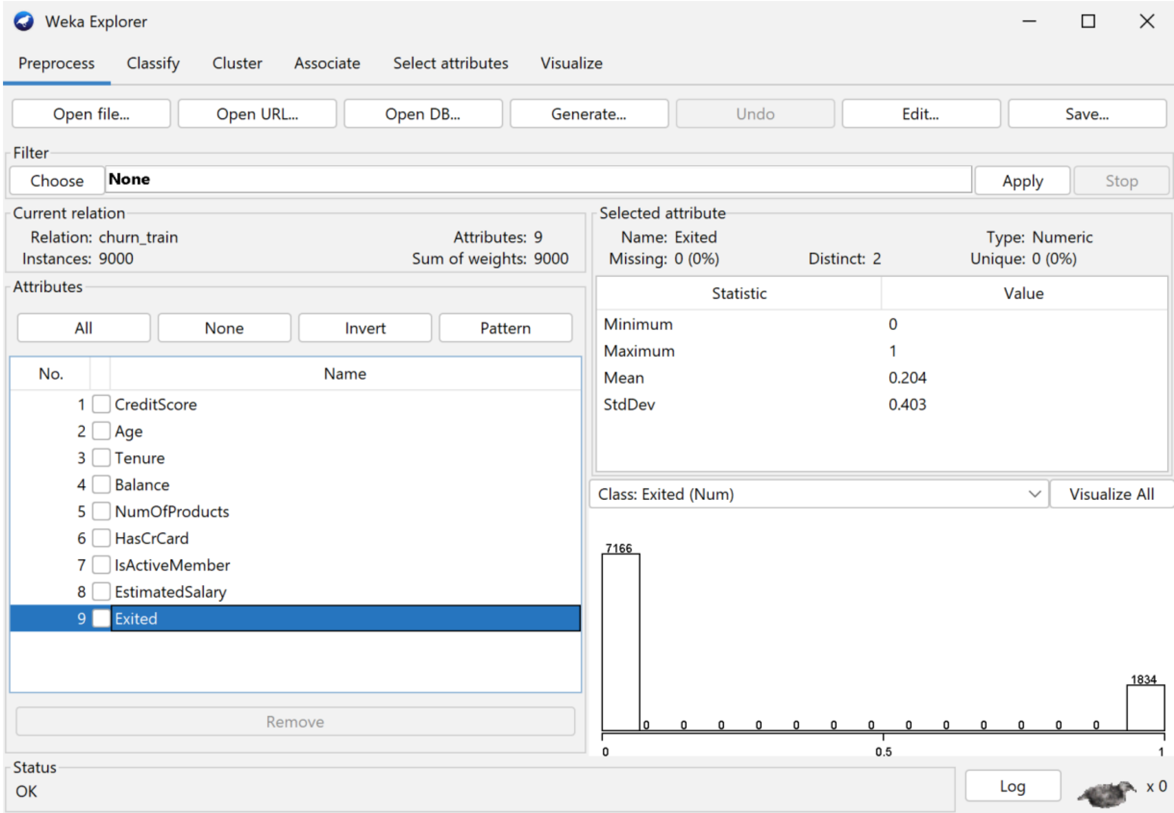

FIGURE 5 | Raw dataset before conversion (class is numeric)

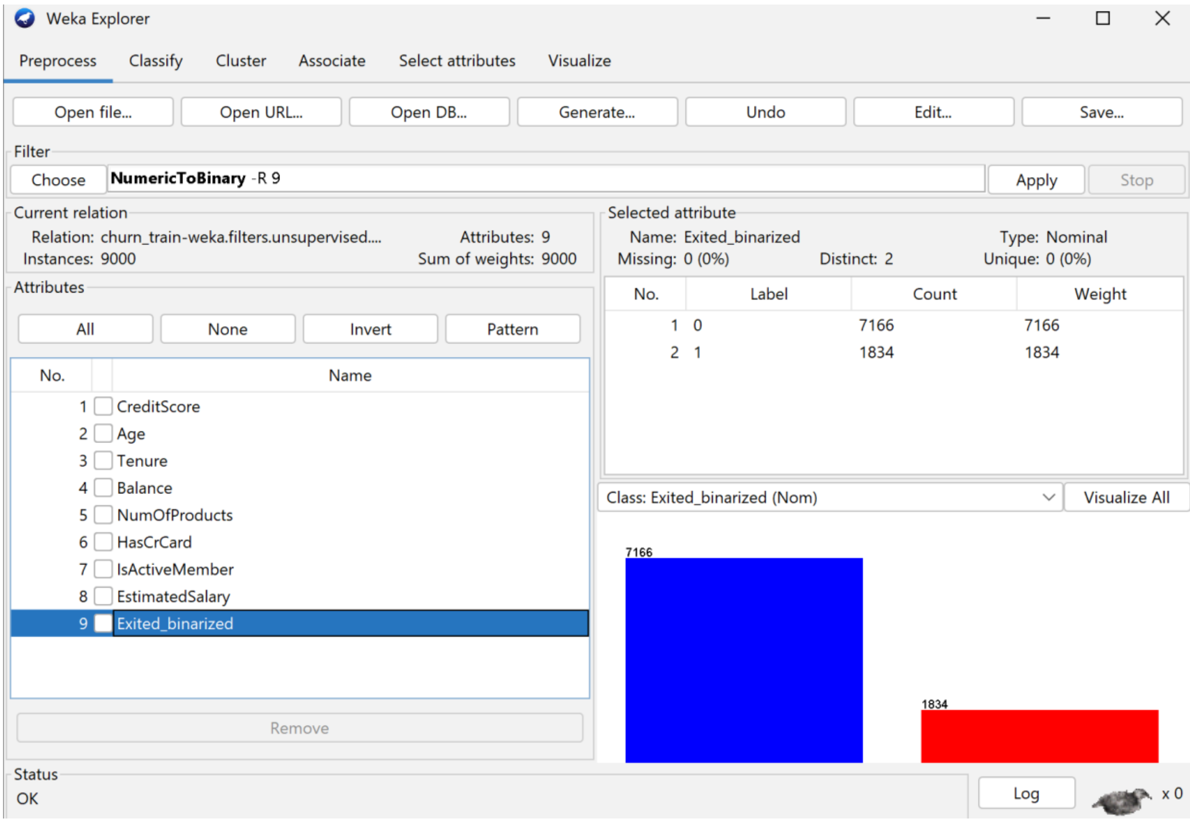

FIGURE 6 | Class after binarization using NumericToBinary (must change class to "No class" first then change back, other fields can be converted directly)

