Journal of Combinatorial Optimization (2024) 48:7
https://doi.org/10.1007/s10878-024-01196-w

Customer churn prediction using a novel meta-classifier: an investigation on transaction, Telecommunication and customer churn datasets

Fatemeh Ehsani[1] · **Monireh Hosseini**[1]

Accepted: 13 July 2024 / Published online: 3 August 2024

Abstract
With the advancement of electronic service platforms, customers exhibit various purchasing behaviors. Given the extensive array of options and minimal exit barriers, customer migration from one digital service to another has become a common challenge for businesses. Customer churn prediction (CCP) emerges as a crucial marketing strategy aimed at estimating the likelihood of customer abandonment. In this paper, we aim to predict customer churn intentions using a novel robust meta-classifier. We utilized three distinct datasets: transaction, telecommunication, and customer churn datasets. Employing Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees as the five base supervised classifiers on these three datasets, we conducted cross-validation and evaluation setups separately. Additionally, we employed permutation and SelectKBest feature selection to rank the most practical features for achieving the highest accuracy. Furthermore, we utilized BayesSearchCV and GridSearchCV to discover, optimize, and tune the hyperparameters. Subsequently, we applied the refined classifiers in a funnel of a new meta-classifier for each dataset individually. The experimental results indicate that our proposed meta-classifier demonstrates superior accuracy compared to conventional classifiers and even stacking ensemble methods. The predictive outcomes serve as a valuable tool for businesses in identifying potential churners and taking proactive measures to retain customers, thereby enhancing customer retention rates and ensuring business sustainability.

Keywords Electronic service · Customer churn · Transaction · Telecommunication · Meta-classifier

Extended author information available on the last page of the article

1 Introduction

In today's fiercely competitive business landscape, customers represent the most invaluable asset of every enterprise (Seturi 2024). Companies must not only attract visitors but also engage them, convert them into customers, and retain them over the long term. Building a strong relationship with customers is imperative to prevent their departure (Ehsani and Hosseini, 2023a). This involves offering products and services tailored to their desires and consistently striving to satisfy them (Ehsani and Hosseini, 2023b). Customer churn prediction (CCP) models are designed to identify customers with the highest likelihood of attrition (Rao et al. 2024). By doing so, these models enable businesses to enhance the effectiveness of customer retention campaigns and reduce the costs associated with churn (Verbeke et al. 2012). In this brief overview, we delve into this critical issue.

1.1 Overview

The rise of electronic commerce platforms has sparked significant shifts in customer behavior and decision-making (Pan and Zhou 2020; Wu et al. 2023). Suppliers have leveraged various electronic marketing spaces to showcase their products and services and reach their target customers (Varadarajan et al. 2022). As customers' demands evolve over time and they gain access to numerous options in the electronic marketing environment, their purchasing power and ability to switch between alternatives have increased (Calvano and Polo 2021). Furthermore, they are inundated with vast amounts of information, enhancing their knowledge of products and services (Isa and Nayan 2020). Nearly all electronic commerce organizations employ diverse strategies to engage with visitors and convert them into customers (Anshari et al. 2019). They demonstrate significant perseverance in providing incentives and establishing strong connections to foster long-term relationships with their customers (Mkansi 2022). Offering various amenities to serve customers efficiently, they aim to influence their lifetime value (Calderón-Monge and Ramírez-Hurtado 2022). However, despite marketing programs incorporating various comprehensive sets of business applications to enhance efficiency and effectiveness in managing customer demands, customer churn from one market to another remains a persistent challenge for businesses.

1.2 Problem description

Customer churn is defined as the discontinuation of business between customers or subscribers and a firm or service (Verbeke et al. 2012). It occurs when customers depart from a firm due to dissatisfaction or competitive situations, such as the introduction of new products. Even a reduction in the frequency of purchases by customers can contribute to this issue for businesses (Çelik and Osmanoglu 2019). Customers who opt to stop purchasing can be categorized as intentional or unintentional churners. Intentional churners are customers who actively choose to end their relationship with the marketers, whereas unintentional churn occurs when the firm or service discontinues the customer's access due to reasons like delayed payment, non-

payment for products, or misuse of the firm's data and information (Tsai and Chen 2010). A potential solution to address this challenge is predicting which customers are likely to leave.

1.3 Customers churn prediction

The analysis of customer churn rate involves researching the likelihood of customers moving from one business to another and ceasing to purchase their products or services (Ahn et al. 2019). Customer churn prediction (CCP) is a marketing strategy aimed at assisting marketers in adopting practical approaches to acquire new customers, retain existing ones, and assure customers about the quality of the firm's products and services (Mehralian 2022). The primary objective of CCP is to identify the most valuable customers crucial for maintaining electronic markets (Bachmann et al. 2021). CCP endeavors to enhance satisfaction for both firms and customers by providing insights into customer demands and behaviors (Rane et al. 2023). Another goal of CCP is to establish, manage, and reinforce enduring connections with customers (Asthana 2018). The CCP process seeks to shape perceptions by recognizing customers and providing incentives to retain them (Mahajan and Gangwar 2017). Customer retention occurs when an existing organization meets customers' demands. The revenue of companies typically relies on profits derived from customer purchasing decisions, making customers the most valuable asset of any business. CCP is of utmost importance in revenue estimation models across various industries such as electronic banking, telecommunication, insurance, and retailing (Çelik and Osmanoglu 2019). Every business aims to acquire new customers, upsell and cross-sell to them, and extend their retention period. Maintaining current customers is more cost-effective (Vijaya and Sivasankar 2018; Lalwani et al. 2022) because, in today's saturated and intensely competitive markets, acquiring new customers can be over twenty times more expensive for enterprises than retaining existing ones. Additionally, the value proposition of organizations is directly proportional to the number of active customers engaging in transaction and making multiple purchases over time. The profitability of satisfied and loyal customers is higher and depends on parameters such as expenses, investment capacity, profitability, cash flow, and firm size (Çelik and Osmanoglu 2019).

1.4 Project planning

In the current research, CCP focuses on recognized customers, who represent the most profitable segment in transaction, telecommunication, and customer churn datasets. These datasets contain demographic, historical, and behavioral information of customers, enabling the evaluation of their potential value and prediction of their future desires. We employ churn prediction to identify less engaged customers using an ensemble of classifiers approach. The objective is to pinpoint the rate of customers transitioning from a service to competitors and encourage them to remain with our service. We introduce a novel meta-classifier composed of five more practical classifiers to achieve the highest prediction accuracy. The main contributions of this study are outlined below:

- We analyzed transaction, telecommunication, and customer churn datasets individually, identifying their samples and explaining each feature.
- We performed data preprocessing to clean, encode, scale, and transform the datasets in preparation for further analysis.
- Considering the imbalance in the transaction and telecommunication datasets, we implemented the SMOTE-ENN resampling method to enhance the predictive performance of classifiers.
- We utilized the permutation and SelectKBest feature selection with f_classif to rank the most practical features in each dataset and optimize predictions.
- To select and fine-tune the best parameters for each classifier, we employed the BayesSearchCV and GridSearchCV hyperparameter optimization method.
- We applird Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees as the five base supervised classifiers on each dataset separately, using the same cross-validation and evaluation setup.
- Furthermore, we introduced a novel meta-classifier to evaluate the performance of customer churn predictions, which outperforms other classifiers in terms of accuracy and error rates.
- Additionally, we executed ten runs per dataset to report both a mean and a standard deviation of each evaluation to approve the superiority of the new proposed meta-classifier.
- Finally, we compared the experimental results of the new meta-classifier with Stacking, a powerful ensemble-based classification method, to demonstrate the superior performance of our proposed approach.

