Prediction of Customer Churn Risk with Advanced Machine Learning Methods

Oguzhan Akan

Department of Computer Science,
California State University,
Northridge, CA, USA
E-mail: oguzhan.akan.95@my.csun.edu

Abhishek Verma*

Department of Computer Science,
California State University,
Northridge, CA, USA
E-mail: abhishek.verma@csun.edu
* corresponding author

Sonika Sharma

Department of Commerce,
Shaheed Bhagat Singh College,
University of Delhi, Delhi, India
E-mail: sonika.sharma@sbs.du.ac.in

Abstract: Customer churn risk prediction is an important area of research as it directly impacts the revenue stream of businesses. An ability to predict customer churn allows businesses to come up with better strategies to retain existing customers. In this research we perform a comprehensive comparison of feature selection methods, upsampling methods, and machine learning methods on the customer churn risk dataset. (i) Our research compares likelihood-based, tree-based, and layer-based machine learning methods on the churn dataset. (ii) Models built on the churn dataset without upsampling performed better than oversampling methods. However, SMOTE and ADASYN helped stabilize model performance. (iii) The models built on ADASYN dataset were slightly better than the SMOTE counterparts. (iv) It was observed that XGBoost and Deep Cascading Forest combined with XGBoost were consistently better across all metrics compared to other methods. (v) Information Value analysis performed better than PCA. In particular, IVR DCFX model has the best AUROC score with 89.1%.

Keywords: Customer Churn; Deep Neural Networks; Deep Cascading Forest; Smote; Adasyn.

Reference to this paper should be made as follows: Akan, O., Verma, A., and Sharma, S. (202X) 'Prediction of Customer Churn Risk with Advanced Machine Learning Methods,' *International Journal of Data Science*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Oguzhan Akan received his Master of Science in Computer Science degree from California State University, Northridge. His research interests are in machine learning, deep learning, and big data analytics.

Abhishek Verma received his Ph.D. in Computer Science from New Jersey Institute of Technology, NJ, USA. His research interests are in data science, big data analytics, machine learning, and deep learning on big datasets.

Sonika Sharma received her Ph. D. in finance from University of Delhi, Delhi, India. She is Associate Professor, Department of Commerce, Shaheed Bhagat Singh College, University of Delhi, Delhi, India. Her research interests are in finance, insurance, and business analytics.

This journal is a revised and expanded version of a conference entitled “Machine Learning Models for Customer Churn Risk Prediction," *the 13th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, Oct. 26-29, 2022, Virtual.

1 Introduction

A major focus of businesses is to reduce the risk of customer churn and its negative impact on revenue. For this reason, the data regarding risk of churn is reported directly to the corporation’s upper-level management. Customer churn can be understood as a customer discontinues buying a company’s products or services. Companies must proactively look out for customers that have higher churn risk and retain them by offering discounts or promotions. Hence, churn prediction and customer retention has an enormous business value on the profitability of a business and thereby has an extremely important real-world application.

The motivation of our research is the challenging nature of churn prediction due to multiple factors that could impact the possibility of customer churn. Some of those factors are product pricing, competition, changes in customer’s preference, product quality, customer service, ease of buying, product availability, impact of inflation and other economic conditions, etc. A business must continually assess the risk of customer churn due to aforementioned situations and based on parameters related to the customer such as sales volume, gender, age, changes to buying habit such as online vs in a store, income, location, how frequently customer logs in the company’s website, complaint resolution, etc. Once the customer churn risk is identified a business should create a plan to address the risk early enough to maintain its customer base.

Previous systems for predicting churn risk were rule based expert systems, such systems lacked the predictive ability in comparison to more modern machine learning based systems. Machine learning methods such as logistic regression and random forests have been used in the past to measure the risk of churn. In recent years several new techniques have been created for data sampling, feature selection, and supervised machine learning. Recent machine learning methods such as deep cascading forest and deep neural networks have been successful in various domains on video, image, speech, and numerical data. These state-of-the-art techniques have not been previously applied to churn prediction. Our research applies those techniques and compares the performance with several other traditional machine learning techniques.

The objective of our study is to establish a comprehensive modeling pipeline and demonstrate how the steps taken for coming up with predictions affect the performance, this includes exploring feature selection methods to decrease the dataset's dimensions. Dimensionality reduction serves multiple purposes, such as selecting discriminative features for classification, increasing the time or space efficiency for running machine learning models or to mitigate potential overfitting during modeling. Our research employs Information Value (IV) technique and Principal Component Analysis (PCA) for dimensionality reduction.

We balance the churn dataset synthetically so that the machine learning models could precisely fit the feature space. Our study delves into two key inquiries: (i) assessing the superiority of upsampling compared to no-upsampling, and (ii) determining which technique, between Synthetic Minority Oversampling Technique (SMOTE) Chawla et al. (2002) and Adaptive Synthetic Sampling (ADASYN) He et al. (2008), yields better results in terms of the metric *area under the receiver operating characteristics curve (AUROC)* score. Additionally, our research places significant emphasis on comparing the performance of conventional machine learning approaches with emerging methods like deep cascading forest and deep neural networks using churn dataset. We describe machine learning methods in Section 5. Research includes studying the effects of incorporating XGBoost method Chen and Guestrin (2016) into the prediction layers of Deep Cascading Forests Zhou and Feng (2017) and Deep Neural Networks.

This paper is structured as follows. Section 2 discusses the related work. Next, we describe the churn dataset in Section 3. Sections 4 and 5 describe the proposed methodology. Next, the experiment setup and results are discussed in Sections 6 and 7. Papes draws conclusion and discusses future directions in Section 8 .

2 Related Work

While numerous studies have investigated churn risk prediction, only a limited number have formally evaluated the efficacy of Deep Neural Networks (DNN) in this domain. Mena et al. (2019) applied Long Short Term Memory (LSTM) to predict churn on sequential data. They conclude that LSTM gives better results compared to the model built with Logistic Regression by 25% on a set of static features. Recursive Neural Networks (RNNs) could also be applied due to the sequential nature of data.

In their research, Vafeiadis et al. (2015) compare multiple machine learning models to forecast customer churn risk. Their study encompasses models such as Naïve Bayes, Support Vector Machines, Decision Trees, Logistic Regression, Adaptive Boosting, and Neural Networks. The authors determine that SVM with a polynomial kernel, enhanced by AdaBoost, achieves the highest performance in terms of F-1 score on their dataset. Due to the rapid development in the area of machine learning, better methods have emerged since the aforementioned study such as XGBoost and DNN.

A recent study by Ahmad et al. (2019) focused on predicting churn risk using machine learning techniques. They utilized a dataset from a telecommunication company to develop churn prediction models employing XGBoost, Decision Trees, Gradient Boosting, and Random Forest algorithms. Their investigation reveals that XGBoost demonstrates a 2.5% improvement in AUROC score compared to the Gradient Boosting method. Their research only investigates tree-based methods. We in our research investigate both tree-based and other categories of machine learning methods.

In their study, Lalwani et al. (2022) examine various machine learning techniques for predicting churn risk using data from a telecommunications company. Their analysis encompasses machine learning models derived from statistical methods like Naïve Bayes, likelihood-based approaches such as Logistic Regression, tree-based methodologies, and ensemble methods including Random Forest, XGBoost, and Adaboost and SVM. They determine that XGBoost and Adaboost yield the most effective performance, with no notable difference between them.

Rahman et al. (2020) perform customer churn prediction using KNN, SVM, Decision Tree, and Random Forest. The dataset used in their study is from the banking sector. The authors conclude that Random Forest gave best accuracy when combined with oversampling. Sisodia et al. (2017) build machine learning models to predict employee churn on human resource analytics dataset. They construct a heatmap and correlation matrix to illustrate the relationships between attributes. For prediction purposes, they employ various machine learning models, including Linear Support Vector Machine, C 5.0 Decision Tree classifier, Random Forest, k-nearest neighbor, and Naïve Bayes classifier. Bhatnagar et al. (2019) implement two classification models based on supervised machine learning. They find that KNN outperforms Logistic Regression in predicting customer churn, achieving a 2.0% higher accuracy rate.

Our research aims to provide a comparison of most up to date machine learning methods, which includes XGBoost and Deep Cascading Forest. Several preprocessing techniques are applied on the churn dataset. It also includes the comparison of traditional machine learning methods to more recent Deep Neural Networks, such a comparison fills the gap from previous research.