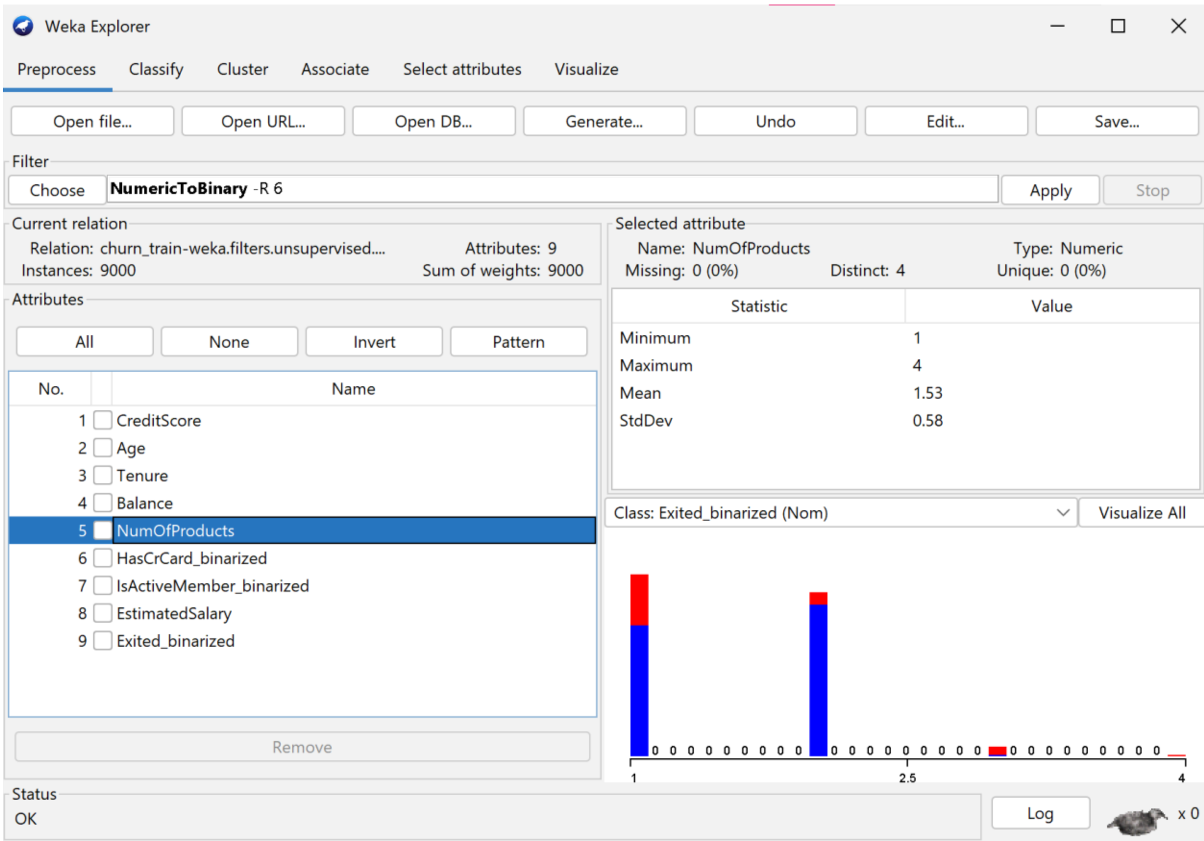

FIGURE 7 | NumOfProducts as a numeric

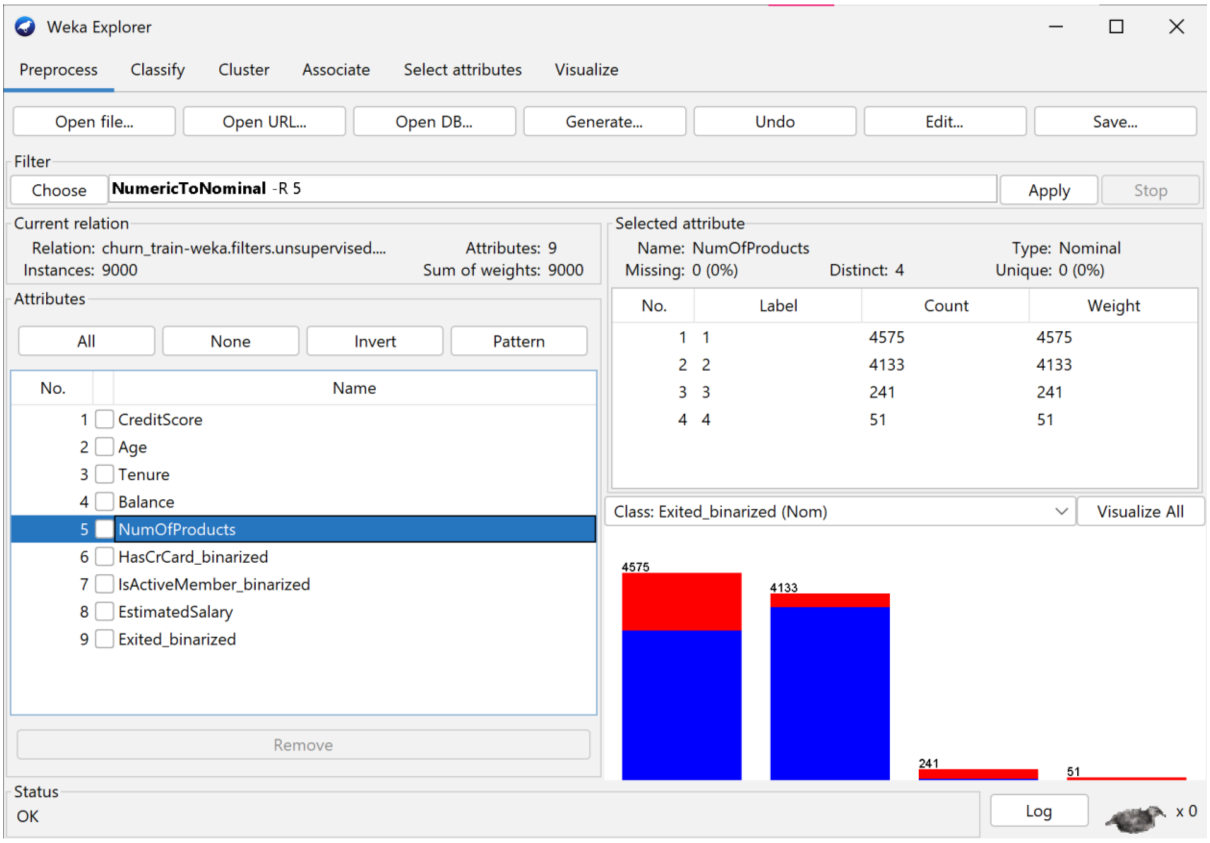

FIGURE 8 | NumOfProducts after Nominalisation

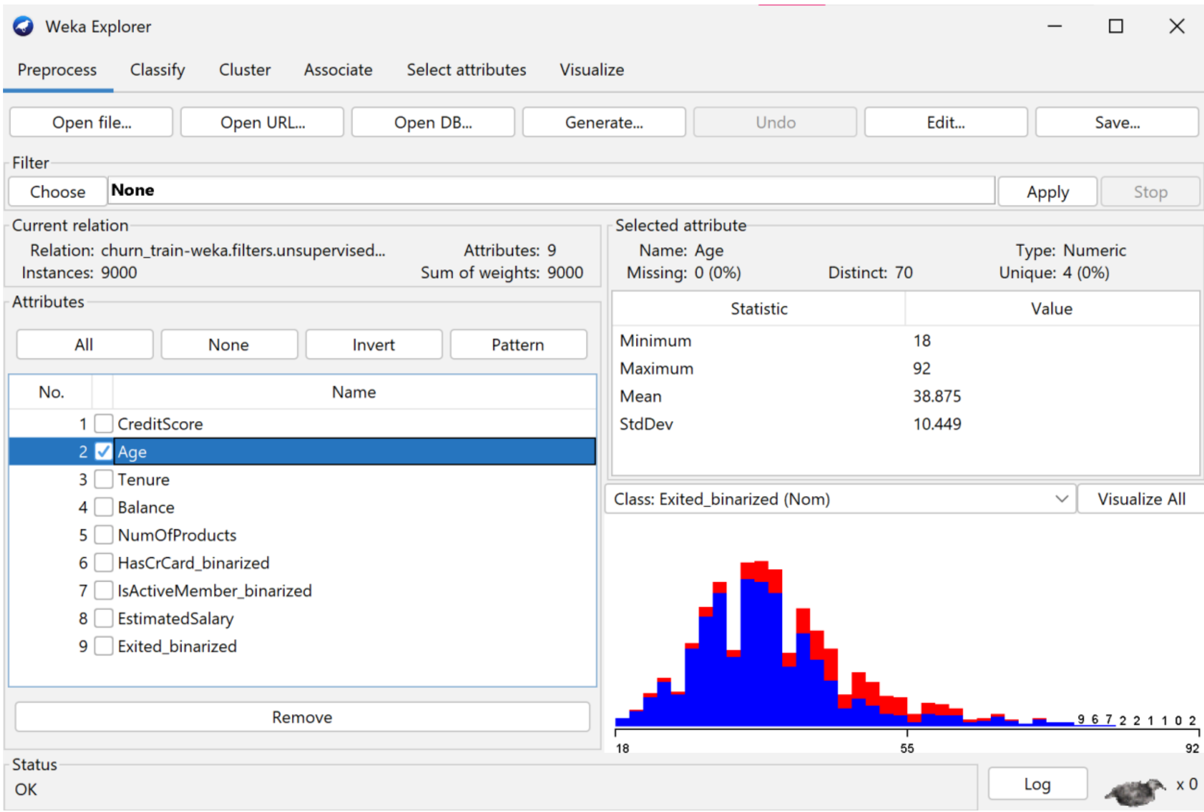

FIGURE 9 | Age

A.2 Modelling Outputs

A.2.1 Algorithms

TABLE 5 | Model and WEKA algorithm

Model	WEKA	Description
k-nearest neighbour (KNN)	IBk	Instance-Based learner
decision tree (DT)	J48	C4.5
support vector machine (SVM)	SMO	Sequential Minimal Optimisation
random forest (RF)	RandomForest	Random forest

A.2.2 Initial Sense Check with Default Parameters

TABLE 6 | Initial sense check using single 10-fold cross-validation with imbalanced training set (9000) and default model parameters (majority class 0=retained; minority class 1=churned)

Model	Build (Sec)	Accuracy	Kappa	Class	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
Baseline	0.01	79.62%	0	0	1.000	1.000	0.796	1.000	0.887	0.499
				1	0.000	0.000	?	0.000	?	0.499
KNN	**0.01**	78.31%	0.3226	0	0.868	**0.550**	0.860	0.868	0.864	0.659
				1	**0.450**	0.132	0.467	**0.450**	0.458	0.659
DT	0.17	**85.07%**	0.4417	0	**0.967**	0.603	0.862	**0.967**	**0.912**	0.793
				1	0.397	0.033	0.753	0.397	0.520	0.793
SVM	1.42	81.91%	0.1888	0	0.994	0.864	0.818	0.994	0.897	0.565
				1	0.136	**0.006**	**0.853**	0.136	0.234	0.565
RF	2.31	85.04%	**0.4595**	0	0.957	0.565	**0.869**	0.957	0.911	**0.827**
				1	0.435	0.043	0.720	0.435	**0.542**	**0.827**

A.2.3 Baseline (ZeroR)

```
Correctly Classified Instances        7166               79.6222 %
Incorrectly Classified Instances      1834               20.3778 %
Kappa statistic                          0
Mean absolute error                      0.3245
Root mean squared error                  0.4028
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.796      1.000    0.887      ?        0.499     0.796     0
                 0.000    0.000    ?          0.000    ?          ?        0.499     0.204     1
Weighted Avg.    0.796    0.796    ?          0.796    ?          ?        0.499     0.675

=== Confusion Matrix ===

    a    b   <-- classified as
 7166    0 |    a = 0
 1834    0 |    b = 1
```

FIGURE 10 | Baseline (ZeroR) (default model parameters, single 10-fold cross-validation, imbalanced training set)