2 Datasets description

We conducted our research using three distinct datasets. The "transaction data" pertains to information regarding transactions made by customers when they purchase products or services. This dataset includes customer characteristics, payment information, loan and task details, as well as a chronological sequence of events involving customers (Hand 2018). Telecommunication involve the electronic transmission of information across distances. This information can take the form of voice telephone calls, data, text, images, or video. Telecom providers gather data from various sources, such as call detail records, network logs, and customer history (Loo and Ngan 2012). The "customer churn dataset" encompasses demographic, historical, and subscription details of customers from a service provider company.

2.1 Transaction dataset

The transaction dataset comprises information on 10,000 unique customers of an English bank, encompassing 14 attributes related to the customers. These samples were collected from the bank's database, sourced from www.Kaggle.com. The first column, "Row Number," serves as a non-deterministic analytic function allocating a unique number to each customer in the dataset. The "Customer Id" column assigns a

distinct identification value to prevent duplications. When customers register on the bank's website, they choose a "Surname" for identification purposes during account creation and login. However, these three features solely serve for customer identification and do not influence their behaviors or decisions regarding leaving the bank. Moreover, they exhibit high correlations with each other according to the Correlation Matrix, prompting our decision to remove them. The "Credit Score" attribute is a numerical representation of a customer's creditworthiness, based on factors such as credit history, number of open accounts, types of loans, debt levels, and payment patterns. Lenders utilize this score to assess the likelihood of loan repayment within the specified timeframe, assigning an integer value between 300 and 850 to each customer. Scores above 700 typically indicate promising borrowers who may receive lower interest rates (W. Liu, 2022). Customers with higher credit scores tend to remain loyal and continue utilizing the bank's services. "Geography" denotes the country of origin of the participating customers, including France, Spain, and Germany. "Gender" indicates that the customer is male or female, while "Age" represents how old they are. Generally, younger customers exhibit higher retention rates compared to older ones. "Tenure" reflects the duration for which a customer remains subscribed to the bank, with younger clients typically displaying lower loyalty and higher churn rates. "Balance" encompasses all deposits and withdrawals in a customer's account, reflecting the total amount of money. This variable is significant in churn prediction, as customers with lower balances are more likely to leave. "NumOfProducts" indicates the number of products purchased through the bank, while "Has Cr Card" specifies whether a customer possesses a credit card. Active customers, denoted by "Is Active Member," engage in frequent transactions, indicating loyalty to the bank. "Estimated Salary" represents the approximate earnings of each customer. Similar to balance, customers with higher salaries tend to engage in more transactions. The transaction dataset focuses on binary classification, with "Exited" serving as the target variable to evaluate the accuracy of the algorithms employed. "Exited" acts as the class label, with 7,963 customers belonging to the positive class, indicating those who did not abandon the bank and its services. Conversely, the remaining 2,037 customers are classified in the negative class as they opted to leave the bank. This distribution results in an imbalanced binary classification dataset, which may pose challenges during model training and evaluation.

2.2 Telecommunication dataset

In the highly competitive telecommunication industry, the annual churn rate ranges from 15–25%. Customers have a plethora of service providers to choose from and often switch between them actively. The telecommunication dataset used in this research comprises 7043 unique customers and 21 attributes obtained from www.Kaggle.com. The "Customer ID" is a unique identifier for each customer. However, according to the correlation matrix, it has no impact on churn prediction and is solely used for customer identification. Consequently, we have decided to remove it. The dataset includes demographic information about customers: "gender" indicates whether the customer is male or female, "SeniorCitizen" indicates whether the customer is a senior citizen, "Partner" represents whether the customer has a partner,

and "Dependents" signifies whether the customer has dependents. This dataset comprises the services each customer has enlisted for: "PhoneService" shows whether the customer has a phone service, "MultipleLines" demonstrates whether the customer has multiple lines, "InternetService" denotes the customer's internet service provider, "OnlineSecurity" specifies if the customer has online security, "OnlineBackup" affirms if the customer has online backup, 'DeviceProtection' confirms if the customer has device protection, "TechSupport" declears whether the customer has tech support, "StreamingTV" expresses if the customer has streaming TV, and "StreamingMovies" cites whether the customer has streaming movies. The dataset also contains customer account information: "tenure" allocates the number of months the customer has remained with the company, "Contract" assigns the contract term of the customer, such as Month-to-month, One year, or Two years. "PaperlessBilling" infers whether the customer has opted for paperless billing, while "PaymentMethod" refers the customer's payment method, such as Electronic check, Mailed check, Bank transfer, or Credit card. "MonthlyCharges" mentions the amount charged to the customer monthly, while "TotalCharges" reflects the total amount charged to the customer. "Churn" serves as the target variable, indicating whether the customer has left within the last month or not. Among the dataset, 5174 customers belong to the positive class, indicating they have not abandoned the telecommunication services. The remaining 1869 customers have been classified in the negative class as they decided to leave. This imbalance in our binary classification dataset requires attention.

2.3 Customer churn dataset

The customer churn dataset comprises 100,000 unique customers and 9 attributes related to customer historical usage behavior, demographic information, and subscription details sourced from www.Kaggle.com. The attributes are as follows: "CustomerID" dedicates a unique identifier for each customer, and "Name" indicates the customer's name. However, based on the correlation matrix, these two features solely serve for customer identification and have no effect on churn prediction, leading us to remove them. "Age" represents the age of the customer, while "Gender" denotes the gender (Male or Female). "Location" specifies the customer's base, with options including Houston, Los Angeles, Miami, Chicago, and New York. "Subscription_Length_Months" implicates the number of months the customer has been subscribed. "Monthly_Bill" reflects the monthly bill amount for the customer, and "Total_Usage_GB" defines the total usage in gigabytes. "Churn" serves as the target variable, representing whether the customer has churned or not. Out of the dataset, 50,221 customers belong to the positive class, indicating they did not abandon, while the remaining 49,779 have been classified in the negative class as they decided to leave. The churn variables are almost evenly distributed, suggesting no problem of class imbalance. Therefore, we can conclude that our data is balanced.