3 Description of Churn Risk Dataset and Research Hypothesis

For this research, HackerEarth's customer churn dataset from a business offering products and services online was selected HackerEarth’s Churn Dataset (2021). This dataset contains various demographic and aggregated transaction details for training purposes, along with a target feature indicating whether the customer is at risk of churn. The dataset comprises 35,829 data points with 19 training features. Notably, the dataset exhibits class imbalance, with only 5,393 true labels assigned for the churn class, accounting for 17.72% of the data. This dataset characteristic influenced our decision-making process regarding reshaping training data and selecting appropriate model evaluation metrics. Preprocessing steps were necessary, involving cleaning missing or inaccurately populated data, encoding categorical features for usability, and upsampling training data to address the imbalance issue. Detailed descriptions of the preprocessing steps are provided later in this paper.

The research hypothesis in this paper is to evaluate various machine learning methdologies along with data preprocessing and feature selection strategies to be able to correctly classify if a customer would churn or not based on the training data from the preprocessed churn dataset. The model performance is compared using several different metrics for an in depth analysis of churn prediction detailed in the experiment results section of this paper. Another goal of this research is to compare the runtime of the various machine learning models. In addition, we compare the impact of various preprocessing and feature extraction techniques on the perfomance of the machine learning models.

4 Proposed Feature Selection and Data Preprocessing Methodology

4.1 Handling Missing Data or Inaccurate Data

Initially, values for $avg\_frequency\_login\_days$ below zero are substituted with zero. This decision stems from the belief that these values are erroneous, and zero is selected as the nearest non-negative integer. Data points corresponding to $days\_since\_last\_login$ with values below zero are excluded. This action is justified by the data description indicating that these customers have already churned. Incorporating such data points could introduce noise into the model.

Additionally, 9.48% of the values for the $avg\_frequency\_login\_days$ feature were found to be missing. Downey and King (1998) propose in their research that filling missing values with the mean of their respective categories is a suitable method for features with less than 20% missing values. Upon implementing this strategy on the churn dataset, the dataset size is reduced to 30,977 data points.

4.2 Categorical Feature Encoding

In our research, we utilized three commonly known encoding methods to convert categorical features into numerical features: (1) Ordinal Encoding, (2) Binary Encoding, and (3) One-Hot Encoding. Notably, both Binary Encoding and One-Hot Encoding methods are optimized within Python's Category Encoder module. This optimization automatically excludes newly generated features that exhibit no variation McGinnis et al. (2018). This feature is extremely helpful to reduce computer memory requirement and it enabled us to fit our models in the RAM.

4.2.1 Ordinal Encoding (OE)

Ordinal Encoding involves transforming non-numerical data into numerical data by assigning an integer value to each category. However, a drawback of this method is its assumption of order and its inability to accommodate newly introduced categories in the data Potdar et al. (2017). In the churn dataset, Ordinal Encoding resulted in the creation of 12 new features.

4.2.2 Binary Encoding (BE)

Binary Encoding is an alternative encoding method that represents categories in binary format. Initially, categories are converted to integers, starting from zero, which are then transformed into their binary equivalents. Binary Encoding results in the creation of $log_2(d)$ new columns for each categorical feature with d unique values Seger (2018). In the customer churn dataset, Binary Encoding led to the generation of 41 new features.

4.2.3 One-Hot Encoding (OH)

One-Hot Encoding is a method that shares some similarities with Binary Encoding. However, the main distinction lies in One-Hot Encoding's creation of a single additional feature for each category, rather than converting features into binary representations. Consequently, a significant drawback of One-Hot Encoding is its production of features equal in number to the unique values of a categorical variable Seger (2018). Despite this drawback, One-Hot Encoding features is included in the dataset due to its potential for providing additional informative value compared to other encoding methods. In our dataset, One-Hot Encoding resulted in the creation of 38 new features. When considering the features generated from Ordinal Encoding, Binary Encoding, and One-Hot Encoding together, the total number of features amounts to 91.

4.3 Training and Test Data Partitions

Stratified data splitting is a technique that maintains the balance of true class label ratios in both training and test partitions while preserving randomness Vilarino et al. (2005). The stratified partitioning method provided by Scikit-learn Pedregosa et al. (2011) was employed, ensuring that the data remained closely distributed in both the training and test partitions, 15.08% in training and 15.07% in test. Additionally, 20% of the data was allocated for the test partition. Following the partitioning, the training and test partitions comprised 24,781 and 6,196 data points respectively.

4.4 Feature Selection Methodology

After partitioning the cleaned data into training and test sets, we apply two distinct feature selection methods: (1) Information Value (IV) and (2) Principal Component Analysis (PCA). Subsequently, we construct several machine learning models using the features selected by these two approaches and compare their performance.

4.4.1 Information Value (IV)

Information Value (IV) serves as a tool for exploratory data analysis, allowing assessment of a given factor on the target feature. It is applicable to both categorical and continuous features Verma. (2020). Weight of Evidence (WOE) is employed to compute IV, which represents the natural logarithm of the ratio of distributions of non-events to events. In our study, events correspond to churned customers, represented as 1, while non-events are represented by retained customers as 0.

An IV less than 0.02 is deemed non-informative and should be excluded during modeling. Conversely, an IV exceeding 0.5 may lead to overfitting. Out of the 91 categorical features generated with category encoders, only 33 features exhibited an IV score greater than 0.02.

Categorical Features. IV was applied using encoded features. For each categorical feature, a bubble graph displaying the population size and target distribution per category was generated. For instance, Figure 1 illustrates the differentiation of churned customers per category for the feature named $BE\_membership\_category\_1$.

Continuous Features. The IV calculation for continuous features involves an additional step known as *binning*. This process can be conceptualized as converting continuous features into categorical features. However, in the modeling stage, the raw forms of continuous features are utilized. The binning process involved several steps:

1. The number of bins was set to 10 to ensure an adequate number of data points in each bin.

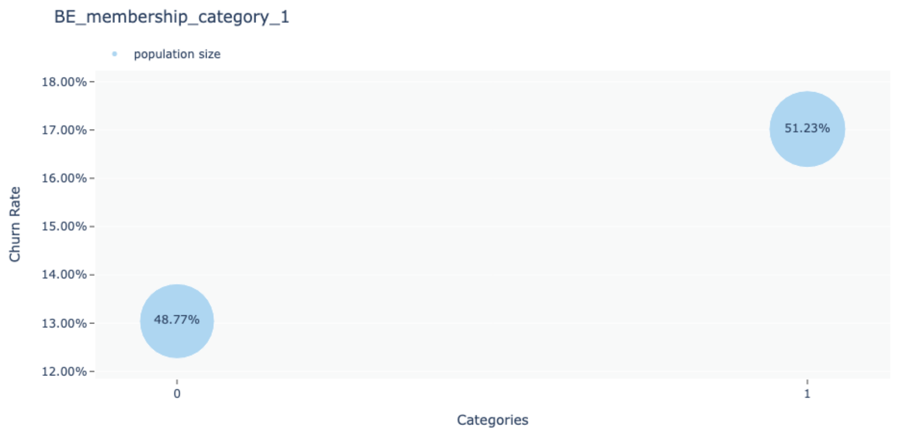

Figure 1 Information Value (IV) analysis graph for the binary-encoded categorical feature membership_category. The category "0" constitutes 49% of the dataset with 13% churn rate, while the category "1" constitutes the remaining 51% of the dataset with 17% churn rate. The IV for this feature is 0.024.

2. The maximum value within a feature was capped at the 95th percentile, and the minimum value was capped at the 5th percentile.
3. The feature values were sorted in ascending order, and a bin number was assigned to each data point.
4. Each bin number was treated as a category, and IV calculation was applied.

This process enabled the calculation of Information Value for continuous features by treating them as categorical features after binning.

Figure 2 shows a visualization of each bin and the corresponding churn ratio for the feature $avg\_frequency\_login\_days$. This graph is useful for understanding how binning works and how the feature is correlated to the target. A linear decreasing trend of the churn rate in the graph means there is a negative correlation while an increasing trend means a positive correlation. Sinusoidal or polynomial-like trends are considered weak correlation. Out of the seven continuous features, five features had an IV value higher than 0.02. Total number of features are 38.