A.2.4 *K*-Nearest Neighbors (IBk)

```
Correctly Classified Instances        7048               78.3111 %
Incorrectly Classified Instances      1952               21.6889 %
Kappa statistic                          0.3226
Mean absolute error                      0.217
Root mean squared error                  0.4657
Relative absolute error                 66.8495 %
Root relative squared error            115.6031 %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.868    0.550    0.860      0.868    0.864      0.323    0.659     0.852     0
                 0.450    0.132    0.467      0.450    0.458      0.323    0.659     0.322     1
Weighted Avg.    0.783    0.465    0.780      0.783    0.782      0.323    0.659     0.744

=== Confusion Matrix ===

    a    b   <-- classified as
 6223  943 |    a = 0
 1009  825 |    b = 1
```

FIGURE 11 | KNN (default model parameters, single 10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPlainText -
Analysing:  Percent_correct
Datasets:   1
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       27/10/2023, 00:47

Dataset                  (1) lazy.IBk | (2) lazy. (3) lazy. (4) lazy. (5) lazy. (6) lazy. (7) lazy.
----------------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)   78.67 |   83.07 v   83.72 v   83.94 v   83.97 v   83.90 v   83.76 v
----------------------------------------------------------------------------------------------------
                               (v/ /*) |   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)


Key:
(1) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(2) lazy.IBk '-K 5 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(3) lazy.IBk '-K 9 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(4) lazy.IBk '-K 15 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(5) lazy.IBk '-K 17 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(6) lazy.IBk '-K 19 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(7) lazy.IBk '-K 29 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
```

FIGURE 12 | KNN—Accuracy—comparing k = 1, 5, 9, 15, 17, 19, 29, with 17 best for accuracy. v indicates that against *k* = 1 at 5% statistical significance, the others are statistically better. (10x10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPlainText -
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       27/10/2023, 00:48

Dataset                  (1) lazy.IB | (2) lazy (3) lazy (4) lazy (5) lazy (6) lazy (7) lazy
---------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)    0.66 |   0.77 v   0.79 v   0.81 v   0.81 v   0.81 v   0.81 v
---------------------------------------------------------------------------------------------
                              (v/ /*) |  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)


Key:
(1) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(2) lazy.IBk '-K 5 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(3) lazy.IBk '-K 9 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -308018
(4) lazy.IBk '-K 15 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(5) lazy.IBk '-K 17 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(6) lazy.IBk '-K 19 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
(7) lazy.IBk '-K 29 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"' -30801
```

FIGURE 13 | KNN—AUC ROC—comparing k = 1, 5, 9, 15, 17, 19, 29, with 15 and above best for AUC ROC. v indicates that against *k* = 1 at 5% statistical significance, the others are statistically better. (10x10-fold cross-validation, imbalanced training set)

A.2.5 Decision Tree (J48)

```
Number of Leaves  :     42

Size of the tree :      79


Time taken to build model: 0.17 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7656               85.0667 %
Incorrectly Classified Instances      1344               14.9333 %
Kappa statistic                          0.4417
Mean absolute error                      0.2236
Root mean squared error                  0.344
Relative absolute error                 68.911  %
Root relative squared error             85.3904 %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.967    0.603    0.862      0.967    0.912      0.473    0.793     0.910     0
                 0.397    0.033    0.753      0.397    0.520      0.473    0.793     0.578     1
Weighted Avg.    0.851    0.487    0.840      0.851    0.832      0.473    0.793     0.842

=== Confusion Matrix ===

    a    b   <-- classified as
 6927  239 |    a = 0
 1105  729 |    b = 1
```

FIGURE 14 | Decision tree (J48) (default hyperparameters, single 10-fold cross-validation, imbalanced training set)

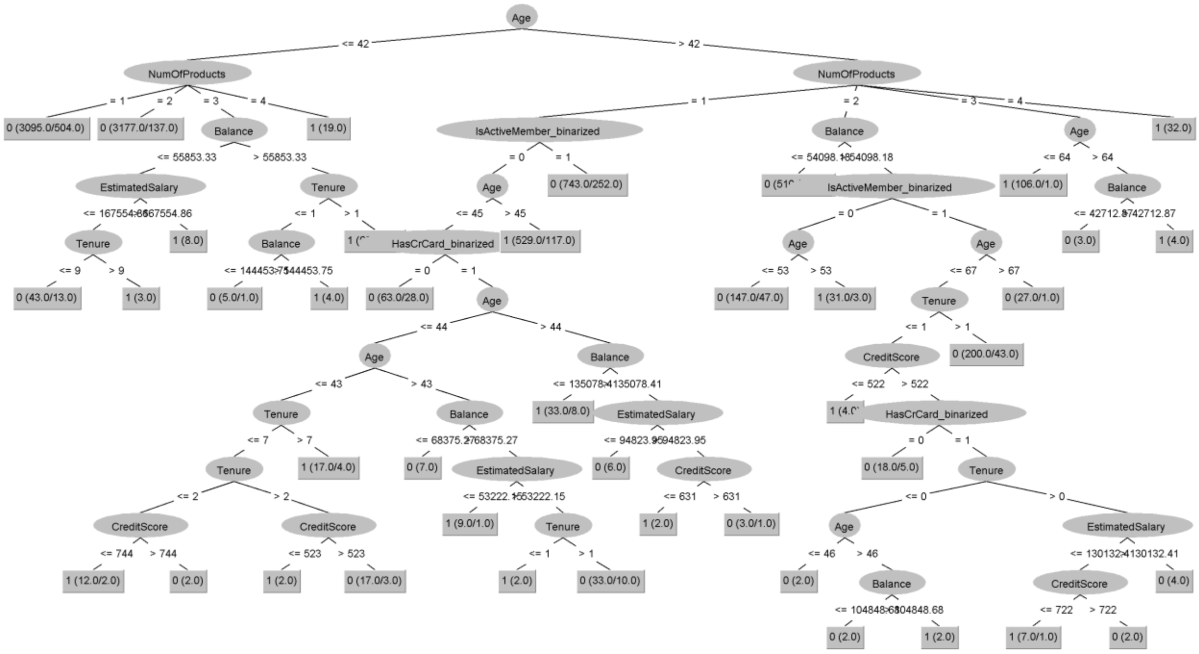

FIGURE 15 | Decision tree showing Age and then NumOfProducts as most important attributes. (default hyperparameters, single 10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrix
Analysing:  Percent_correct
Datasets:   1
Resultsets: 8
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       27/10/2023, 22:03


Dataset                   (1) trees.J4 | (2) trees (3) trees (4) trees (5) trees (6) trees (7) trees (8) trees
------------------------------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)   85.14 |   85.15     85.14     85.35     85.40     85.37     85.29     84.90 *
------------------------------------------------------------------------------------------------------------------
                               (v/ /*) |   (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)   (0/0/1)


Key:
(1) trees.J48 '-C 0.25 -M 2' -217733168393644444
(2) trees.J48 '-C 0.001 -M 2' -217733168393644444
(3) trees.J48 '-C 0.01 -M 2' -217733168393644444
(4) trees.J48 '-C 0.025 -M 2' -217733168393644444
(5) trees.J48 '-C 0.05 -M 2' -217733168393644444
(6) trees.J48 '-C 0.1 -M 2' -217733168393644444
(7) trees.J48 '-C 0.15 -M 2' -217733168393644444
(8) trees.J48 '-C 0.35 -M 2' -217733168393644444
```

FIGURE 16 | Decision tree—Accuracy—comparing confidenceFactor default 0.25 (smaller value prune mores) (10x10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixP
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 8
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       27/10/2023, 22:03


Dataset                   (1) trees.J | (2) tree (3) tree (4) tree (5) tree (6) tree (7) tree (8) tree
------------------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)   0.80 |   0.80     0.80     0.81     0.81     0.81     0.81     0.80
------------------------------------------------------------------------------------------------------
                              (v/ /*) |  (0/1/0)  (0/1/0)  (0/1/0)  (0/1/0)  (0/1/0)  (0/1/0)  (0/1/0)


Key:
(1) trees.J48 '-C 0.25 -M 2' -217733168393644444
(2) trees.J48 '-C 0.001 -M 2' -217733168393644444
(3) trees.J48 '-C 0.01 -M 2' -217733168393644444
(4) trees.J48 '-C 0.025 -M 2' -217733168393644444
(5) trees.J48 '-C 0.05 -M 2' -217733168393644444
(6) trees.J48 '-C 0.1 -M 2' -217733168393644444
(7) trees.J48 '-C 0.15 -M 2' -217733168393644444
(8) trees.J48 '-C 0.35 -M 2' -217733168393644444
```

FIGURE 17 | Decision tree—AUC-ROC—comparing confidenceFactor default 0.25 (smaller value prunes more) (10x10-fold cross-validation, imbalanced training set)

```
Number of Leaves  :     29

Size of the tree :      53


Time taken to build model: 0.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7687               85.4111 %
Incorrectly Classified Instances      1313               14.5889 %
Kappa statistic                          0.4493
Mean absolute error                      0.224
Root mean squared error                  0.3387
Relative absolute error                 69.0107 %
Root relative squared error             84.088  %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.972    0.605    0.863      0.972    0.914      0.486    0.806     0.921     0
                 0.395    0.028    0.780      0.395    0.525      0.486    0.806     0.607     1
Weighted Avg.    0.854    0.487    0.846      0.854    0.835      0.486    0.806     0.857

=== Confusion Matrix ===

    a    b   <-- classified as
 6962  204 |    a = 0
 1109  725 |    b = 1
```

FIGURE 18 | Decision tree—AUC-ROC—confidenceFactor (smaller value prunes more—0.1) (10-fold cross-validation, imbalanced training set)