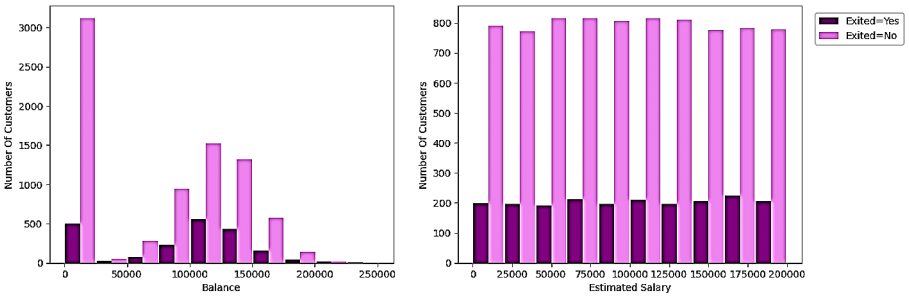

Fig. 1 Histogram plots for continuous features in transaction dataset

Fig. 2 Box plots for numerical features in telecommunication dataset

3 Methodology

Predicting customer churn poses a significant challenge in the business domain. The ability to forecast churn rates has markedly improved with advancements in data mining tools and machine learning classifiers. From this study's standpoint, customer churn prediction involves predicting whether customers belong to churner or non-churner categories based on their historical behavior data. The primary contributions of this study are as follows:

3.1 Data preprocessing and visualization

Data preprocessing approaches encompass transformation methods that redefine a target variable and approaches enabling the estimation of uplift by introducing additional predictor variables incorporated within a standard predictive framework (Devriendt et al. 2021). To preprocess the data, we initiated with data cleaning. This involved conducting Exploratory Data Analysis (EDA) to ensure that all features possessed the correct data type, handling missing values, eliminating duplicate records and outliers from the datasets, and checking for rare categories. Some categories may be prevalent in the dataset, while others may appear only in a few observations. Rare values in categorical variables tend to lead to overfitting, especially in tree-based methods. Furthermore, rare labels may be present in the training set but not in the test set, or vice versa, causing the classifier to struggle to accurately evaluate them. Moving to the feature engineering process, some features are continuous, as illustrated in Fig. 1:"Balance" and "EstimatedSalary" in the transaction dataset. Some features are numerical such as those shown in Fig. 2: "TotalCharges", "MonthlyCharges",

and "tenure" in the telecommunication dataset. Next, we applied Feature Encoding techniques to convert categorical features into numerical ones, which is crucial for achieving high accuracy. Then, we scaled numeric features to ensure they had similar scales and were not biased during model training. Across all three datasets, we utilized "Label Encoder" for encoding and employed "Standard Scaler" for feature scaling and transformation. Subsequently, we assessed the correlation between independent and dependent features using the correlation matrix. Highly correlated features that could impact churn prediction quantitatively were removed, as mentioned earlier. Finally, recognizing that the transaction and telecommunication datasets were imbalanced, we addressed this issue by employing the SMOTE-ENN resampling method.

3.2 Solving class imbalance and class overlap

In binary-classification problems, both class overlap and class imbalance pose significant challenges. These issues often exhibit the most substantial negative impact among potential features, leading to poor performance across various classifiers (Vuttipittayamongkol et al. 2021). Class imbalance refers to an uneven distribution of classes within a dataset, where the minority class is notably smaller in size yet holds primary significance with a relatively high misclassification cost. While, class overlap occurs when there is ambiguity between two classes. These problems are typically addressed using a kernel function that maps the original data to a highly dimensional feature space, aiming to maximize the linear separability of the original data within this space. Overlapping data contain regions where the probabilities of different classes are nearly equal, posing challenges for inference due to nearly identical a priori probabilities in these overlapping areas, making it difficult or even impossible to distinguish between the two classes. (Lee and Kim 2018). Many experts suggest that combining oversampling and undersampling methods with data cleansing approaches, such as SMOTE-Tomek and SMOTE-ENN, may concsider as a feasible solution for these challenges (Santos et al. 2022). In our study, we chose to apply the SMOTE-ENN method to overcome this issue because the undersampling process of SMOTE-ENN is more rigorous compared to SMOTE-Tomek. SMOTE (Synthetic Minority Oversampling Technique) achieves oversampling by creating artificial examples based on similarities in the feature space among existing minority samples (Verbeke et al.,. 2012), while ENN (Edited Nearest Neighbour) conducts undersampling by removing majority class samples from both the original and sample result datasets, specifically targeting those instances where the nearest minority class samples lead to misclassification. This approach effectively eliminates majority class instances close to the border where misclassifications occur (Santos et al. 2022). In the current research, we utilized both oversampling and undersampling methods to address the challenges posed by overlapping and imbalanced datasets, such as those found in transaction and telecommunication data. Since the customer churn dataset is balanced, we did not need to apply these techniques. As demonstrated in Code 1, we imported SMOTE-ENN from the "imblearn.combine" module within the Jupyter notebook environment for the transaction dataset. We applied the same code to the telecommunication dataset.

```
# Train Test Splitting
sns.set_theme(style="white")

# initialize X as features and y as target
X = train.drop(['Exited'], axis=1).values
y = train['Exited'].values

# split the data into train and test sets with a test size of 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=21)
```

```
# count the occurrences of target value in
train['Exited'].value_counts()
```

```
Exited
0    7962
1    2038
Name: count, dtype: int64
```

```
from imblearn.combine import SMOTEENN
st=SMOTEENN()
X_train_st,y_train_st = st.fit_resample(X_train, y_train)
X_train_resampled, y_train_resampled = st.fit_resample(X_train, y_train)
print("The number of classes before fit {}".format(Counter(y_train)))
print("The number of classes after fit {}".format(Counter(y_train_st)))
```

```
The number of classes before fit Counter({0: 5569, 1: 1431})
The number of classes after fit Counter({1: 3151, 0: 2173})
```

Code 1 Solving imbalanced data using SMOTE-ENN in transaction dataset

3.3 Feature selection

Feature selection is a technique used in classification to identify and the top relevant features from a larger set of features in a dataset. Feature selection significantly influences the improvement of a classifier's performance and avoid overfitting by reducing the number of features used in a classifier. Researchers have developed methods for feature selection aimed at reducing dimensionality, calculation complexity, and overfitting. These techniques aim to extract the most relevant features from customers' input vectors for prediction (Coussement 2014). One of the most commonly used feature selection methods is SelectKBest, a type of filter-based feature selection method. In filter-based feature selection methods, the feature selection process is independent of any specific machine learning algorithm. Instead, it relies on statistical measures to score and rank the features (Effrosynidis and Arampatzis 2021).

3.3.1 Permutation feature selection

Permutation Feature Importance (PFS) is a technique used in machine learning to assess the importance of different features in a predictive model. PFS is a model inspection method that measures the contribution of each feature to a fitted model's statistical performance on a given dataset. This technique is particularly useful for non-linear or opaque estimators. It involves randomly shuffling the values of a single feature and observing the resulting degradation in the model's score. By breaking the relationship between the feature and the target, PFS determines how much the model relies on that particular feature (Altmann et al. 2010). PFS measures the importance

of a feature by calculating the increase in the model's prediction error after permuting the feature. A feature is considered "important" if shuffling its values increases the model error, indicating that the model relied on the feature for its predictions. Conversely, a feature is deemed "unimportant" if shuffling its values leaves the model error unchanged, indicating that the model ignored the feature for its predictions (Fisher et al. 2019). The permutation importance algorithm is as follows (Fisher et al. 2019):

Input: Trained model m, feature matrix X, target vector y, error measure $L(y,\ m)$

1. Estimate the original model error $e_{orig} = L(y,\ m\,(X\,))$ (e.g. mean squared error)
2. For each feature j do:

- Generate feature matrix X_{perm} by permuting feature j in the data X. This breaks the association between feature j and true outcome y.
- Estimate error $e_{perm} = L\,(y,\ m\,(X_{perm}\,))$ based on the predictions of the permuted data.
- Calculate permutation feature importance as quotient $FI_j = e_{perm}\ /\ e_{orig}$.