Discarding Highly Correlated Features. As the final step of feature selection using IV, the features are initially ranked based on their IV values in descending order. Subsequently, for each feature pair (f_i, f_j), the correlation between the pairs is computed. We discard those features where the correlation exceeds 95%. Figures 3 and 4 illustrate the correlation matrix constructed before and after removing highly correlated features, respectively. Following the elimination of highly correlated features, the final training set for IV comprises a total of 31 features of those 26 are categorical and 5 are continuous features.

The overall data preprocessing steps on the churn dataset are visualized in Figure 5. As can be seen in the figure, the raw data is first cleaned by handling incorrect value and missing values. This step includes discarding some data points and filling missing values with category mean or with the best approximate approach. Then, the encoded datasets produced

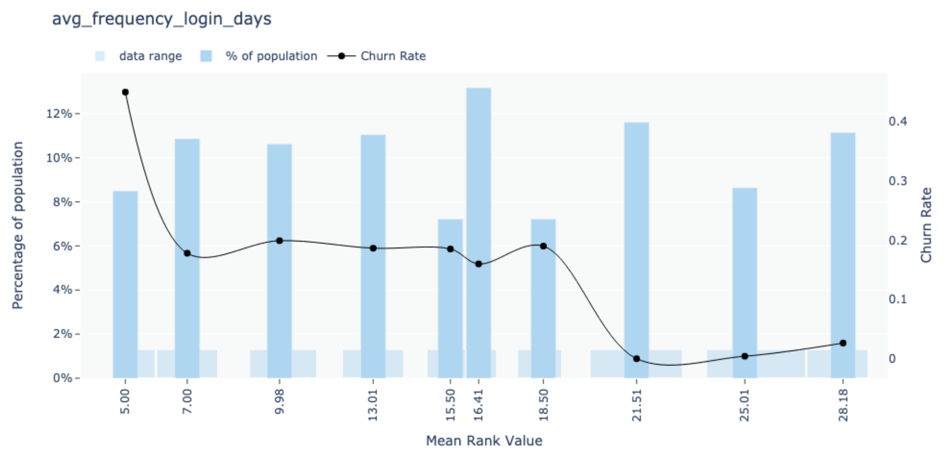

Figure 2 Information Value (IV) analysis graph for the feature $avg\_frequency\_login\_days$. The blue bar represents the bin's distribution in the whole population. The reason why the bins have different distribution is that during the ranking stage of binning, same values are assigned to one-higher bin, causing the number of data points for that bin to be higher. The light blue colored bar aims to show the minimum and maximum values for that specific bin. Finally, the churn rate decreases as the value on $x - axis$ increases.

from Ordinal, Binary, and One-Hot encoding are concatenated to create the preprocessed dataset that is clean and ready for modeling directly, or for further processing.

4.4.2 Feature Selection Using Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is an unsupervised method for reducing feature dimensionality. Its objective is to maximize variance with fewer features. PCA involves calculating eigenvectors from the normalized dataset and selecting the eigenvector with the highest eigenvalue, representing the direction along which the data is most dispersed Alpaydin et al. (2014).

PCA was applied to 91 features of the churn dataset. A minimum variance of 95% was chosen for the resulting features to minimize loss in informative components. Application of PCA resulted in selecting 40 features. Notably, the number of features selected via PCA exceeds that of the IV approach, which selected 31 features.

4.5 Training Datasets Generated by Applying Sampling Techniques

We generate three datasets by applying IV feature selection and no upsampling, SMOTE, upsampling, ADASYN upsampling named as IVR, IVS, and IVA respectively. The fourth dataset is named as PCAS generated by applying PCA feature selection and SMOTE upsampling.

4.5.1 Random Oversampling (ROS)

Random Oversampling is a straightforward technique used to address class imbalance in a dataset. It involves randomly duplicating instances of the minority class until the number of minority and majority class instances are equalized. This method simplifies the task of

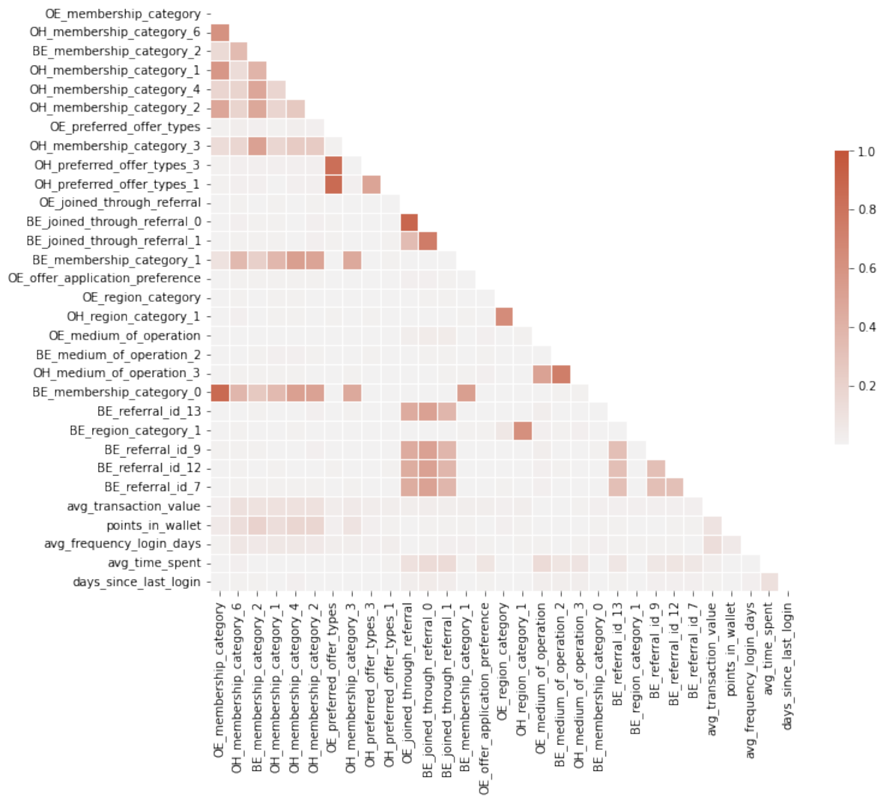

Figure 3 Correlation matrix before dropping highly correlated features. Dark brown areas represent high correlation.

balancing an imbalanced dataset by creating new instances of the minority class through random duplication He and Ma (2013).

4.5.2 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE (Synthetic Minority Over-sampling Technique) has been widely utilized as an upsampling method since its introduction by Chawla et al. (2002). The method creates pseudo-random data in the feature space by following these steps:

1. Select a random instance of the minority class
2. Create a synthetic instance by:
   (a) Select K neighbors ($K = 5$ in this research)
   (b) Choose a random neighbor amongst K neighbors
   (c) Randomly selecting a point along the linear distance from the source point to the neighbor.
3. Repeat above steps until the number of instances in the $n_{minority} = n_{majority}$

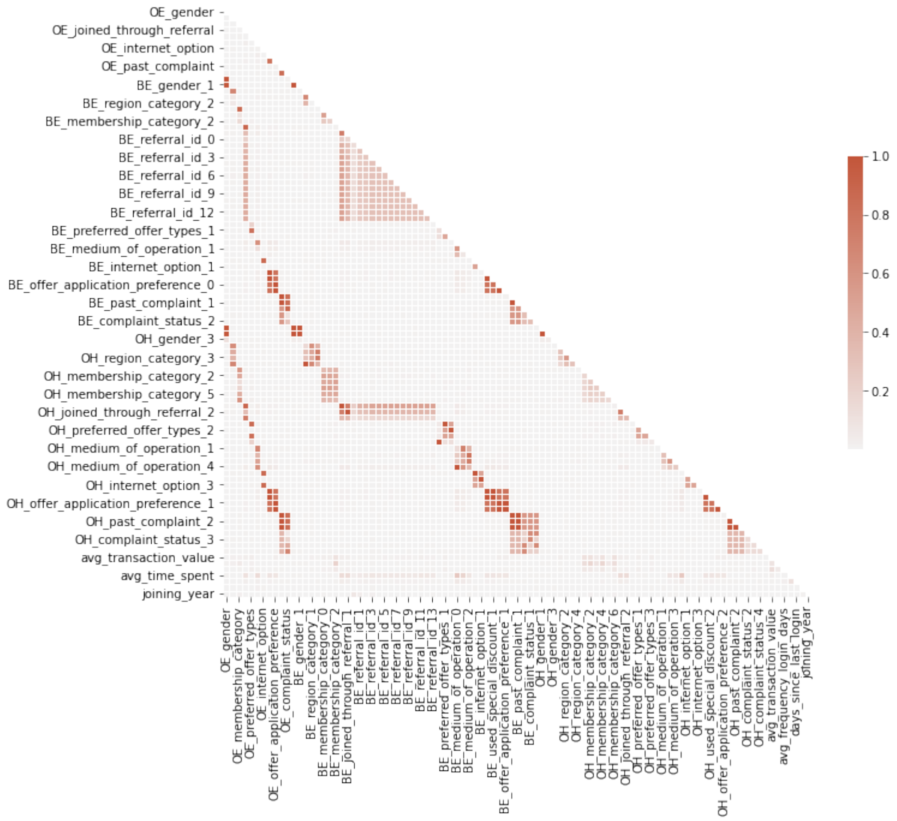

Figure 4 Correlation matrix after dropping highly correlated features. The features that are less significant are discared.