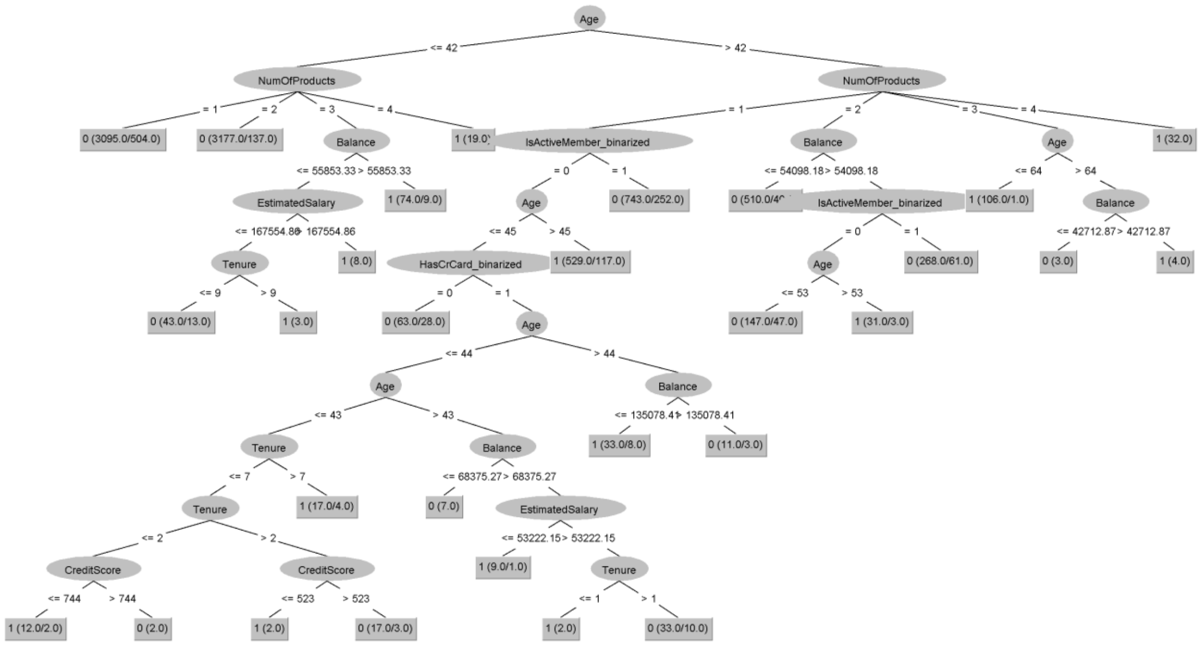

FIGURE 19 | Decision tree after further pruning with hyperparameter tuning (confidenceFactor 0.1) reduced nodes to 53 and leaves to 29 (10-fold cross-validation, imbalanced training set)

A.2.6 Support Vector Machine (SMO)

```
Correctly Classified Instances        7372               81.9111 %
Incorrectly Classified Instances      1628               18.0889 %
Kappa statistic                          0.1888
Mean absolute error                      0.1809
Root mean squared error                  0.4253
Relative absolute error                 55.7356 %
Root relative squared error            105.587  %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.994    0.864    0.818      0.994    0.897      0.295    0.565     0.818     0
                 0.136    0.006    0.853      0.136    0.234      0.295    0.565     0.292     1
Weighted Avg.    0.819    0.689    0.825      0.819    0.762      0.295    0.565     0.711

=== Confusion Matrix ===

    a    b   <-- classified as
 7123   43 |    a = 0
 1585  249 |    b = 1
```

FIGURE 20 | SMV (SMO) (default model parameters, single 10-fold cross-validation, imbalanced training set)

```
Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: 0, 1

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0      * (normalized) CreditScore
 +       0.0008 * (normalized) Age
 +      -0.0003 * (normalized) Tenure
 +      -0.0004 * (normalized) Balance
 +      -1.0002 * (normalized) NumOfProducts=1
 +      -0.9999 * (normalized) NumOfProducts=2
 +       0.9997 * (normalized) NumOfProducts=3
 +       1.0004 * (normalized) NumOfProducts=4
 +       0.0002 * (normalized) HasCrCard_binarized=1
 +       0      * (normalized) IsActiveMember_binarized=1
 +      -0.0003 * (normalized) EstimatedSalary
 +       0.0001

Number of kernel evaluations: 9021283 (44.114% cached)
```

FIGURE 21 | SVM using Linear Kernel. Attribute weights. (single 10-fold cross-validation, imbalanced training set)

```
Number of kernel evaluations: 9021283 (44.114% cached)



Time taken to build model: 1.4 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7372               81.9111 %
Incorrectly Classified Instances      1628               18.0889 %
Kappa statistic                          0.1888
Mean absolute error                      0.1809
Root mean squared error                  0.4253
Relative absolute error                 55.7356 %
Root relative squared error            105.587  %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.994    0.864    0.818      0.994    0.897      0.295    0.565     0.818     0
                 0.136    0.006    0.853      0.136    0.234      0.295    0.565     0.292     1
Weighted Avg.    0.819    0.689    0.825      0.819    0.762      0.295    0.565     0.711

=== Confusion Matrix ===

    a    b   <-- classified as
 7123   43 |    a = 0
 1585  249 |    b = 1
```

FIGURE 22 | SVM using Linear Kernel. Fast to build and lower number of kernel evaluations for the same accuracy. (single 10-fold cross-validation, imbalanced training set)

```
Number of support vectors: 4029

Number of kernel evaluations: 456870957 (4.583% cached)



Time taken to build model: 66.14 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7372               81.9111 %
Incorrectly Classified Instances      1628               18.0889 %
Kappa statistic                          0.1888
Mean absolute error                      0.1809
Root mean squared error                  0.4253
Relative absolute error                 55.7356 %
Root relative squared error            105.587  %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.994    0.864    0.818      0.994    0.897      0.295    0.565     0.818     0
                 0.136    0.006    0.853      0.136    0.234      0.295    0.565     0.292     1
Weighted Avg.    0.819    0.689    0.825      0.819    0.762      0.295    0.565     0.711

=== Confusion Matrix ===

    a    b   <-- classified as
 7123   43 |    a = 0
 1585  249 |    b = 1
```

FIGURE 23 | SVM using RBF Kernel. Significant time to build and very high number of kernel evaluations for the same accuracy. (single 10-fold cross-validation, imbalanced training set)

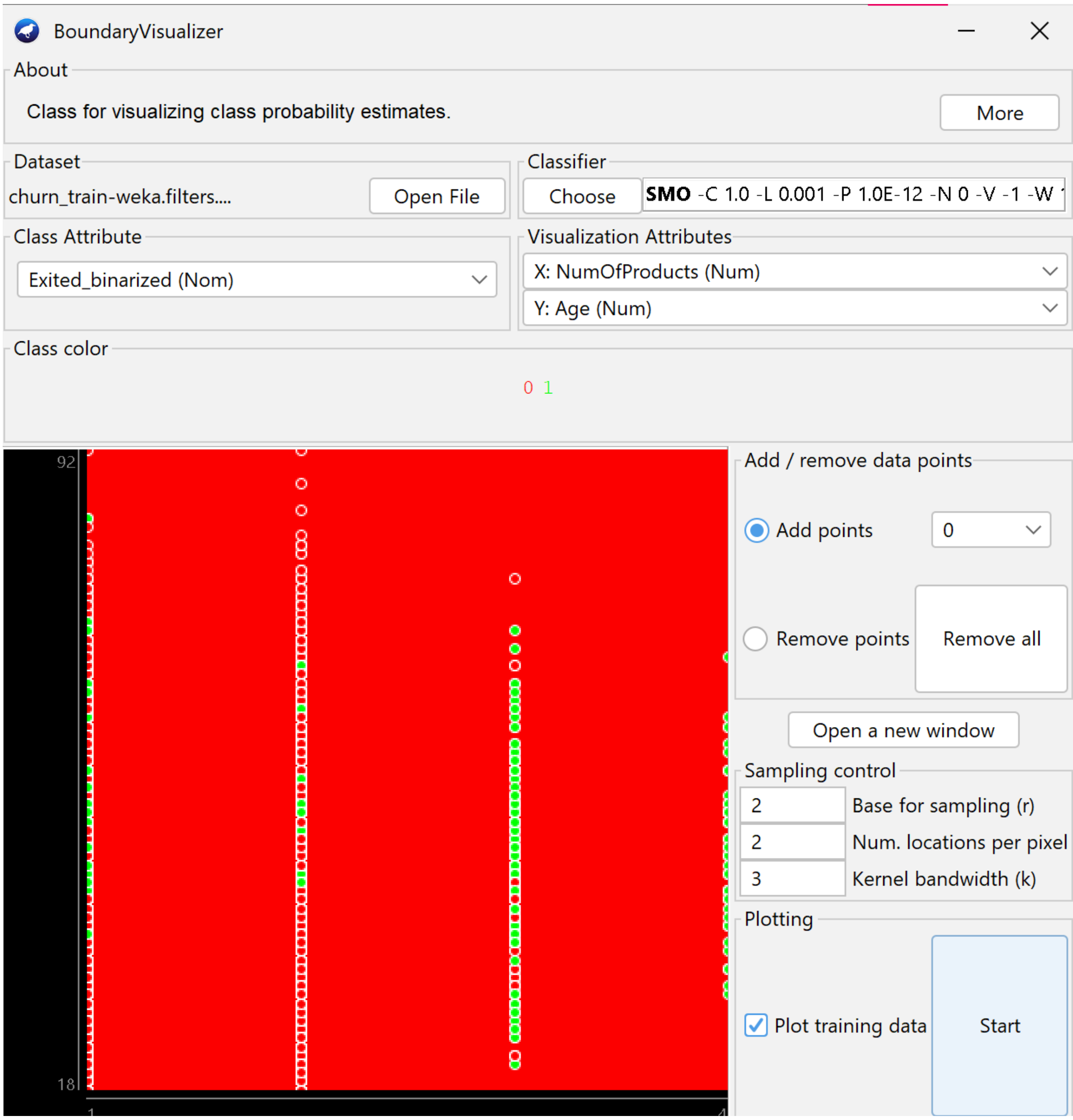

FIGURE 24 | SVM Boundary Visualiser between two highest information gain attributes, Age and NumOfProducts (numeric), does not display a boundary (imbalanced trainng set)