3. Sort features by descending FI_j

There are two separate libraries for executing permutation feature importance in the Jupyter notebook environment: the first is "eli5.sklearn" and the second is "sklearn.inspection." Since we utilized most of our experiments from the "sklearn" library, we preferred to use "sklearn.inspection" for the permutation feature importance, as displayed in Code 2, which illustrates PFI on the transaction dataset.

3.3.2 SelectKBest feature selection

SelectKBest is a univariate method that retains only the specified number of highest-scoring features. It employs statistical tests such as the chi-squared test, ANOVA F-test, or mutual information score to rank the features based on their relationship with the output variable. Then, it selects the top K features with the highest scores for inclusion in the final feature subset. SelectKBest is user-friendly and efficiently reduces the feature set to a manageable number, which is particularly crucial when dealing with large datasets (Achal et al. 2023). SelectKBest has two parameters: K and the score function. K represents the number of top features to be selected, while the score function evaluates feature importance (Elmi et al. 2024). There are various types of score functions available. "f_regression" is used for linear regression and computes the F-value between the feature and the target. "mutual_info_regression" is used for regression and calculates the mutual information between two random features. "f_classif," the default parameter, is used for classification and computes the ANOVA F-value between the feature and the target. "mutual_info_classif" is used for classification and computes the mutual information between two discrete features. "chi2" is employed for classification and computes the chi-squared statistics between each feature and the target. "SelectPercentile" selects the highest X% of the features based on the score function (Karthikeyan et al. 2023). The SelectKBest method was

```
•[26]: # Generate permutation importance mean and standard deviation values
from sklearn.inspection import permutation_importance
perm_imp = permutation_importance(rf, X_train_resampled, y_train_resampled, random_state=42)
data_perm = {'imp_mean':perm_imp['importances_mean'],
             'imp_std':perm_imp['importances_std']}
# Store values in a dataframe and sort by largest mean permutation importance value
rf_perm = pd.DataFrame(data_perm, index=x_train_resampled.columns).sort_values('imp_mean')
rf_perm.sort_values('imp_mean',ascending=False,inplace=True)
rf_perm
```

[26]:

	imp_mean	imp_std
NumOfProducts_3	0.004057	0.000430
NumOfProducts_2	0.002029	0.000522
Geography_1	0.001600	0.000679
NumOfProducts_1	0.001229	0.000357
Age_53	0.000800	0.000232
...	...	...
Tenure_6	0.000000	0.000000
Tenure_7	0.000000	0.000000
CreditScore_761	0.000000	0.000000
IsActiveMember_1	-0.000571	0.000452
Gender_0	-0.002314	0.000455

556 rows × 2 columns

Code 2 Permutation feature importance in sklearn liberary in transaction dataset

```
# Define feature selection
from sklearn.feature_selection import f_classif,SelectKBest
fs = SelectKBest(score_func=f_classif, k=3)
# Apply feature selection
fs.fit(X_train_resampled, y_train_resample)
```

```
▾ SelectKBest
SelectKBest(k=3)
```

Code 3 The SelectKBest feature selection

extracted from the "sklearn.feature_selection" module, employing StratifiedKFold for cross-validation (CV). In this study, since we are dealing with a binary classification problem, we decided to perform SelectKBest using "f_classif" and set "k = 3" in the "sklearn.feature_selection" environment, as indicated in Code 3. For instance, in the customer churn dataset, "Age," "Location," and "Monthly_Bill" were identified as the three best-selected features.

3.4 Hyperparameter optimization

Hyperparameter optimization (HPO) involves identifying the optimal parameters and fine-tuning them within the context of machine learning models. This process typically entails iterating over a subset of tuning parameters related to regularization, kernels, or learning rates, in order to find the best combination that maximizes the model's performance (Claesen et al. 2014). GridSearchCV, a powerful module of the "Sklearn.model_selection" package, is utilized for HPO. It iterates through a subset of various parameters, combines these parameters, and fits the model to the training dataset. Subsequently, it identifies the best values and combinations of parameters that yield the highest accuracy. This process involves setting up the hyperparameters in a discrete grid, testing all combinations of values within the grid, and evaluating performance using cross-validation (CV). The optimal mixture of values for the hyperparameters is determined by the point in the grid that maximizes the average value in cross-validation (Shuai et al. 2018). In this study, GridSearchCV was employed to optimize and fine-tune the hyperparameters of the classifiers, with StratifiedKFold chosen as the cross-validation strategy.

3.4.1 GridSearchCV

GridSearchCV, a powerful module from the "sklearn.model_selection" package, is used for hyperparameter optimization (HPO). It iterates through a subset of various parameters, combining them and fitting the model to the training dataset. Subsequently, it identifies the best values and combinations of parameters that yield the highest accuracy. This process involves setting up the hyperparameters in a discrete grid, testing all combinations of values within the grid, and evaluating performance using cross-validation (CV). The optimal mixture of hyperparameter values is determined by the point in the grid that maximizes the average value in cross-validation. GridSearchCV is renowned for selecting the most suitable execution parameters from a predefined set, which has been widely demonstrated to effectively discover and adjust hyperparameters, making it an excellent optimization tool for classifiers (Shuai et al. 2018). GridSearchCV was applied through the "sklearn.model_selection" module within the Jupyter environment. In this study, GridSearchCV was employed to optimize and fine-tune the hyperparameters of the classifiers, with StratifiedKFold chosen as the cross-validation strategy. Code 4 shows the procedure of using GridSearchCV to find and tune the hyperparameters of the AdaBoost classifier.

3.4.2 BayesSearchCV

BayesSearchCV is a tool for Bayesian optimization over hyperparameters. Bayesian optimization is a sequential design strategy for the global optimization of black-box functions that does not assume any specific functional forms. It is usually employed to optimize expensive-to-evaluate functions (Victoria and Maragatham 2021). BayesSearchCV, a powerful module from the "Scikit-Optimize (skopt)" package, is used for Hyperparameter Optimization (HPO). One advantage of BayesSearchCV is its ability to reach the best results more quickly than GridSearchCV with the

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import AdaBoostClassifier

# create options for gridsearch
param_grid = {  'n_estimators': [50, 80, 100],
                'learning_rate': [0.001, 0.01, 0.1]
             }

grid = GridSearchCV(estimator = AdaBoostClassifier(random_state=123),
                    param_grid = param_grid,
                    cv = StratifiedKFold(n_splits=5, random_state=123, shuffle=True),
                    n_jobs=-1,
                    verbose=0,
                    scoring='accuracy')

# fit grid to the mode
grid.fit (X_train_resampled, y_train_resampled)

# Summarize the results
print("Best: %f using %s" % (grid.best_score_, grid.best_params_))
```

```
Best: 0.824571 using {'learning_rate': 0.1, 'n_estimators': 100}
```

Code 4 The GridSearchCV hyperparameters optimization