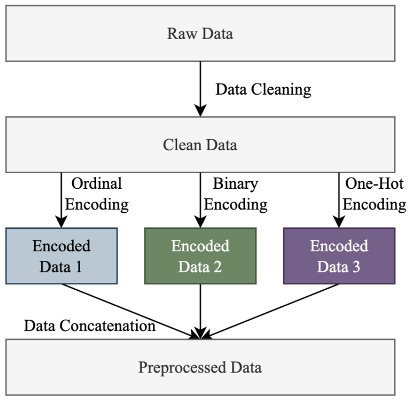

Figure 5 Data preprocessing steps applied to churn dataset.

Those steps are visualized for a two-feature case in Figure 6. Figure 7 demonstrates an example of before and after visualization of SMOTE method applied on the churn dataset. As can be seen from the figure, newly generated data instances have expanded the original

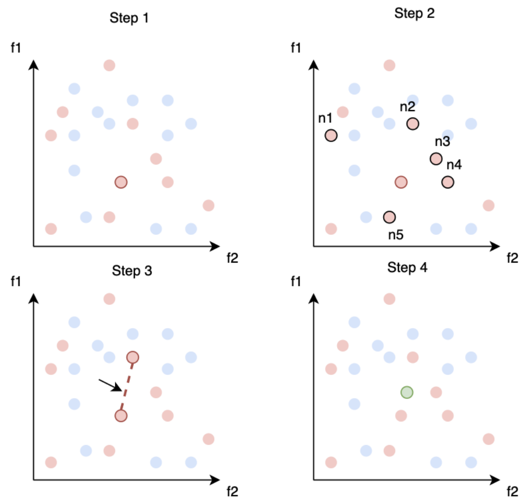

Figure 6 An example iteration of SMOTE method. Assuming the red-colored class is the minority, one instance is randomly selected (Step 1). Then, the nearest $K = 5$ neighbors are identified and one of them is randomly picked (n2). Lastly, a random point in the Euclidean space between origin point and n2 was selected and generated as the synthetic instance.

data. Chawla et al. (2002) observed that the area under the ROC curve benefits from SMOTE upsampling and concludes that SMOTE may increase minority class recognition. Chawla et al. also suggests developing a method to select K in an adaptive and automated manner, and other strategies to generate synthetic instances. The training sets created by SMOTE method are notated as IVS and PCAS, which also indicates the feature selection method in the abbreviation.

4.5.3 Adaptive Synthetic Sampling (ADASYN) Technique

He et al. (2008) introduced ADASYN (Adaptive Synthetic Sampling), which builds upon the SMOTE method with a slightly different approach to creating new instances. ADASYN aims to balance the minority class by generating synthetic instances and assigning a weight to each minority class instance, representing its level of difficulty to learn. This method creates a “thickness” in instances by assigning higher weight to them and thereby generates more instances around a data point. Consequently, it shifts the decision boundary in the modeling stage towards the underrepresented extremes.

Figure 7 and Figure 8 show the class balances after applying the three upsampling methods on the training set. Figure 8 shows ROS and SMOTE equalize the minority and majority classes exactly while ADASYN results in a slightly (negligible) less balanced dataset. Additionally, as can be seen from Figure 7, ROS generates an equalized dataset while condensing the minority class, while the generated instances of SMOTE and ADASYN are more spread out.

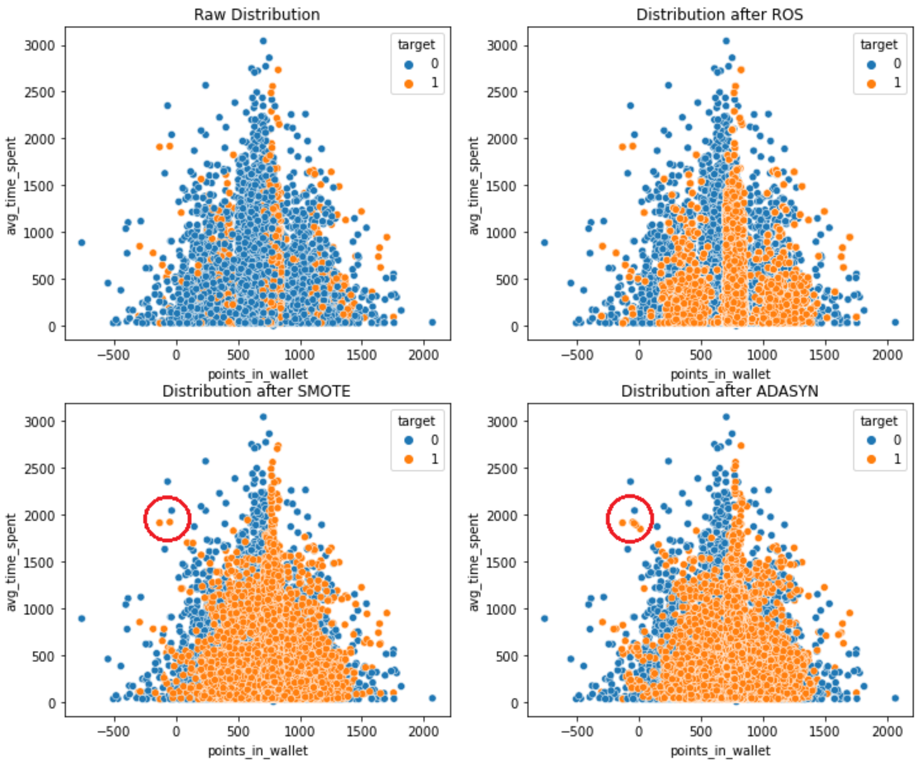

Figure 7 Visual representation of sample generation: $x - axis$ represents the continuous feature $points\_in\_wallet$ and $y - axis$ represents avg_time_spent. Class distribution after ROS is more condensed than SMOTE and ADASYN as expected since ROS does not add offset to generated instances. Also, notice the red circle in SMOTE and ADASYN graphs. In SMOTE, the distant instances are treated the same way as instances that are close to other minority instances. Therefore, SMOTE produces only one extra instance inside the red circle, while ADASYN produces four extra instances for the same data point.

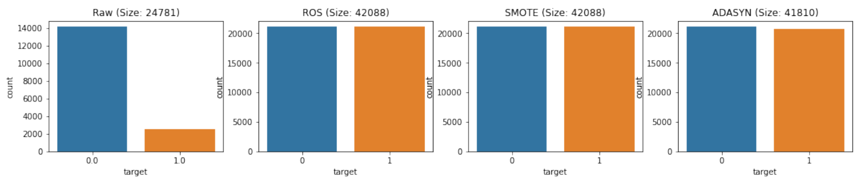

Figure 8 Class balance after applying upsampling methods. In the original dataset, the data belonging to class 1 in the training set was 15.08%, which is risen exactly to 50% after ROS and SMOTE, and close to 50% after ADASYN.

5 Proposed Methodology for Churn Risk Classification Based on Machine Learning

This section describes methods used for modeling. Common machine learning methods such as Logistic Regression and Random Forest are chosen to have a basis for comparison rather than to investigate their performance. Recent methods are chosen to investigate their performance, such as the Deep Cascading Forest Zhou and Feng (2017) and its modification. Deep Neural Networks are included to have a deeper comparison between tree-based models and neural networks. This way, the research manages to compare likelihood-based, tree-based, and layer-based methods on the churn dataset.

5.1 Logistic Regression (LR)

Logistic Regression seeks to maximize the likelihood function, which in the case of binary classification is binary cross-entropy, achieved using the sigmoid function. Convergence in Logistic Regression can be determined by either a predefined number of iterations or a minimum required improvement on the error. In our research, we employ a maximum of 1,000 iterations or stop early if there is no improvement or convergence.