A.2.7 Random Forest (RandomForest)

```
Correctly Classified Instances        7654               85.0444 %
Incorrectly Classified Instances      1346               14.9556 %
Kappa statistic                          0.4595
Mean absolute error                      0.2173
Root mean squared error                  0.3367
Relative absolute error                 66.9529 %
Root relative squared error             83.5896 %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.957    0.565    0.869      0.957    0.911      0.480    0.827     0.938     0
                 0.435    0.043    0.720      0.435    0.542      0.480    0.827     0.644     1
Weighted Avg.    0.850    0.459    0.838      0.850    0.836      0.480    0.827     0.878

=== Confusion Matrix ===

    a    b   <-- classified as
 6856  310 |    a = 0
 1036  798 |    b = 1
```

FIGURE 25 | Random forest (RandomForest) (default model parameters, single 10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.
Analysing:  Percent_correct
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       26/10/2023, 22:39


Dataset                   (1) trees.Ra | (2) trees (3) trees (4) trees
-----------------------------------------------------------------------
churn_train-weka.filters.(100)   84.91 |   84.89     84.97     85.08
-----------------------------------------------------------------------
                               (v/ /*) |   (0/1/0)   (0/1/0)   (0/1/0)


Key:
(1) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(2) trees.RandomForest '-P 100 -I 50 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(3) trees.RandomForest '-P 100 -I 200 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(4) trees.RandomForest '-P 75 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
```

FIGURE 26 | Random forest—Accuracy—N and *k* hyperparameter comparison with (N, k): (default = 100, 100), (100, 50), (100, 200), (75, 100). (10x10-fold cross-validation, imbalanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.F
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       26/10/2023, 22:40


Dataset                   (1) trees.R | (2) tree (3) tree (4) tree
-------------------------------------------------------------------
churn_train-weka.filters.(100)   0.83 |   0.82 *   0.83 v   0.83
-------------------------------------------------------------------
                              (v/ /*) |  (0/0/1)  (1/0/0)  (0/1/0)


Key:
(1) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(2) trees.RandomForest '-P 100 -I 50 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(3) trees.RandomForest '-P 100 -I 200 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(4) trees.RandomForest '-P 75 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
```

FIGURE 27 | Random forest—AUC ROC—N and *k* hyperparameter comparison with (N, *k*): (default = 100, 100), (100, 50), (100, 200), (75, 100). (10x10-fold cross-validation, imbalanced training set)

A.3 Evaluation Outputs

A.3.1 Evaluation Matrix and Metrics

For classification (categorical) outputs, the following metrics and explanations come from the University of Essex Online (N.D.), and WEKA's Witten et al. (2017) and Bouckaert et al. (2022).

A.3.1.1 Kappa Statistic

- Kappa statistic compares model accuracy to random baseline.
- Higher is better as values near 0 means model is no better than random.

A.3.1.2 Confusion Matrix

TABLE 7 Confusion Matrix

		TRUE CLASS	
		Positive	**Negative**
PREDICTED CLASS	**Positive**	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

- Higher numbers on the diagonal (TP and TN), with lower FP and FN, visually indicates higher success of the algorithm.

A.3.1.3 Accuracy

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)}$$

- For WEKA Explorer: Correctly Classified Instances
- Higher accuracy is better
- Not useful for imbalanced datasets

A.3.1.4 Precision

$$Precision = \frac{TP}{TP + FP}$$

- Percentage of positive predictions that were correct
- Higher precision is better
- Correctly tagged divided by tagged

A.3.1.5 Recall (True Positive (TP) rate)

$$TP\ Rate = Recall = \frac{TP}{TP + FN}$$

- Percentage of correctly found positive cases
- Higher is better
- Correctly tagged divided by should be tagged
- In WEKA this is the same as TP rate

A.3.1.6 False Positive (FP) Rate

$$FP\ Rate = \frac{FP}{FP + TN}$$

- FP rate in WEKA is FP divided by total negatives (FP+TN)

- Percentage incorrectly predicted as positive
- Lower is better

A.3.1.7 F-Measure

$$F_\beta = (1 + \beta^2) \times \frac{P \times R}{\beta^2 P + R}$$

- Higher F-measure means better balance of precision and recall.
- Higher is better, maximum is 1.
- Model with higher precision, recall and F1 is better (especially for minority positive class in an imbalanced dataset.)
- $F_{\beta=1}$ measure equally favours both precision and recall
- β specifies if weight is applied more to precision or recall.

A.3.1.8 AUC-ROC

- Receiver Operating Characteristic Area Under the Curve
- AUC-ROC ranges from 0.0 to 1.0, higher is better
- Model performance should be above AUC-ROC
- AUC of 0.0 is 100% wrong; 1.0 is 100% correct; 0.5 is no better than chance
- A good model AUC above 0.8
- One of the best metrics for an imbalanced dataset as evaluates both positive and negative classes
- Measures between the True Positive (TP) Rate on the y-axis and the False Positive (FP) Rate on the x-axis

A.4.1 Imbalanced Training Evaluation

TABLE 8 | Hyperparameter selection (imbalanced training set)

Model	Hyperparameter	WEKA Name	Default	Tuned	Description
KNN	*k (k-nearest)*	KNN	1	17	17 was best AUC-ROC and accuracy
DT	Pruning level	confidenceFactor	0.25	0.1	0.1 had slightly better AUC-ROC and created simpler tree (smaller with fewer leaves)
SVM	Kernel	kernel	PolyKernel	PolyKernel	Compared to RBFKernel which was slower with same results.
RF	*k* (trees in RF)	numIterations	100	100	Default was best. Tried 50, 100, and 200.
	N (bag % of training set)	bagSizePercent	100	100	Default was best. Tried 100 and 75.

TABLE 9 | Imbalanced training evaluation using single 10-fold cross-validation with imbalanced training set (9000) and tuned hyperparameters (KNN k = 17, DT confidenceFactor = 0.1) (majority class 0=retained; minority class 1=churned)

Model	Build (Sec)	Accuracy	Class	TP Rate (Recall)	FP Rate	Precision	F-Measure	ROC Area
Baseline	0.01	79.62%	0	1.000	1.000	0.796	0.887	0.499
			1	0.000	0.000	?	?	0.499
KNN (17)	**0**	83.88%	0	0.976	0.697	0.845	0.906	0.808
			1	0.303	0.024	**0.763**	0.434	0.808
DT (0.1)	0.17	**85.41%**	0	**0.972**	0.605	0.863	**0.914**	0.806
			1	0.395	0.028	0.780	0.525	0.806
SVM	1.42	81.91%	0	0.994	0.864	0.818	0.897	0.565
			1	0.136	**0.006**	0.853	0.234	0.565
RF	2.31	85.04%	0	0.957	**0.565**	**0.869**	0.911	**0.827**
			1	**0.435**	0.043	0.720	**0.542**	**0.827**

A.4.1.1 K-Nearest Neighbor

```
IB1 instance-based classifier
using 17 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7549               83.8778 %
Incorrectly Classified Instances      1451               16.1222 %
Kappa statistic                          0.3596
Mean absolute error                      0.2331
Root mean squared error                  0.3478
Relative absolute error                 71.8356 %
Root relative squared error             86.343  %
Total Number of Instances             9000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.976    0.697    0.845      0.976    0.906      0.412    0.808     0.926     0
                 0.303    0.024    0.763      0.303    0.434      0.412    0.808     0.570     1
Weighted Avg.    0.839    0.560    0.829      0.839    0.810      0.412    0.808     0.854

=== Confusion Matrix ===

    a    b   <-- classified as
 6993  173 |    a = 0
 1278  556 |    b = 1
```

FIGURE 28 | KNN (k=17, single 10-fold cross-validation, imbalanced training set)