```
from skopt import BayesSearchCV
from sklearn.ensemble import RandomForestClassifier
from skopt.space import Real, Categorical, Integer
# Define the hyperparameter configuration space
rf_params = {
    'n_estimators': Integer(10,100),
    "max_features":Integer(1,64),
    'max_depth': Integer(5,50),
    "min_samples_split":Integer(2,11),
    "min_samples_leaf":Integer(1,11),
    "criterion":['gini','entropy']
}
clf = RandomForestClassifier(random_state=0)
Bayes = BayesSearchCV(clf, rf_params, cv = StratifiedKFold(n_splits=5, random_state=123, shuffle=True),
                      n_iter=20, n_jobs=-1, scoring='accuracy')
#number of iterations is set to 20, you can increase this number if time permits
Bayes.fit(X_train_resampled, y_train_resampled)
print(Bayes.best_params_)
bclf = Bayes.best_estimator_
print("Accuracy:"+ str(Bayes.best_score_))
```

```
OrderedDict([('criterion', 'entropy'), ('max_depth', 46), ('max_features', 50), ('min_samples_leaf', 2),
             ('min_samples_split', 11), ('n_estimators', 48)])
Accuracy:0.8424288914819881
```

Code 5 The BayesSearchCV hyperparameters optimization

same parameters, although it may not be as precise in some cases. In our work, we employed both GridSearchCV and BayesSearchCV for hyperparameter tuning. The hyperparameters obtained from GridSearchCV remain constant and fixed in each run of the tuning program (Zhao et al. 2024). Code 5 illustrates the procedure of using BayesSearchCV with the Random Forest classifier.

4 Experimental setup

We chose Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees as the five robust and practical classifiers. To identify the most influential features in the datasets, we performed feature selection (FS) methods. We utilized the SelectKBest feature selection method with the default "f_classif" parameter, as well as Permutation feature selection. These two methods assign higher ranks to features based on their importance. We then fine-tuned their hyperparameters to optimize performance. We applied SelectKBest (SKB) and GridSearchCV (GS) together, and Permutation feature selection (PFS) and BayesSearchCV (BS) together for all classifiers separately in each of the three datasets. For ensemble classification, we proposed the "Oracle" meta-classifier, combining the classifiers to achieve the highest precision rate. Additionally, we compared the experimental results with the "Stacking" ensemble meta-classifier. Implementation-wise, we utilized Decision Tree from the "sklearn.tree" library, and Random Forest, AdaBoost, Extra Trees, and Stacking from the "sklearn.ensemble" library. XGBoost was implemented using the "XGBoost" library. Furthermore, we imported the Oracle classifier from the "DESlib" library, which is thoroughly documented and facilitates the execution of classification models to achieve high accuracy. Additionally, we executed ten runs per dataset (per random seed from a list of values [n1, n2, …, n10]) to report both the mean and standard deviation of each evaluation, confirming the superiority of the new proposed meta-classifier.

4.1 Decision tree

A Decision Tree is a tree-shaped structure that derives classification rules from a training dataset through inductive learning. It partitions a large dataset into smaller sets sequentially (Çelik and Osmanoglu 2019). In this structure, the leaves represent class labels, such as "churner" or "non-churner," while branches illustrate the connections of variables leading to these class labels. Despite its inability to capture complex and nonlinear relationships among variables, a Decision Tree often achieves high accuracy in churn prediction problems (Asthana 2018; Vafeiadis et al. 2015). The process of improving a Decision Tree involves two main stages: building and pruning. During the building phase, the dataset is recursively partitioned until all samples in each partition share identical values. In the pruning phase, certain branches, typically those with the highest calculated error rate or containing noisy data, are removed to refine the tree structure (Johny and Mathai 2017).

4.2 Random forest

Random Forest is comprised of multiple tree structures and combines the subspaces and bagging elements of Decision Trees randomly. At each node of the tree, Random Forest selects a subset of variables and then extracts the best split among those variables for that node. Utilizing the bagging technique, Random Forest constructs the training dataset for each tree by de-correlating them, fitting them to bootstrap-resampled datasets. This method selects a random sample of variables after splitting each tree for the subsequent partition. The conclusions drawn from Random Forest

are based on the majority vote of the individual trees. By employing the bagging technique with a large number of trees, Random Forest addresses some weaknesses of non-ensemble Decision Trees, such as overfitting and lack of robustness (Çelik and Osmanoglu 2019).

4.3 Extreme gradient boosting (XGBoost)

XGBoost is a supervised classifier designed for tabular or structured datasets. It employs a Decision Tree classifier with a gradient boosting approach to accelerate computation time, reduce memory usage, and improve accuracy (Çelik and Osmanoglu 2019). Gradient boosting involves a method where residuals and novel algorithm models are used to calculate errors efficiently. These components are then combined to optimally utilize existing resources for training the prediction classifier. Additionally, XGBoost utilizes gradient descent to minimize or reduce the loss function (Lalwani et al. 2022).

4.4 Adaptive boosting (AdaBoost)

AdaBoost is an adaptive supervized classifier utilized for data classification by amalgamating multiple weak or base learners, such as Decision Trees, into a potent learner. While individual learners may be weak, as long as each performs slightly better than random guessing, the final model is proven to converge to a strong learner. Interestingly, AdaBoost, although primarily employed to combine weak base learners, has demonstrated effectiveness in combining strong base learners as well, yielding an even more accurate model. The functioning of AdaBoost involves weighting instances in the training dataset based on the accuracy of previous classifications. This adaptive approach allows AdaBoost to iteratively improve its performance by focusing more on misclassified instances in subsequent iterations (Wyner et al. 2017).

4.5 Extremely randomized trees (Extra Trees)

Extra Trees is a classifier primarily rooted in Decision Trees. Similar to Random Forest, Extra Trees randomizes certain decisions and subsets of data to mitigate overfitting and over-learning from the data. While it shares similarities with Random Forest, such as building multiple trees and utilizing random subsets of features for node splits, Extra Trees differs in two key aspects. Firstly, it does not bootstrap observations, meaning it samples without replacement. Secondly, nodes are split based on random splits rather than the best splits. Additionally, in contrast to Random Forest, where all samples are randomly selected for both features and observations, Extra Trees exclusively randomizes feature selection. This random splitting mechanism often yields results superior to those obtained by Random Forest to some extent (Sharaff and Gupta 2019).

5 Ensemble selection meta-classifier

One of the key considerations in optimizing a multiple classifier system is the selection of a suitable group of classifiers, often referred to as an "Ensemble of Classifiers," from a pool of available options. An essential aspect of this process is ensemble selection, which aims to choose the most effective classifiers from a diverse range to achieve optimal recognition rates (Jan and Verma 2020). The ensemble selection mechanism functions as a meta-classifier, providing a collection of classifier implementations to assess each one's efficiency. This method selects the most appropriate classifiers for class label prediction. This module adopts a fixed and definite subset of the primary ensemble to reduce space complexity, operating in two phases. The first phase, "orders-based selection," involves selecting a relevant objective function to determine the pertinent classifiers. The second phase, "optimization-based selection," entails finding an appropriate search algorithm to apply this criterion effectively (Wang et al. 2021).

5.1 Stacking ensemble selection Meta-classifier

Stacking, also known as stacked ensembles or stacked generalization, represents a robust ensemble selection strategy that amalgamates predictions from numerous base classification models (classifiers) to yield a final prediction with enhanced performance (Dey and Mathur 2023). It stands as an advanced meta-classifier surpassing the concepts of bagging and boosting. Illustrated in Fig. 3, stacking diverges from merely aggregating the outputs of classifiers by incorporating a higher-level model, referred to as the meta-model or aggregator, which is trained on the outputs of the base classifiers to formulate the ultimate prediction. Stacking is instrumental in predicting multiple nodes to construct a novel model and enhance classifier performance. By enabling the training of multiple classifiers to tackle similar problems and leveraging their combined outputs, stacking facilitates the creation of a new model with improved performance (Alexandropoulos et al. 2019). Code 6 demonstrates the incorporation of Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees assembled in the Stacking meta-classifier.

5.2 Oracle ensemble selection meta-classifier

The Oracle stands out as a meta-classifier composition method designed to select the initial classifiers capable of accurately predicting the class label for a given instance. The Oracle is an abstract method that represents an ideal classifier selection scheme. This abstract meta-classifier is considered an ideal model, often estimating the upper bound of classification precision. Typically, the Oracle selects classifiers that reliably anticipate the correct class label, making it an exemplary ensemble selection framework. During a training phase preceding classification, the regions of competence of each classifier are specified. Figure 4 outlines the Oracle procedure, comprising three distinct stages: (1) Generation, (2) Selection, and (3) Aggregation. In Generation, a pool containing M classifiers $\mathrm{C} = \{\mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{M}\}$ trains to produce a collection of

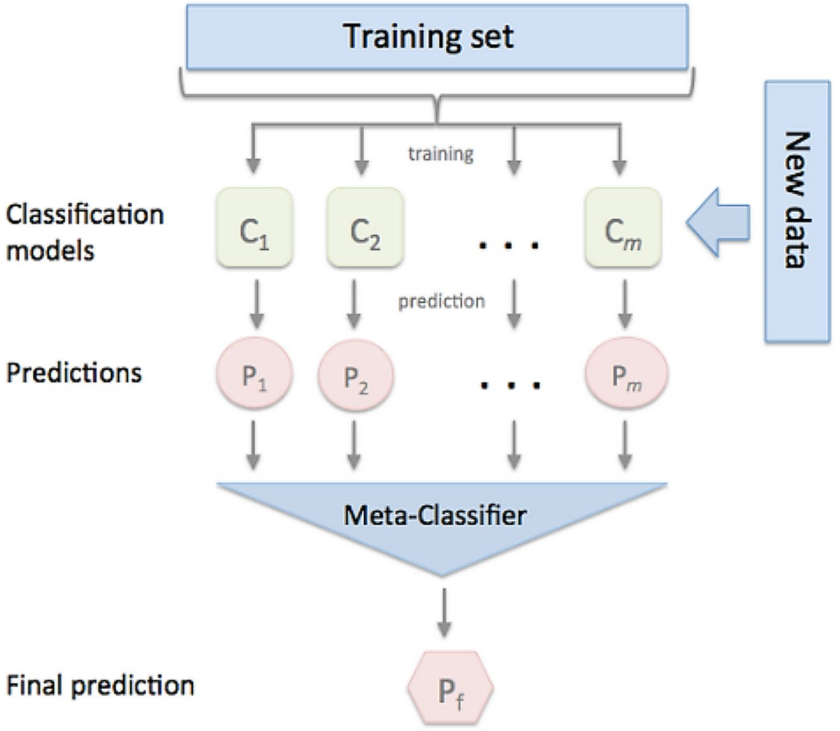

Fig. 3 Stacking ensemble selection meta-classifier procedure