5.2 Random Forest (RF)

Random Forest is an ensemble model that utilizes bagging to create a predetermined number of base estimators, often referred to as decision trees. These base estimators generate predictions independently, and the final predictions are obtained by averaging the predictions made by each base estimator Alpaydin et al. (2014). Ultimately, bagging generates L datasets which are independently used for creating decision trees. Once decision trees are built, majority agreement is used for voting. The predictions generated from the base learners (decision trees) are counted and the class that gets most votes is the final prediction for a given data point.

5.3 XGBoost (XGB)

XGBoost is an optimized method built on top of the Gradient Boosting method. The optimization is made by parallelly boosting the trees so that the XGBoost's performance is highly efficient and accurate. As Chen and Guestrin (2016) describes, the Gradient Boosting method is a tree-based method where instead of creating ensemble of trees, the model runs in an additive manner where L estimators are generated in each iteration and the best estimator among L that minimizes the objective function is chosen, this is a binary logistic function. A set of prespecified parameters are used to define the L and the number of iterations to be made for convergence. Although there are rule-of-thumb parameters, our research makes use of randomized search to find the best combination among possible parameter sets.

5.4 Deep Cascading Forest (DCF)

Zhou and Feng (2017) proposed an ensemble tree-based method called Deep Cascading Forest (DCF). This method was shown to give good results when the available training data is not enough to train large deep learning models. The layered architecture of DCF can be found in Figure 9.

We build DCF model on the churn dataset, and replace the predictor layer of DCF with XGBoost to further improve the performance. This approach was chosen since XGBoost has shown good performance on various machine learning tasks. The XGBoost combined with DCF method is referenced as *DCFX* in our research.

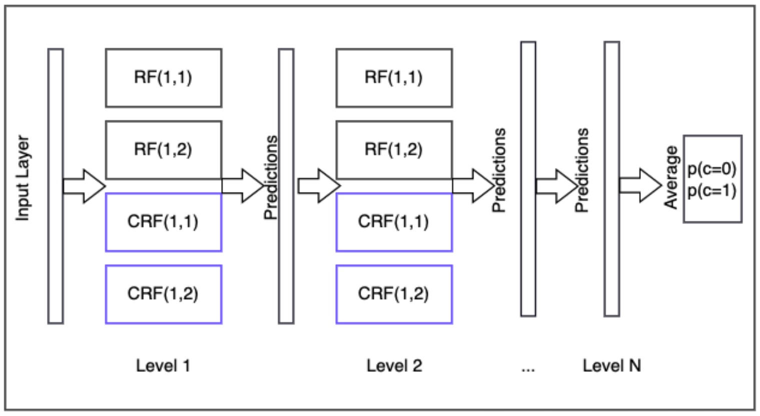

Figure 9 Architecture of Deep Cascading Forest (DCF) model used in this research. Each level has a default configuration of two random forests and two completely random forests. DCF is an ensemble of ensembles method that combines the scores produced at each layer to create the next layer's forests Zhou and Feng (2017). For the Random Forests (RF) the features are selected based on the best Gini index. In Completely Random Forests (CRF) features are selected randomly in each split and uses all features in training set. The predictor layer produces final predictions by averaging output scores at the level N.

5.5 Deep Neural Network (DNN)

Deep Neural Networks (DNNs) consist of multiple hidden layers, with each layer's output serving as an input to the subsequent layer. This hierarchical structure allows DNNs to learn features at various levels of abstraction, ultimately producing classification probabilities or regression values in the final layer. The primary advantage of DNNs lies in minimizing human oversight by automatically learning relevant features from the data Alpaydin et al. (2014). DNNs can have multiple hidden layers, with each layer potentially containing a different number of nodes or neurons. The activation function used in each layer determines how a single node will be activated for a given input instance, playing a crucial role in the network's ability to learn complex patterns and relationships within the data. Lastly, the predictor layer combines probabilities produced in the next-to-last dense layer to come up with a single prediction for a data instance. This research also investigates the effect of the predictor layer's activation function, comparing Rectifier Linear Unit (ReLU) Nair and Hinton (2010), Sigmoid function, and XGBoost.

Deep Neural Networks (DNNs) are more susceptible to overfitting compared to simpler models. One approach to mitigate the overfitting problem is to reduce model complexity by introducing a dropout layer between dense layers. A dropout layer randomly deactivates a fraction of neurons during training, effectively reducing the interconnections between neurons and preventing over-reliance on specific features or patterns in the data. By randomly dropping out neurons, the model learns more robust and generalized representations, thus reducing the risk of overfitting. This technique encourages the network to learn more diverse and robust features, improving its ability to generalize to unseen data. As can be seen from DNN-Model 1 results in a smooth convergence after 50 epochs. On the other hand, if the performance does not improve over epochs, the learning rate could be reduced by a factor such as 0.1 to help reach the local optima. Additionally, DNN-Model 3

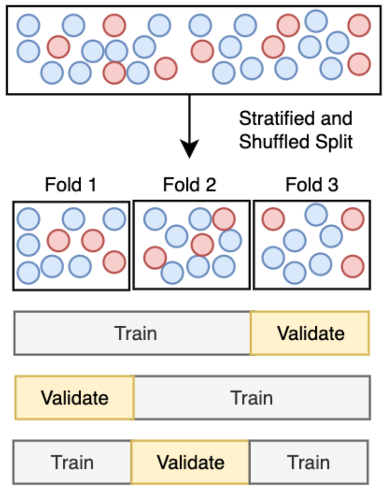

Figure 10 Three-Fold Cross Validation (CV) example. Data is split in such a way that one instance may be present in more than one-fold and some instances may not be present in any of the folds. Each fold has a validation set and training set. The final result is the average score produced from three-fold validation.

includes the sigmoid activation in prediction layer while in *DNNX* we replace the prediction layer by XGB estimator.

5.6 Cross Validation (CV)

Cross Validation is a validation method where the data is split into K fold and for each fold *i,* all the folds except fold *i* are included in the training and fold *i* is used as an out-of-sample validation set to minimize possible overfitting. $K = 3$ was picked for this research to have enough data points for each fold to eliminate bias. The split method used in this research is stratified and shuffled split. Stratification preserves the class ratios across splits while shuffling adds randomization to the splits. An example of K-Fold CV is shown in Figure 10.

5.7 Randomized Search Cross Validation (RSCV)

Grid Search is a searching method that generates all permutations from a given parameter set with multiple options. For example, if there is a set of $n_{estimators}$ with a size of K and another set of $m_{\max\_nodes}$ with a size of L, Grid Search will run $K \times M$ models with the given parameters (Figure 11). To run all combinations is time consuming and could be inefficient, because (1) each iteration takes some amount of time and (2) if a parameter combination gives poor results, then it is likely that its neighbors could give similar results. A Randomized Search, however, is a Grid Search method with a constraint of maximum number of combinations. Randomized Search runs exactly the given the number of combinations by choosing randomly from the parameter grid, which is the set of all possible combinations of parameters. CV is continuously applied during the randomized search and test scores are computed for each parameter combination. Finally, we select the combination that returns the best test score.

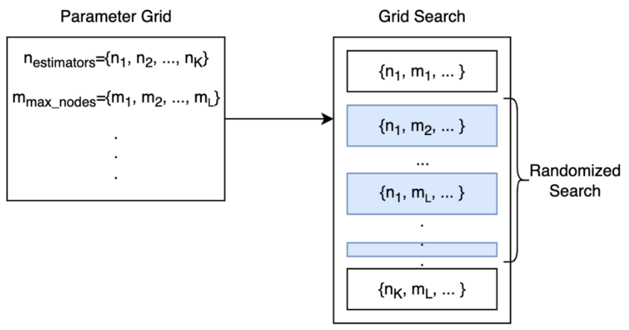

Figure 11 Example of a Grid Search and Randomized Search. Grid Search considers all possible combinations while Randomized Search only picks a given number of combinations. The number of combinations can get extremely large if more parameters are added to the grid.