A.5.1 Oversampled (Balanced) Training Evaluation

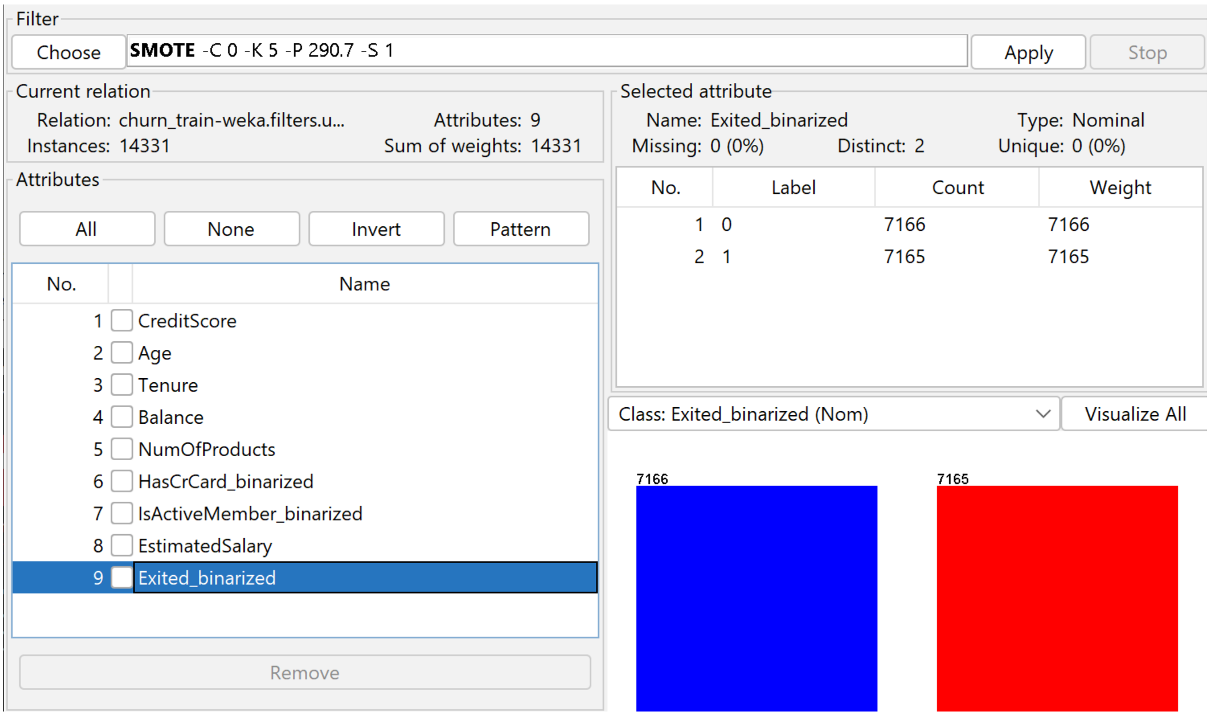

FIGURE 29 | After SMOTE, creates a balanced dataset. Randomised applied next.

TABLE 10 | Oversampled balanced training evaluation using single 10-fold cross-validation with balanced training set (14,331) and same hyperparameters as imbalanced (majority class 0=retained; minority class 1=churned)

Model	Build (Sec)	Accuracy	Class	TP Rate (Recall)	FP Rate	Precision	F-Measure	ROC Area
Baseline	0	50.00%	0	1.000	1.000	0.500	0.667	0.500
			1	0.000	0.000	?	?	0.500
KNN (17)	**0**	81.15%	0	0.769	0.146	0.840	0.803	0.894
			1	0.854	0.231	0.787	0.819	0.894
DT (0.1)	0.19	88.03%	0	**0.930**	0.169	0.846	**0.886**	0.925
			1	0.831	**0.070**	**0.922**	0.874	0.925
SVM	14.92	78.34%	0	0.790	0.224	0.780	0.785	0.783
			1	0.776	0.210	0.787	0.782	0.783
RF	3.4	**88.24%**	0	0.905	**0.140**	**0.866**	0.885	**0.948**
			1	**0.860**	0.095	0.900	**0.880**	**0.948**

TABLE 11 | New hyperparameter selections (oversampled balanced training set)

Model	Hyperparameter	WEKA Name	Default	Tuned	Description
KNN	*k (k-nearest)*	KNN	1	9	9 was best accuracy and just slightly lower AUC-ROC to 17 *(0.891 instead of 0.894) Precision, recall and F-measure were all better.
DT	Pruning level	confidenceFactor	0.25	0.05	0.05 had slightly better AUC-ROC but matched 0.1 on accuracy. It did create a simpler tree (smaller with fewer leaves)
SVM	Kernel	kernel	PolyKernel	PolyKernel	RBFKernel was significantly slower.
RF	*k* (trees in RF)	numIterations	100	100	Default was best for AUC-ROC and only .05 better for 200. For simplicity and speed keeping 100. Tried 50, 100, and 200.
	N (bag % of training set)	bagSizePercent	100	100	Default was best. Tried 100 and 75.

TABLE 12 | Oversampled balanced training evaluation using single 10-fold cross-validation with balanced training set (14,331) and new hyperparameters for balanced training set (majority class 0=retained; minority class 1=churned)

Model	Build (Sec)	Accuracy	Class	TP Rate (Recall)	FP Rate	Precision	F-Measure	ROC Area
Baseline	0	50.00%	0	1.000	1.000	0.500	0.667	0.500
			1	0.000	0.000	?	?	0.500
KNN (9)	**0**	81.53%	0	0.777	0.146	0.842	0.808	0.891
			1	0.854	0.231	0.787	0.819	0.891
DT (0.05)	0.19	88.03%	0	**0.926**	0.166	0.848	**0.886**	0.926
			1	0.834	**0.074**	**0.919**	0.875	0.926
SVM	14.92	78.34%	0	0.790	0.224	0.780	0.785	0.783
			1	0.776	0.210	0.787	0.782	0.783
RF	3.4	**88.24%**	0	0.905	**0.140**	**0.866**	0.885	**0.948**
			1	**0.860**	0.095	0.900	**0.880**	**0.948**

TABLE 13 | Comparison between imbalanced and oversampled balanced training sets with hypertuned parameters (KNN k = 17 or 9, DT confidenceFactor = 0.1 or 0.05)

Model	IMBALANCED Accuracy (KNN = 17, DT = 0.1)	OVERSAMPLED Accuracy (KNN = 17, DT = 0.1)	OVERSAMPLED Accuracy (KNN = 9, DT = 0.05)	IMBALANCED ROC Area (KNN = 17, DT = 0.1)	OVERSAMPLED ROC Area (KNN = 17, DT = 0.1)	OVERSAMPLED ROC Area (KNN = 9, DT = 0.05
Baseline	79.62%	50.00%	50.00%	0.499	0.500	50.00%
KNN	83.88%	81.15%	81.53%	0.808	0.894	0.891
DT	**85.41%**	88.03%	88.03%	0.806	0.925	0.926
SVM	81.91%	78.34%	78.34%	0.565	0.783	0.783
RF	85.04%	**88.24%**	**88.24%**	**0.827**	**0.948**	**0.948**

A.5.1.1 K-Nearest Neighbor

```
IB1 instance-based classifier
using 17 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       11629               81.1458 %
Incorrectly Classified Instances      2702               18.8542 %
Kappa statistic                          0.6229
Mean absolute error                      0.2554
Root mean squared error                  0.3627
Relative absolute error                 51.0826 %
Root relative squared error             72.5483 %
Total Number of Instances            14331

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.769    0.146    0.840      0.769    0.803      0.625    0.894     0.887     0
                 0.854    0.231    0.787      0.854    0.819      0.625    0.894     0.878     1
Weighted Avg.    0.811    0.189    0.814      0.811    0.811      0.625    0.894     0.883

=== Confusion Matrix ===

    a    b   <-- classified as
 5512 1654 |    a = 0
 1048 6117 |    b = 1
```

FIGURE 30 | KNN with hyperparameter k=17 (10-fold cross-validation, balanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPla
Analysing:  Percent_correct
Datasets:   1
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 01:54


Dataset                   (1) lazy.IBk | (2) lazy. (3) lazy. (4) lazy. (5) lazy. (6) lazy. (7) lazy.
---------------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)   79.35 |   81.33 v   81.48 v   81.24 v   81.15 v   80.99 v   80.75 v
---------------------------------------------------------------------------------------------------
                              (v/ /*) |   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)   (1/0/0)


Key:
(1) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"'
(2) lazy.IBk '-K 5 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"'
(3) lazy.IBk '-K 9 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"'
(4) lazy.IBk '-K 15 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\'
(5) lazy.IBk '-K 17 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\'
(6) lazy.IBk '-K 19 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\'
(7) lazy.IBk '-K 29 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\'
```

FIGURE 31 | KNN—Accuracy—hyperparameter comparison. 9 had highest accuracy (10x10-fold cross-validation, balanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPl
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 7
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 01:55


Dataset                   (1) lazy.IB | (2) lazy (3) lazy (4) lazy (5) lazy (6) lazy (7) lazy
------------------------------------------------------------------------------------------------
churn_train-weka.filters.(100)   0.79 |   0.88 v   0.89 v   0.89 v   0.89 v   0.89 v   0.89 v
------------------------------------------------------------------------------------------------
                              (v/ /*) |  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)  (1/0/0)


Key:
(1) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"
(2) lazy.IBk '-K 5 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"
(3) lazy.IBk '-K 9 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\"
(4) lazy.IBk '-K 15 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\
(5) lazy.IBk '-K 17 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\
(6) lazy.IBk '-K 19 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\
(7) lazy.IBk '-K 29 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R first-last\\\"\
```

FIGURE 32 | KNN—AUC-ROC—hyperparameter comparison. 9+ had highest AUC-ROC (10x10-fold cross-validation, balanced training set)

```
IB1 instance-based classifier
using 9 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       11684               81.5296 %
Incorrectly Classified Instances      2647               18.4704 %
Kappa statistic                          0.6306
Mean absolute error                      0.244
Root mean squared error                  0.3639
Relative absolute error                 48.7995 %
Root relative squared error             72.7822 %
Total Number of Instances            14331

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.777    0.146    0.842      0.777    0.808      0.632    0.891     0.878     0
                 0.854    0.223    0.793      0.854    0.822      0.632    0.891     0.866     1
Weighted Avg.    0.815    0.185    0.817      0.815    0.815      0.632    0.891     0.872

=== Confusion Matrix ===

    a    b   <-- classified as
 5567 1599 |    a = 0
 1048 6117 |    b = 1
```

FIGURE 33 | KNN with hyperparameter k=9 (10-fold cross-validation, balanced training set)

A.5.1.2 Decision Tree

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experi
Analysing:  Percent_correct
Datasets:   1
Resultsets: 5
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 02:57