```
[52]: # Stacking
estimators = [('dt', dt),
              ('rf', rf),
              ('xgb', xgb),
              ('ada', ada),
              ('ex', ex)]

Stack = StackingClassifier(estimators = estimators, final_estimator = MLPClassifier())

[53]: Stack.fit(x_train_resampled, y_train_resampled)

[53]: StackingClassifier
      dt: DecisionTreeClassifier
      rf: RandomForestClassifier
      xgb: XGBClassifier
      ada: AdaBoostClassifier
      ex: ExtraTreesClassifier
      final_estimator: MLPClassifier
```

Code 6 The stacking meta-classifier execution

precise, various, and informative classifiers. There are six main strategies for generating a diverse pool of classifiers:

1. Different Initializations: If the training process depends on initialization, employing various initializations may yield different classifiers.
2. Different Parameters: This strategy involves generating base experts with varying configurations of hyperparameters, resulting in different decision boundaries.

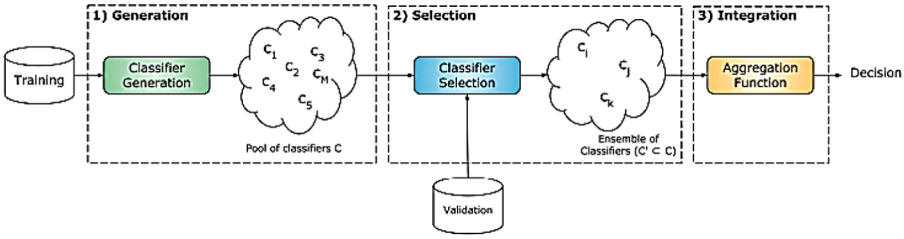

Fig. 4 Oracle ensemble selection meta-classifier procedure

3. Different Architectures: This approach includes training multiple Multi-Layer Perceptron (MLP) neural networks with differing numbers of hidden layers.
4. Different Classifier Models: Here, various classification models are combined to diversify the pool of classifiers.
5. Different Training Sets: Each base classifier is trained using a distinct distribution of the training set.
6. Different Feature Sets: This methodology is applicable in scenarios where data can be represented in distinct feature spaces.

In this study, we adopted the second and sixth strategies, utilizing hyperparameter configurations and feature selection to construct the meta-classifier. In Selection phase, an ensemble of the most suitable classifiers ($C^{'} \subseteq C$) is chosen based on the validation situation. Here, $C^{'}$ is selected during the training phase, according to a selection criterion estimated in the validation dataset. The same ensemble $C^{'}$ is used to predict the label of all test samples in the generalization phase. In the Aggregation phase, the selected base classifiers produce outputs using a compound rule to predict class labels. This phase involves merging the outputs obtained from these classifiers according to a combination rule. The combination of base classifiers can be executed based on class labels, as seen in the Majority Voting scheme, or by utilizing the scores acquired by the base classifier for each class in the classification problem. In this method, the scores from the base classifiers are understood as fuzzy class memberships or the posterior probability that a test sample x_j belongs to a particular class. (Cruz et al. 2018).

As shown in Eq. 1, the level of competence $\delta_{i,j}$ of a base classifier c_i is simply computed as the local accuracy achieved by c_i for the region of competence θ_j. The classifier with the highest level of competence $\delta_{i,j}$ is selected.

$$\delta_{i,j} = \sum_{k=1}^{K} P(w_l \mid x_k \in w_l, c_i) \tag{1}$$

Where, $\theta_j = \{x_1, \ldots, x_K\}$ is the region of competence of the test sample x_j defined on the validation data, K indicates the size of the region of competence and w_l relates the correct label of x_j. The Oracle is classically defined in the literature as a strategy that correctly classifies each query instance of x_j if any classifier c_i from the pool of classifiers C predicts the correct label for x_j. Based on Eq. 2, the level of competence

$\delta_{i,j}$ of a base classifier c_i is equals to 1 if it predicts the correct label for ofx_j, and 0 otherwise.

$$\begin{cases} \delta_{i,j} = 1, & if\ c_i\ correctly\ classifies\ x_j \\ \delta_{i,j} = 0 & otherwise \end{cases} \tag{2}$$

The Oracle represents the ideal classifier selection scheme. It always selects the classifier that predicted the correct label, for any given query sample. The Oracle meta-classifier is a relatively new approach that has received limited attention, particularly in prediction problems, let alone in churn prediction. Due to its static and stable nature, this meta-classifier tends to perform better when integrated with static supervised classifiers. In this study, we employed Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees, as they inherently possess static characteristics. Code 7 exhibits the execution of these five classifiers in the funnel of the Oracle meta-classifier. Additionally, we enhanced the Oracle's structure by incorporating the SelectKBest feature selection along with GridSearchCV separately, and also Permutation feature selection leveraging the BayesSearchCV separately.

6 Evaluation and results

This study focuses on developing customer churn prediction by employing static supervised classifiers on transaction, telecommunication, and customer churn datasets. We implemented Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees in constructing Stacking and Oracle meta-classifiers. BayesSearchCV and GridSearchCV hyperparameter optimizations were applied to each classifier to determine the most suitable parameters. Additionally, we incorporated Permutation and SelectKBest filter-based feature selection techniques. This integration of Stacking and Oracle meta-classifiers with feature selection and hyperparameter optimization aims to enhance churn prediction accuracy. To ensure the superiority of the