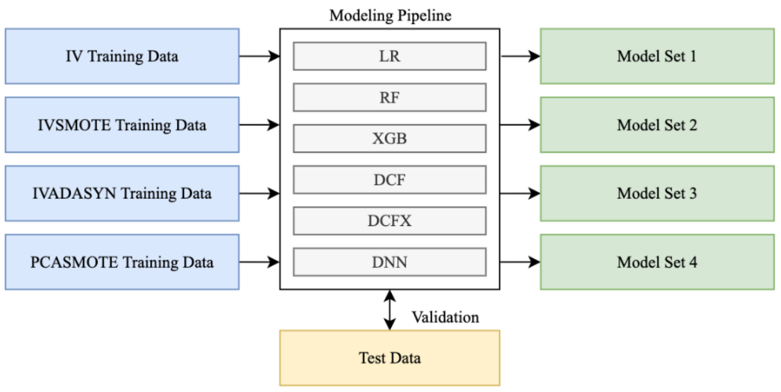

Figure 12 Diagram showing the experiment setup. There are four model sets generated at the end. Each of them consists of six machine learning models. This leads to a comprehensive comparison across various feature selection, oversampling, and machine learning methods.

6 Experiment Setup and Model Evaluation Metrics

There are three main experiment comparisons that our research aims to perform:

- The effect of IV and PCA feature selection
- No oversampling. SMOTE and ADASYN oversampling over raw data
- Machine Learning methods compared to each other, namely, Logistic Regression (LR), XGBoost (XGB), Random Forest (RF), Deep Cascading Forest (DCF), Deep Cascading Forest with XGBoost (DCFX), Deep Neural Networks (DNN), Deep Neural Networks with XGBoost (DNNX).

The modeling approach followed in this research is shown in Figure 12. A detailed description of parameter set used in the experiment is presented in Table 1.

6.1 Model Evaluation Metrics

The metrics used for model evaluation are briefly described as follows:

- **Accuracy** answers the question "How accurately does the model predict the label correctly?" and its formula is $(TP + TN)/(TP + TN + FP + FN)$.
- **Precision** is used to answer, "When the model predicts true, how many times is it actually true?" and calculated by $TP/(TP + FP)$.
- **Recall** or TP-Rate answers, "From all true labels, how many times the model predicts true?" and therefore calculated by $TP/(TP + FN)$.
- **Specificity** answers, "From all false labels, how many times the model predicts false?" and thus calculated by $TN/(TN + FP)$.
- **F-1 score** is the harmonic mean of precision and recall metrics such that it produces a single metric that benefits from both metrics. The formula for F-1 score is $2 \times (Precision \times Recall)/(Precision + Recall)$. Notice that the maximum value of F-1 score can be 1 and minimum can be 0. Also, note that a score closer to 1 means better performance in terms of both precision and recall. If one of them is significantly lower than the other, then the F-1 score will also be lower.
- **AUROC** is the area under the Receiver Operating Characteristics (ROC) curve. The ROC curve is drawn by calculating recall (or TP-Rate) and specificity (or FP-Rate) for every possible threshold in prediction probabilities. Note that when threshold is set to 1, both TP-Rate and FP-Rate must be 1, and similarly, when threshold is set to 0, both TP-Rate and FP-Rate must be 0. Therefore, the ROC curve always starts at (0,0) and ends at (1,1). A random model therefore will result in 50% AUROC score and the perfect model will have a 100% AUROC score.

6.2 Determining an Optimal Cutoff Point for Assigning a Category

Determining the optimal threshold for classifying the minority class, i.e., churned customers involves calculating the True Positive Rate (TP-Rate) and False Positive Rate (FP-Rate) for every possible threshold. Subsequently, the Area Under the Receiver Operating Characteristic (AUROC) curve is calculated for each threshold. The threshold that maximizes the AUROC is considered optimal. In essence, the point where the TP-Rate intersects with the FP-Rate represents the optimal threshold for a given model. This threshold strikes a balance between correctly identifying positive instances (True Positives) while minimizing false alarms (False Positives).

Table 1: Description of Parameters used in Various Models on the Churn Dataset

Model Name, Datasets, Parameter(s)
Model Name: LR **Datasets: IVR, IVS, IVA, PCAS**

Continued on next page

Table 1: Description of Parameters used in Various Models on the Churn Dataset (Continued)

max_iter=1000. Maximum number of iterations for running update function.

penalty=l2. The regularization type is used to penalize the loss function in each iteration. L2 is equal to the squared magnitudes of coefficients.

Model Name: RF

Datasets: IVR, IVS, IVA, PCAS

n_estimators=50. Number of decision trees to be used in the forest.

max_depth=6. Maximum depth of a decision tree.

min_samples_split=100. Minimum number of data instances required to make a split from a node.

min_samples_leaf=40. Minimum number of data instances required in a leaf node.

max_leaf_nodes=45. Maximum number of leaf nodes. Note that a tree structure with a depth N usually has 2^N nodes.

bootstrap=True. Use bootstrapping when generating subset for each estimator.

Model Name: XGB

Datasets: IVR, IVS, IVA

n_estimators: [100, 200]. Number of decision trees.

learning_rate: [0.01, 0.05, 0.1]. The value is used to determine how fast the estimator will learn the target features. A very high learning rate could result in missing the local optima while a very low learning rate may take too long to converge.

min_split_loss: [0, 1, 5]. The gamma value to be used for limiting the creation of new splits by setting a minimum threshold. Split aims to reduce the loss.

max_depth: [3, 4, 5]. Maximum depth of a decision tree.

max_leaves: [4, 8, 12]. Maximum number of leaf nodes.

Model Name: XGB

Dataset: PCAS

Modifications below are added the IV models to improve performance.

n_estimators: [200, 300]

max_depth: [4, 5, 6]

Model Name: DCF

Datasets: IVR, IVS, IVA

Continued on next page

Table 1: Description of Parameters used in Various Models on the Churn Dataset (Continued)

n_estimators: [2, 3, 4]. Number of estimators in each layer. *n_trees: [100, 200].* Number of decision trees to be built in each estimator. *max_layers: [2, 3].* Number of layers to be used in the deep forest. *max_depth: [3, 4, 5].* Maximum depth of a decision tree. *min_samples_leaf: [50, 100].* Maximum number of leaf nodes.
Model Name: DCF **Dataset: PCAS** Modifications below are added the IV models to improve performance. *n_estimators: [200, 300]* *max_depth: [4, 5, 6]*
Model Name: DCFX **Datasets: IVR, IVS, IVA** Same parameter set as that in DCF-IV model.
Model Name: DCFX **Datasets: PCAS** Same parameter set as that in DCF-PCA model.
Model Name: DNN **Datasets: IVR, IVS, IVA, PCAS** **Model 1:** Normalization Layer Dense Layer with 200 nodes and ReLU activation 40% Dropout Layer Dense Layer with 100 nodes and ReLU activation 30% Dropout Layer Dense Layer with 50 nodes and ReLU activation Prediction Layer **Model 2:**

Continued on next page

Table 1: Description of Parameters used in Various Models on the Churn Dataset (Continued)

Normalization Layer
Dense Layer with 512 nodes and ReLU activation
60% Dropout Layer
Dense Layer with 256 nodes and ReLU activation
60% Dropout Layer
Dense Layer with 128 nodes and ReLU activation
Prediction Layer
Model 3:
Normalization Layer
Dense Layer with 128 nodes and ReLU activation
Dense Layer with 64 nodes and ReLU activation
40% Dropout Layer
Dense Layer with 32 nodes and sigmoid activation
Prediction Layer
Model 4:
Normalization Layer
Dense Layer with 128 nodes and ReLU activation
Dense Layer with 64 nodes and ReLU activation
Prediction Layer

7 Experiment Results and Discussion

The subsections below discuss the results of machine learning models generated on churn dataset using IV and PCA for feature selection and no upsampling, SMOTE and ADASYN for oversampling.

7.1 DNN Loss Convergence

Figure 13 shows the loss by epoch plot of each of the four DNN models on IVS dataset. For Model 1, a desirable convergence of loss between training and validation sets is observed. Starting from high amount of loss, the model converges to 0.21 after ~30 epochs. On the other hand, Model 2 fails to converge and underfits. The DNN model losses suggests that the dropout layers in Model 2 cause a vast amount of information loss, therefore the DNN cannot learn at all. On the other hand, Model 3 results in overfitting although the amount of loss in the final epoch is less than the Model 1. Lastly, Model 4 shows a fluctuating loss between epochs. This is because of the low number of nodes in both layers of Model 4.