Dataset                   (1) trees.J4 | (2) trees (3) trees (4) trees (5) trees
--------------------------------------------------------------------------------
churn_train-weka.filters.(100)   87.63 |   88.04 v   88.04 v   87.99     87.94
--------------------------------------------------------------------------------
                                (v/ /*) |   (1/0/0)   (1/0/0)   (0/1/0)   (0/1/0)


Key:
(1) trees.J48 '-C 0.25 -M 2' -217733168393644444
(2) trees.J48 '-C 0.1 -M 2' -217733168393644444
(3) trees.J48 '-C 0.05 -M 2' -217733168393644444
(4) trees.J48 '-C 0.025 -M 2' -217733168393644444
(5) trees.J48 '-C 0.02 -M 2' -217733168393644444
```

FIGURE 34 | Decision tree —Accuracy—hyperparameter comparison. 0.1 and 0.05 had highest accuracy (10x10-fold cross-validation, balanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.exper
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 5
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 02:59


Dataset                   (1) trees.J | (2) tree (3) tree (4) tree (5) tree
----------------------------------------------------------------------------
churn_train-weka.filters.(100)   0.92 |   0.92     0.93     0.92     0.92
----------------------------------------------------------------------------
                               (v/ /*) |  (0/1/0)  (0/1/0)  (0/1/0)  (0/1/0)


Key:
(1) trees.J48 '-C 0.25 -M 2' -217733168393644444
(2) trees.J48 '-C 0.1 -M 2' -217733168393644444
(3) trees.J48 '-C 0.05 -M 2' -217733168393644444
(4) trees.J48 '-C 0.025 -M 2' -217733168393644444
(5) trees.J48 '-C 0.02 -M 2' -217733168393644444
```

FIGURE 35 | Decision tree —AUC-ROC—hyperparameter comparison. 0.05 had highest AUC-ROC (10x10-fold cross-validation, balanced training set)

```
Number of Leaves  :     76

Size of the tree :      149


Time taken to build model: 0.24 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       12616               88.0329 %
Incorrectly Classified Instances      1715               11.9671 %
Kappa statistic                          0.7607
Mean absolute error                      0.1826
Root mean squared error                  0.3094
Relative absolute error                 36.5102 %
Root relative squared error             61.887  %
Total Number of Instances            14331

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.926    0.166    0.848      0.926    0.886      0.764    0.926     0.892     0
                 0.834    0.074    0.919      0.834    0.875      0.764    0.926     0.935     1
Weighted Avg.    0.880    0.120    0.884      0.880    0.880      0.764    0.926     0.914

=== Confusion Matrix ===

    a    b   <-- classified as
 6638  528 |    a = 0
 1187 5978 |    b = 1
```

FIGURE 36 | Decision tree with 149 nodes and 76 leaves (confidenceFactor=0.05, single 10-fold cross-validation, oversampled balanced training set)

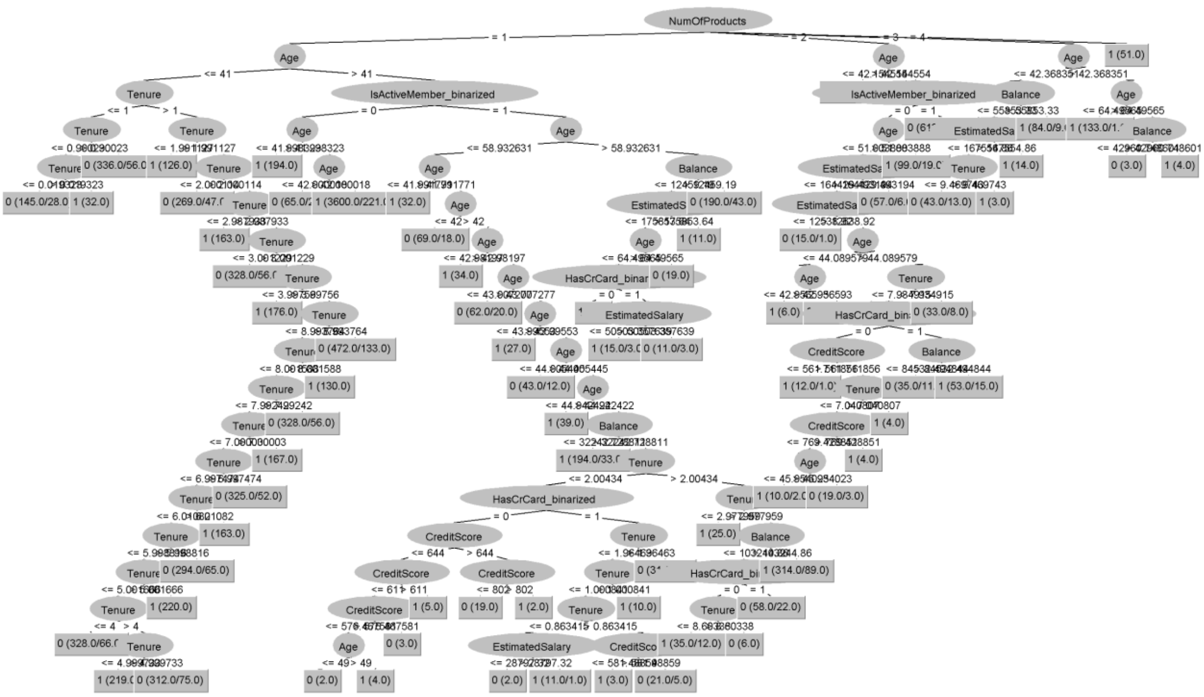

FIGURE 37 | Decision tree showing NumOfProducts and then Age as most important attributes. (confidenceFactor=0.05, single 10-fold cross-validation, oversampled balanced training set)

A.5.1.4 Random Forest

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPla
Analysing:  Percent_correct
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 04:03


Dataset                  (1) trees.Ra | (2) trees (3) trees (4) trees
----------------------------------------------------------------------
churn_train-weka.filters.(100)   88.32 |   88.19     88.37     88.24
----------------------------------------------------------------------
                              (v/ /*) |   (0/1/0)   (0/1/0)   (0/1/0)


Key:
(1) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(2) trees.RandomForest '-P 100 -I 50 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(3) trees.RandomForest '-P 100 -I 200 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(4) trees.RandomForest '-P 75 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
```

FIGURE 38 | Random forest—Accuracy—N and *k* hyperparameter comparison with (N, k): (default = 100, 100), (100, 50), (100, 200), (75, 100). (10x10-fold cross-validation, oversampled balanced training set)

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4,5,6 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPl
Analysing:  Area_under_ROC
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       28/10/2023, 04:04


Dataset                  (1) trees.R | (2) tree (3) tree (4) tree
-------------------------------------------------------------------
churn_train-weka.filters.(100)   0.95 |   0.95 *   0.95 v   0.95
-------------------------------------------------------------------
                             (v/ /*) |  (0/0/1)  (1/0/0)  (0/1/0)


Key:
(1) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(2) trees.RandomForest '-P 100 -I 50 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(3) trees.RandomForest '-P 100 -I 200 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(4) trees.RandomForest '-P 75 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
```

FIGURE 39 | Random forest—AUC-ROC—N and *k* hyperparameter comparison with (N, k): (default = 100, 100), (100, 50), (100, 200), (75, 100). (10x10-fold cross-validation, oversampled balanced training set)

A.6.1 Test Set Results

A.6.1.1 TEST SET RESULTS

TABLE 14 | TEST SET RESULTS COMPARISON SUMMARY

Model	Balance	Accuracy	Class	TP Rate (Recall)	FP Rate	Precision	F-Measure	ROC Area
KNN (17)	Imbalanced	84.1%	0	0.971	0.670	0.851	0.907	0.809
			1	0.330	0.029	0.744	0.457	0.809
DT (0.1)	Imbalanced	**85.8%**	0	**0.980**	0.621	0.861	**0.917**	0.802
			1	0.379	0.020	0.828	0.520	0.802
SVM	Imbalanced	82.5%	0	0.996	0.847	0.822	0.901	0.574
			1	0.153	**0.004**	**0.912**	0.262	0.574
RF	Imbalanced	84.9%	0	0.950	0.547	0.872	0.909	0.828
			1	**0.453**	0.050	0.697	0.549	**0.828**
KNN (9)	Oversampled	74.2%	0	0.758	**0.320**	**0.903**	0.824	0.791
			1	0.680	0.242	0.417	0.517	0.791
DT (0.05)	Oversampled	85%	0	0.934	0.478	0.885	0.908	0.803
			1	0.522	0.066	0.667	**0.586**	0.803
SVM	Oversampled	74.7%	0	0.789	0.419	0.881	0.833	0.685
			1	0.581	0.211	0.581	0.483	0.685
RF	Oversampled	83.7%	0	0.907	0.438	0.890	0.899	**0.843**
			1	**0.562**	0.093	0.606	0.583	**0.843**

A.6.1.2 Imbalanced

A.6.1.2.1 K-Nearest Neighbour: TEST RESULT (Imbalanced)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervi
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         841               84.1    %
Incorrectly Classified Instances       159               15.9    %
Kappa statistic                          0.38
Mean absolute error                      0.2339
Root mean squared error                  0.3462
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.971    0.670    0.851      0.971    0.907      0.423    0.809     0.926     0
                 0.330    0.029    0.744      0.330    0.457      0.423    0.809     0.581     1
Weighted Avg.    0.841    0.540    0.829      0.841    0.816      0.423    0.809     0.856