```
[41]: # Oracle Meta-Classifier
from deslib.static.oracle import Oracle
# define classifiers to use in the pool
classifiers = [ex, dt, rf, xgb, ada]
# fit each classifier on the training set
for c in classifiers:
    c.fit(x_train_resampled, y_train_resampled)
# define the Meta-Des model
model = Oracle(pool_classifiers=classifiers, random_state=123, n_jobs=-1)
# fit the model
model.fit(x_train_resampled, y_train_resampled)
# make predictions on the test set
y_pred = model.predict(x_test, y_test)
# evaluate predictions
score = accuracy_score(y_test, y_pred)
print('Accuracy: %.5f' % (score))

Accuracy: 0.91800

[42]: Metrics = pd.DataFrame({'Accuracy': round(accuracy_score(y_test, y_pred), 4),
                           'Precision': round(precision_score(y_test, y_pred), 4),
                           'Recall': round(recall_score(y_test, y_pred), 4),
                           'F1 Score': round(f1_score(y_test, y_pred), 4),
                           'ROC_AUC_Score': round(roc_auc_score(y_test, y_pred), 4),
                           'Cohen-Kappa Score':round(cohen_kappa_score(y_test, y_pred), 4)
                          }, index=['Oracle'])
Metrics
```

[42]:

	Accuracy	Precision	Recall	F1 Score	ROC_AUC_Score	Cohen-Kappa Score
Oracle	0.918	0.9918	0.5983	0.7464	0.7985	0.701

Code 7 The Oracle Meta-classifier execution

Oracle meta-classifier, we conducted two different analyses. Firstly, we integrated SelectKBest feature selection along with GridSearchCV for all classifiers across the three datasets. Secondly, we employed Permutation feature selection leveraging BayesSearchCV. The experimental results indicate that incorporating SelectKBest feature selection along with GridSearchCV yields slightly better results compared to Permutation feature selection leveraging BayesSearchCV. However, regardless of the feature selection methods and hyperparameters optimization methods used, the Oracle meta-classifier consistently achieves higher prediction accuracy and lower error rates. Upon compilation of SelectKBest along with GridSearchCV, the accuracy of Oracle reached 91.80%, 90.94%, and 90.73% in the transaction, telecommunication, and customer churn datasets, respectively. This performance surpasses that of each individual classifier (Decision Tree, Random Forest, XGBoost, AdaBoost, Extra Trees), as well as the Stacking meta-classifier. To further validate the superiority of the Oracle meta-classifier, we executed ten runs per dataset to report both the mean and standard deviation of each evaluation, confirming the effectiveness of the newly proposed meta-classifier. Tables 1 and 2, and 3 present the mean and standard deviation of experimental results for all classifiers, Stacking, and Oracle across the transaction, telecommunication, and customer churn datasets, respectively. Oracle also achieves superior mean Precision, Recall, F1 Score, ROC-AUC Score, and Cohen's Kappa Score compared to the other mentioned classifiers, as well as the Stacking meta-classifier, with lower standard deviations in Precision, Recall, F1 Score, ROC-AUC Score, and Cohen's Kappa Score. Considering the ROC-AUC Score as a standard criterion for performance appraisal, we classify classifiers based on their scores. A score below 0.6 denotes a "weak discriminator" between two classes. Scores between 0.6 and 0.7 signify an "acceptable discriminator," while scores ranging from 0.7 to 0.8 indicate an "excellent discriminator." When the score falls between 0.8 and 0.9, the classifier is deemed a "perfect discriminator" (Mallett et al. 2014). Referring to Table 1, in the transaction dataset, the Random Forest classifier yields an ROC-AUC below 0.6, indicating weak discrimination between churner and non-churner classes. Decision Tree and Extra Trees fall within the range of acceptable discrimination. XGBoost, AdaBoost, and Stacking exhibit excellent discrimination. Notably, the Oracle meta-classifier outperforms others with a higher ROC-AUC score. In Table 2, concerning the telecommunication dataset, Decision Tree achieves acceptable discrimination. Random Forest, XGBoost, and Extra Trees demonstrate excellent discrimination. AdaBoost and Stacking perform exceptionally perfect discrimination, yet Oracle surpasses them all. As shown in Table 3, across the customer churn dataset, all classifiers, including Stacking, exhibit weak performance, whereas Oracle excels as an excellent discriminator. Cohen's Kappa Score is a statistical measure used to quantify the level of agreement between two raters (Vetter and Schober 2018). The interpretation of Cohen's Kappa Score results, can be summarized as follows: In Tables 1 and 2, and 3, values less than 0, such as those for Decision Tree, Random Forest, and AdaBoost in the customer churn dataset, suggest no agreement. Values between 0.01 and 0.20, such as those for XGBoost, Extra Trees, and Stacking in the customer churn dataset, indicate "none to slight" agreement. Values between 0.2 and 0.4, observed in Decision Tree and Random Forest across both the transaction and telecommunication datasets, are considered as "fair" agreement.