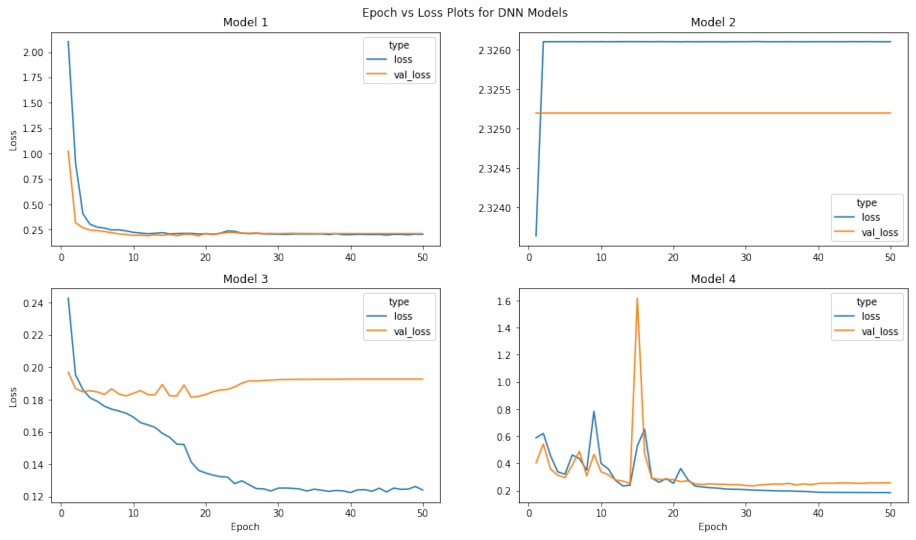

Figure 13 Epoch-by-epoch training and validation loss for four different configurations of DNN models on the IVS dataset.

With the low number of nodes, the model becomes unsteady and is undesirable. Ultimately, we include Model 1 in our evaluation since it has the most appropriate loss plots.

7.2 IVR Model - Information Value Features and No Oversampling

In the IVR dataset the features are selected via the IV method without oversampling. Machine learning methods described in Section 5 were applied to the IVR dataset. Following parameter optimization, a 2% threshold was implemented for the AUROC score to prevent overfitting. Table 2 displays the test set performance of machine learning models on the IVR dataset. According to the table, the DCFX model achieved the highest AUROC score of 89.1%. This score notably outperforms other models, with the exception of the XGBoost model, which attained a score of 88.7%. The Random Forest (RF) model produced competitive results, ranking fourth among all models in terms of AUROC. However, the DNN model for IVR failed to converge with the given parameter set. This indicates that further adjustments or exploration may be necessary to achieve convergence with the DNN model. Although DNNX model had slightly better AUROC score than the DNN model, it is behind the XGB and DCFX.

Table 2: Performance of IVR Model Set

Model	Accuracy	Precision	FPR	TPR	F1	AUROC
LR-IVR	89.0%	71.2%	3.3%	45.6%	55.6%	71.2%
XGB-IVR	89.8%	61.6%	9.6%	86.9%	72.1%	88.7%
RF-IVR	87.1%	54.4%	13.1%	88.0%	67.2%	87.5%
DCF-IVR	87.8%	56.0%	12.4%	88.9%	68.7%	88.2%
DCFX-IVR	89.5%	60.3%	10.4%	88.7%	71.8%	89.1%

Model	Accuracy	Precision	FPR	TPR	F1	AUROC
DNN-IVR	85.1%	50.3%	15.1%	85.7%	63.3%	85.3%
DNNX-IVR	86.5%	53.3%	13.3%	85.7%	65.7%	86.2%

It is useful to analyze results based on various other metrics to gain a comprehensive understanding of model performance. For instance, XGBoost has a better accuracy with 89.8% but the imbalance of the dataset may create a bias in accuracy. On the other hand, Logistic Regression (LR) exhibits the best precision and specificity, however, the low recall indicates that LR fails to capture a significant portion of actual positive instances, resulting in a lower F-1 score and AUROC compared to other methods. This scenario underscores the importance of considering multiple metrics when evaluating model performance. Relying solely on one metric can be misleading and may not provide a comprehensive understanding of a model's capabilities.

Lastly, the resultant confusion matrix from IVR models suggest that LR has achieved a higher accuracy score by simply predicting most of data points as false, missing several actual true classes. On the other hand, XGB and DCFX catches many of the datapoints belonging to the true class with a minimal number of misclassifications. Additionally, while DCF manages to correctly classify datapoints belonging the true class, it also misclassifies a lot of data points, resulting ina a lower TP-Rate.

7.3 IVS Models - Information Value Features and SMOTE Oversampling

The SMOTE models utilizing features selected from the IV method are presented in Table 3. In contrast to the IVR model set, this time DCFX and DNNX yielded the best results, albeit slightly lower than their counterparts in IVR models. For instance, the AUROC for DCFX-IVS is 0.5% lower than DCFX-IVR. This is a good example of how DNNs perform better when fed with more data instances. On the other hand, XGB performed slightly worse than it did for IVR, reason being somewhat overfitting of the model causing more misclassification in test data.

Table 3: Performance of IVS Model Set

Model	Accuracy	Precision	FPR	TPR	F1	AUROC
LR-IVS	83.4%	46.5%	13.8%	67.5%	55.1%	76.8%
XGB-IVS	85.6%	51.3%	14.7%	87.4%	64.6%	86.3%
RF-IVS	86.3%	52.8%	13.7%	86.5%	65.6%	86.4%
DCF-IVS	85.3%	50.8%	14.8%	86.3%	63.9%	85.7%
DCFX-IVS	88.1%	56.8%	12.0%	89.2%	69.4%	88.6%
DNN-IVS	84.8%	49.8%	15.5%	86.7%	63.3%	85.6%
DNNX-IVS	88.5%	58.3%	10.6%	83.9%	68.8%	86.6%

The confusion matrix for LR has significantly higher number of true positives than it had in IVR model set, resulting a 5.7% increase in AUROC score. The highest number of true positives are predicted by DCFX model, which also is the best model in terms of AUROC score. Although the closest model, XGB has only 17 less true positives, the high difference in false positives for XGB caused a 2.2% lower AUROC score.

7.4 IVA Models - Information Value Features and ADASYN Oversampling

The models constructed on the ADASYN dataset demonstrated slightly better performance than SMOTE based models, although they still fell short of the IVR models. The difference is minimal, with only a 0.3% variance between the AUROC scores of DCFX-IVA and DCFX-IVR. It's evident that for the churn dataset, oversampling had no significant positive impact on model performance. Other than the impact of oversampling methods, it is observed that XGB and DCFX were consistently better across all metrics compared to other methods, LR, RF, DCF and DNN. DNNs peformance suffered due to lack of data points leading to a lower activation in network layers. As a result, DNNs may produce weaker models compared to ensemble methods like Random Forests or XGBoost, which are more robust to data scarcity. This highlights the importance of having a sufficiently large and diverse dataset to effectively train complex models like DNNs.. Table 4 shows the performance of IVA model set.

Table 4: Performance of IVA Model Set

Model	Accuracy	Precision	FPR	TPR	F1	AUROC
LR-IVA	80.9%	42.0%	17.3%	70.4%	52.6%	76.6%
XGB-IVA	87.9%	56.3%	12.1%	88.0%	68.7%	88.0%
RF-IVA	85.7%	51.6%	14.5%	86.7%	64.7%	86.1%
DCF-IVA	86.1%	52.4%	13.9%	86.2%	65.2%	86.1%
DCFX-IVA	88.4%	57.5%	11.7%	89.4%	70.0%	88.8%
DNN-IVA	84.9%	50.0%	15.6%	87.7%	63.7%	86.1%
DNNX-IVA	85.0%	50.2%	15.3%	86.8%	63.6%	85.8%

DCFX has 17 less false positives and 2 more true positives compared to IVS DCFX model, which translates into a higher AUROC for this model in the ADASYN generated dataset. Still, the number of false positives is greater than it was in IVR DCFX model by 72 data points.

7.5 PCAS Models - Principal Component Analysis Features and SMOTE Oversampling

The test set results of the PCAS model set are depicted in Table 5. In comparison to other experiment results presented previously, the performance of PCAS is inferior across all models. This experiment highlights that DNNs excel in extracting information when there are more features available. The increase in the number of features by 9 for the PCA dataset compared to the IV dataset made DNN superior over other methods in terms of performance. Also, another comparison may be made within DNN models, DNN and DNNX, which had extremely close performance. It can be concluded that XGBoost predictor layer had slight to no impact on model performance on PCA reduced dataset.