=== Confusion Matrix ===

   a   b   <-- classified as
 774  23 |   a = 0
 136  67 |   b = 1
```

FIGURE 40 | k-Nearest Neighbor: TEST RESULT (Imbalanced, k=17)

A.6.1.2.1 Decision Tree: TEST RESULT (Imbalanced)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervis
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         858               85.8    %
Incorrectly Classified Instances       142               14.2    %
Kappa statistic                          0.4501
Mean absolute error                      0.2244
Root mean squared error                  0.3376
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.980    0.621    0.861      0.980    0.917      0.498    0.802     0.917     0
                 0.379    0.020    0.828      0.379    0.520      0.498    0.802     0.574     1
Weighted Avg.    0.858    0.499    0.854      0.858    0.836      0.498    0.802     0.848

=== Confusion Matrix ===

   a   b   <-- classified as
 781  16 |   a = 0
 126  77 |   b = 1
```

FIGURE 41 | Decision tree: TEST RESULT (Imbalanced, confidenceFactor = 0.1)

A.6.1.2.1 Support Vector Machine: TEST RESULT (Imbalanced)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervis
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         825               82.5    %
Incorrectly Classified Instances       175               17.5    %
Kappa statistic                          0.2159
Mean absolute error                      0.175
Root mean squared error                  0.4183
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.996    0.847    0.822      0.996    0.901      0.331    0.574     0.822     0
                 0.153    0.004    0.912      0.153    0.262      0.331    0.574     0.311     1
Weighted Avg.    0.825    0.676    0.840      0.825    0.771      0.331    0.574     0.718

=== Confusion Matrix ===

   a   b   <-- classified as
 794   3 |   a = 0
 172  31 |   b = 1
```

FIGURE 42 | Support Vector Machine: TEST RESULT (Imbalanced)

A.6.1.2.1 Random Forest: TEST RESULT (Imbalanced)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsuperv
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         849               84.9    %
Incorrectly Classified Instances       151               15.1    %
Kappa statistic                          0.4634
Mean absolute error                      0.2185
Root mean squared error                  0.3361
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.950    0.547    0.872      0.950    0.909      0.479    0.828     0.938     0
                 0.453    0.050    0.697      0.453    0.549      0.479    0.828     0.655     1
Weighted Avg.    0.849    0.446    0.837      0.849    0.836      0.479    0.828     0.880

=== Confusion Matrix ===

   a   b   <-- classified as
 757  40 |   a = 0
 111  92 |   b = 1
```

FIGURE 43 | Random forest: TEST RESULT (Imbalanced)

A.6.1.3 Oversampled (Balanced)

A.6.1.3.1 K-Nearest Neighbor: TEST RESULT (Oversampled)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervised.
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         742               74.2    %
Incorrectly Classified Instances       258               25.8    %
Kappa statistic                          0.3544
Mean absolute error                      0.2894
Root mean squared error                  0.4199
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.758    0.320    0.903      0.758    0.824      0.374    0.791     0.917     0
                 0.680    0.242    0.417      0.680    0.517      0.374    0.791     0.516     1
Weighted Avg.    0.742    0.304    0.804      0.742    0.762      0.374    0.791     0.835

=== Confusion Matrix ===

   a   b   <-- classified as
 604 193 |   a = 0
  65 138 |   b = 1
```

FIGURE 44 | k-Nearest Neighbor: TEST RESULT (Oversampled, k=9)

A.6.1.3.2 Decision Tree: TEST RESULT (Oversampled)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsuperv
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         850               85      %
Incorrectly Classified Instances       150               15      %
Kappa statistic                          0.4957
Mean absolute error                      0.2386
Root mean squared error                  0.354
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.934    0.478    0.885      0.934    0.908      0.501    0.803     0.924     0
                 0.522    0.066    0.667      0.522    0.586      0.501    0.803     0.567     1
Weighted Avg.    0.850    0.394    0.840      0.850    0.843      0.501    0.803     0.852

=== Confusion Matrix ===

   a   b   <-- classified as
 744  53 |   a = 0
  97 106 |   b = 1
```

FIGURE 45 | Decision tree: TEST RESULT (Oversampled, confidenceFactor = 0.05)

A.6.1.3.3 Support Vector Machine: TEST RESULT (Oversampled)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervi
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         747               74.7    %
Incorrectly Classified Instances       253               25.3    %
Kappa statistic                          0.3215
Mean absolute error                      0.253
Root mean squared error                  0.503
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.789    0.419    0.881      0.789    0.833      0.330    0.685     0.863     0
                 0.581    0.211    0.413      0.581    0.483      0.330    0.685     0.325     1
Weighted Avg.    0.747    0.377    0.786      0.747    0.762      0.330    0.685     0.754

=== Confusion Matrix ===

   a   b   <-- classified as
 629 168 |   a = 0
  85 118 |   b = 1
```

FIGURE 46 | Support Vector Machine: TEST RESULT (Oversampled)

A.6.1.3.4 Random Forest: TEST RESULT (Oversampled)

```
=== Re-evaluation on test set ===

User supplied test set
Relation:     churn_test-weka.filters.unsupervised.attribute.NumericToBinary-R9-weka.filters.unsupervi…
Instances:     unknown (yet). Reading incrementally
Attributes:   9

=== Summary ===

Correctly Classified Instances         837               83.7    %
Incorrectly Classified Instances       163               16.3    %
Kappa statistic                          0.482
Mean absolute error                      0.2341
Root mean squared error                  0.3444
Total Number of Instances             1000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.907    0.438    0.890      0.907    0.899      0.483    0.843     0.946     0
                 0.562    0.093    0.606      0.562    0.583      0.483    0.843     0.660     1
Weighted Avg.    0.837    0.368    0.833      0.837    0.835      0.483    0.843     0.888

=== Confusion Matrix ===

   a   b   <-- classified as
 723  74 |   a = 0
  89 114 |   b = 1
```

FIGURE 47 | Random forest: TEST RESULT (Oversampled)

A.7.1 Comparison

A.7.1.1 TRAINING COMPARISON SUMMARY

TABLE 15 | TRAINING COMPARISON SUMMARY

Model	Balance	Accuracy	Class	TP Rate (Recall)	FP Rate	Precision	F-Measure	ROC Area
KNN (17)	Imbalanced	83.88%	0	0.976	0.697	0.845	0.906	0.808
			1	0.303	0.024	**0.763**	0.434	0.808
DT (0.1)	Imbalanced	**85.41%**	0	**0.972**	0.605	0.863	**0.914**	0.806
			1	0.395	0.028	0.780	0.525	0.806
SVM	Imbalanced	81.91%	0	0.994	0.864	0.818	0.897	0.565
			1	0.136	**0.006**	0.853	0.234	0.565
RF	Imbalanced	85.04%	0	0.957	**0.565**	**0.869**	0.911	**0.827**
			1	**0.435**	0.043	0.720	**0.542**	**0.827**
KNN (9)	Oversampled	81.53%	0	0.777	0.146	0.842	0.808	0.891
			1	0.854	0.231	0.787	0.819	0.891
DT (0.05)	Oversampled	88.03%	0	**0.926**	0.166	0.848	**0.886**	0.926
			1	0.834	**0.074**	**0.919**	0.875	0.926
SVM	Oversampled	78.34%	0	0.790	0.224	0.780	0.785	0.783
			1	0.776	0.210	0.787	0.782	0.783
RF	Oversampled	**88.24%**	0	0.905	**0.140**	**0.866**	0.885	**0.948**
			1	**0.860**	0.095	0.900	**0.880**	**0.948**

A.7.1.2 Other Research

Table 7 Performance of the optimized models on the test set. Test data was heldout during the entire process (10% of the dataset)

	True Positive	True Negative	False Positive	False Negative	Accuracy	Precision	F-measure
Decision tree	38.8	39.4	10.6	11.2	78.2	78.5	78.05
Knn	37.8	40.1	9.9	12.2	77.9	79.2	77.36
Elastic net	40.5	35.7	14.3	9.5	76.2	73.9	77.29
Logistic regression	40.4	35.8	14.2	9.6	76.2	74.0	77.26
Svm	39.6	40.7	9.3	10.4	80.3	81.0	80.09
Random forests	40.1	42.6	7.4	9.9	82.8	84.4	82.25

We used training data (90% of the dataset) for model selection: we apply a repeated *k*-fold cross validation (ten independent times) with $k = 10$ to optimize the hyperparameters of each model. We used ROC as the optimizing metric in the training process. After selecting the best hyperparameters, we retrain each model with the entire training set (because we hold out one fold each time in a cross-validation procedure) with these optimized values.

FIGURE 48 | de Lima Lemos et.al with balanced dataset using AUC-ROC for training (de Lima Lemos et al., 2022)

TABLE IV
RESULTS BY APPLYING CLASSIFIERS DIRECTLY

Classifier	Accuracy(%)	Accuracy After oversampling(%)
KNN	81.65	81.37
SVM	79.63	70.36
DT	78.99	91.98
RF	85.18	95.74

TABLE V
RESULTS AFTER MRMR FEATURE SELECTION

Classifier	Accuracy(%)	Accuracy After oversampling(%)
KNN	83.97	82.57
SVM	79.63	69.96
DT	78.32	91.73
RF	83.66	92.95

TABLE VI
RESULTS AFTER RELIEFF FEATURE SELECTION

Classifier	Accuracy(%)	Accuracy After oversampling(%)
KNN	82.15	80.99
SVM	79.63	69.53
DT	77.61	90.74
RF	81.75	92.19

FIGURE 49 | Rahman & Kumar with imbalanced dataset—accuracy before and after oversampling

(Rahman and Kumar, 2020)