Table 5: Performance of PCAS Model Set

Model	Accuracy	Precision	FPR	TPR	F1	AUROC
LR-PCAS	82.5%	45.7%	18.0%	85.5%	59.6%	83.8%
XGB-PCAS	84.0%	48.2%	15.9%	83.1%	61.0%	83.6%
RF-PCAS	82.4%	45.5%	17.9%	84.2%	59.1%	83.1%
DCF-PCAS	83.1%	46.6%	17.2%	84.3%	60.0%	83.6%
DCFX-PCAS	84.8%	49.8%	15.1%	84.0%	62.5%	84.5%
DNN-PCAS	84.8%	49.8%	15.3%	85.4%	62.9%	85.1%
DNNX-PCAS	85.5%	51.3%	14.0%	83.1%	63.4%	84.5%

A conclusion drawn from the PCAS model set is that the models tend to predict more instances as true compared to the IV model sets. For example, the PCAS DCFX model predicts 1,577 data points as true, whereas the corresponding IVR DCFX model predicts only 1,374 data points as true. Consequently, the PCA method could potentially result in a lower True Positive Rate (TP-Rate), indicating that the models trained on the PCAS dataset may have a tendency to classify more instances as positive, including false positives.

7.6 Model Runtimes

The fastest running method was LR as expected, only 44 seconds for a single iteration. Second fastest method was RF since it only creates 50 decision trees with a maximum depth of 6. The remaining models are iteration or epoch based since. The total runtime of XGB and DCF methods exceeded one hour with the RSCV exploring different parameter combinations.

As the model results suggests, DCFX seemed to outperform other methods in most settings. On the other hand, one drawback of DCFX could be its speed. Table 6 shows the total time elapsed to run whole model and averages for one iteration for each model type on a 4-core CPU, and DCFX is clearly the least-efficient method among all. This is expected as: (1) LR is a base estimator that benefits from training only once, (2) RF and XGB run only one ensembling while DCF (and DCFX) runs N_{layers} times ensembling, and (3) Keras is a highly optimized package that was used to run the DNN models, in comparison the implementation of DCF is not optimized for parallel processing.

Table 6: Runtime Analysis of Machine Learning Models

Model Type	RSCV Iterations / Epochs	Total Runtime	Average Runtime/Iteration
LR	1	00:00:44	00:00:44
RF	1	00:01:04	00:01:04
XGB	50	01:02:47	00:01:15
DCF	20	01:47:06	00:05:21
DCFX	20	02:14:33	00:06:43
DNN	50	00:54:40	00:01:06

Table 7 indicates that most of the machine learning methods performed best on dataset without upsampling. The only notable improvement in model performance was observed for Logistic Regression (LR) on the PCAS dataset, where the AUROC score increased by 12.6% compared to the IVR dataset. Still, the remaining methods like XGBoost or DCFX suggests that the churn dataset should not be oversampled for modeling purposes.

Another conclusion drawn from Table 7 is that the IV feature selection method outperforms PCA feature selection. This observation aligns with expectations, as PCA primarily serves as a dimensionality reduction method, where some information loss may occur during the process. Interesting enough, the number of features for IV datasets were lower than the PCA, 31 versus 40.

Table 7: AUROC Comparison of Models Across Generated Datasets

Dataset	LR	RF	XGB	DCF	DCFX	DNN	DNNX
IVR	71.2%	87.5%	88.7%	88.2%	89.1%	85.3%	86.2%
IVS	76.8%	86.4%	86.3%	85.7%	88.6%	85.6%	86.6%
IVA	76.6%	86.1%	88.0%	86.1%	88.8%	86.1%	85.8%
PCAS	83.8%	83.1%	83.6%	83.6%	84.5%	85.1%	84.5%

8 Conclusion and Future Work

We performed a detailed comparison of feature selection methods, upsampling methods, and machine learning methods on the customer churn risk dataset. Deep Neural Networks are included to have a deeper comparison between tree-based models and neural networks. Our research manages to compare likelihood-based, tree-based, and layer-based machine learning methods on the churn dataset.

The first conclusion drawn from the experiments is that models built on the churn dataset without upsampling performed better than oversampling methods. It's notable that both SMOTE and ADASYN helped stabilize model performance, suggesting their effectiveness in mitigating the impact of class imbalance on model training. The models built on the ADASYN dataset slightly outperformed their SMOTE counterparts. Furthermore, the consistent superiority of XGBoost (XGB) and DCFX across all metrics underscores their robustness and reliability in handling churn prediction tasks. IV model set results performed better than PCA model set. In particular, IVR DCFX model has the best AUROC score with 89.1%. This score is significantly better than any other models except for XGBoost model which produced 88.7%. As the model results suggest, DCFX seemed to outperform other methods in most settings.

Future work shall involve increasing the size of the dataset and implementing variants of Deep Neural Networks and Deep Cascading Forest to further improve churn prediction.

References

C. Mena, A. Caigny, K. Coussement, K. De Bock and S. Lessmann, “Churn Prediction with Sequential Data and Deep Neural Networks," School of Business and Economics, Humboldt University of Berlin. IÉSEG School of Management, 2019.

T. Vafeiadis, D. KI, G. Sarigiannidis and K. Chatzisavvas, “A comparison of machine learning techniques for customer churn prediction," Simulation Modelling Practice and Theory, pp. 1-9, 2015.

A. Ahmad, A. Jafar and K. Aljoumaa, “Customer churn prediction in telecom using machine learning in big data platform," Journal of Big Data, vol. 6, no. 28, 2019.

P. Lalwani, M. Mishra, J. Chadha and P. Sethi, “Customer churn prediction system: a machine learning approach," Computing, vol. 104, pp. 1-24, 2022.

E. Alpaydin, Introduction to Machine Learning, Cambridge, Massachussets: The MIT Press, 2014.

T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

Z.-H. Zhou and J. Feng, “Deep Forest: Towards an alternative to deep neural networks," IJCAI, pp. 3553-3559, 2017.

R. G. Downey and C. King, “Missing data in Likert ratings: A comparison of replacement methods," J Gen Psychol Apr;125(2), pp. 175-91, 1998.

W. McGinnis, T. W. Hbghhy, Andrethrill, C. Siu, C. Davison and N. Bollweg, "Scikit-learn-contrib/categorical-encoding: Release for Zenodo," Zenodo, 2018.

K. Potdar, T. Pardawala and C. Pai, “A comparative study of categorical variable encoding techniques for neural network classifiers," International Journal of Computer Applications 175(4), pp. 7-9, 2017.

C. Seger, “An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing," 2018.

F. Vilarino, P. Spyridonos, J. Vitria and P. Radeva, “Experiments with SVM and stratified sampling with an imbalanced problem: Detection of intestinal contractions," Pattern Recognition and Image Analysis, pp. 783-791, 2005.

P. Verma, “Churn Prediction for Savings Bank Customers: A Machine Learning Approach," Journal of Statistics Applications & Probability vol. 9, no. 3, pp. 535-547, 2020.

A. Rencher, Methods of Multivariate Analysis, New York: John Wiley & Sons, Inc., 1995.

H. He and Y. Ma, Imbalanced Learning: Foundations, methods, and Applications, Wiley-IEEE Press, 2013.

N. Chawla, K. Bowyer, L. Hall and W. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

H. He, Y. Bai, E. Garcia and S. Li, “ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning," in IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, 2008.

V. Nair and G. Hinton. (2010). “Rectified linear units improve restricted Boltzmann machines," In Proceedings of the 27th International Conference on Machine Learning, pp. 807–814.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

HackerEarth’s churn risk rate dataset. Accessed on 11/01/2021. https://www.kaggle.com/datasets/imsparsh/churn-risk-rate-hackerearth-ml

M. Rahman and V. Kumar, “Machine learning based customer churn prediction in banking," 4th International Conference on Electronics, Communication and Aerospace Technology, pp. 1196-1201, 2020.

D. S. Sisodia, S. Vishwakarma, and A. Pujahari, “Evaluation of machine learning models for employee churn prediction," IEEE International Conference on Inventive Computing and Informatics, pp. 1016-1020, 2017.

A. Bhatnagar and S. Srivastava, “A robust model for churn prediction using supervised machine learning," IEEE 9th International Conference on Advanced Computing, pp. 45-49, 2019.