Table 1 Summary of the results for classifiers in transaction dataset

Classifiers	Feature Selection+HPO	Accuracy		Precision		Recall		F1 Score		ROC-AUC Score		Cohen's Kappa Score	
		Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv
Decision Tree	SKB+GS	0.832	0.242	0.671	0.136	0.333	0.145	0.445	0.204	0.646	0.163	0.360	0.159
	PFS+BS	0.820	0.237	0.669	0.126	0.361	0.148	0.443	0.199	0.645	0.161	0.358	0.157
Random Forest	SKB+GS	0.824	0.221	0.914	0.145	0.142	0.103	0.246	0.106	0.569	0.304	0.202	0.108
	PFS+BS	0.819	0.277	0.900	0.433	0.175	0.117	0.214	0.110	0.534	0.289	0.197	0.103
XGBoost	SKB+GS	0.866	0.269	0.808	0.303	0.439	0.138	0.569	0.176	0.706	0.264	0.498	0.155
	PFS+BS	0.863	0.258	0.800	0.344	0.468	0.132	0.550	0.168	0.701	0.260	0.462	0.163
AdaBoost	SKB+GS	0.858	0.243	0.719	0.352	0.490	0.179	0.583	0.132	0.721	0.301	0.502	0.167
	PFS+BS	0.850	0.235	0.712	0.394	0.499	0.166	0.580	0.158	0.718	0.298	0.499	0.159
Extra Trees	SKB+GS	0.853	0.205	0.740	0.418	0.419	0.134	0.535	0.173	0.691	0.297	0.456	0.162
	PFS+BS	0.849	0.209	0.733	0.414	0.423	0.139	0.533	0.178	0.684	0.287	0.435	0.158
Stacking	SKB+GS	0.860	0.214	0.765	0.373	0.490	0.174	0.598	0.187	0.726	0.247	0.523	0.132
	PFS+BS	0.856	0.208	0.760	0.394	0.501	0.182	0.596	0.190	0.714	0.252	0.501	0.136
Oracle	SKB+GS	0.918	0.012	0.991	0.004	0.598	0.023	0.746	0.051	0.798	0.042	0.701	0.043
	PFS+BS	0.900	0.018	0.989	0.008	0.547	0.028	0.729	0.062	0.779	0.050	0.690	0.056

Table 2 Summary of the results for classifiers in telecommunication dataset

Classifiers	Feature Selection+HPO	Accuracy		Precision		Recall		F1 Score		ROC-AUC Score		Cohen's Kappa Score	
		Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv
Decision Tree	SKB+GS	0.793	0.219	0.460	0.334	0.545	0.142	0.499	0.218	0.655	0.173	0.285	0.189
	PFS+BS	0.786	0.223	0.454	0.339	0.532	0.138	0.491	0.224	0.651	0.167	0.282	0.197
Random Forest	SKB+GS	0.794	0.206	0.570	0.138	0.539	0.136	0.554	0.144	0.780	0.299	0.390	0.128
	PFS+BS	0.787	0.211	0.568	0.141	0.531	0.130	0.549	0.153	0.777	0.284	0.384	0.134
XGBoost	SKB+GS	0.759	0.218	0.564	0.186	0.576	0.103	0.569	0.176	0.793	0.243	0.403	0.166
	PFS+BS	0.753	0.221	0.563	0.192	0.573	0.108	0.563	0.164	0.789	0.249	0.398	0.172
AdaBoost	SKB+GS	0.746	0.234	0.531	0.199	0.700	0.179	0.604	0.168	0.808	0.284	0.422	0.153
	PFS+BS	0.741	0.226	0.529	0.201	0.689	0.166	0.598	0.174	0.800	0.278	0.415	0.149
Extra Trees	SKB+GS	0.765	0.230	0.583	0.167	0.538	0.151	0.559	0.152	0.787	0.303	0.423	0.137
	PFS+BS	0.759	0.212	0.580	0.153	0.531	0.145	0.557	0.165	0.771	0.295	0.419	0.141
Stacking	SKB+GS	0.769	0.213	0.577	0.115	0.626	0.172	0.612	0.177	0.809	0.263	0.438	0.151
	PFS+BS	0.752	0.220	0.573	0.121	0.618	0.180	0.592	0.181	0.801	0.258	0.428	0.146
Oracle	SKB+GS	0.909	0.024	0.992	0.003	0.854	0.018	0.676	0.054	0.869	0.045	0.625	0.033
	PFS+BS	0.902	0.032	0.990	0.005	0.849	0.022	0.658	0.063	0.862	0.048	0.618	0.039

Table 3 Summary of the results for classifiers in customer churn dataset

Classifiers	Feature Selection+HPO	Accuracy		Precision		Recall		F1 Score		ROC-AUC Score		Cohen's Kappa Score	
		Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv	Mean	Stdv
Decision Tree	SKB+GS	0.499	0.183	0.498	0.321	0.504	0.207	0.501	0.154	0.499	0.232	0.000	0.000
	PFS+BS	0.493	0.176	0.497	0.334	0.502	0.219	0.498	0.160	0.493	0.239	0.000	0.000
Random Forest	SKB+GS	0.496	0.177	0.495	0.317	0.471	0.226	0.483	0.169	0.496	0.241	-0.006	-0.001
	PFS+BS	0.389	0.169	0.493	0.325	0.466	0.218	0.482	0.173	0.490	0.228	-0.008	-0.004
XGBoost	SKB+GS	0.502	0.154	0.501	0.299	0.484	0.188	0.493	0.176	0.502	0.192	0.005	0.006
	PFS+BS	0.494	0.159	0.499	0.304	0.478	0.194	0.488	0.165	0.498	0.200	0.007	0.003
AdaBoost	SKB+GS	0.499	0.143	0.497	0.301	0.399	0.174	0.443	0.159	0.499	0.212	-0.001	-0.001
	PFS+BS	0.495	0.144	0.494	0.310	0.391	0.171	0.440	0.153	0.491	0.218	-0.002	-0.003
Extra Trees	SKB+GS	0.500	0.159	0.498	0.328	0.474	0.213	0.486	0.164	0.500	0.235	0.000	0.000
	PFS+BS	0.492	0.147	0.492	0.331	0.469	0.222	0.481	0.157	0.494	0.242	0.001	0.001
Stacking	SKB+GS	0.503	0.124	0.515	0.278	0.091	0.152	0.155	0.126	0.502	0.186	0.005	0.002
	PFS+BS	0.498	0.131	0.511	0.289	0.088	0.162	0.154	0.131	0.493	0.190	0.003	0.005
Oracle	SKB+GS	0.907	0.020	0.991	0.009	0.545	0.036	0.703	0.027	0.772	0.054	0.654	-0.01
	PFS+BS	0.902	0.019	0.986	0.010	0.536	0.041	0.700	0.032	0.766	0.060	0.649	-0.01

Values between 0.4 and 0.6, found in XGBoost, AdaBoost, Extra Trees, and Stacking across both the transaction and telecommunication datasets, indicate "moderate" agreement. Values between 0.6 and 0.8, such as those for Oracle in all three datasets, are considered as "substantial" agreement.

We also utilized a confusion matrix and classification report to examine errors and interpret the outcomes of binary churn classification. The True Positive Rate (TPR) denotes the percentage of customers who remained with the service. In most cases, this proportion exceeds the True Negative Rate (TNR), which represents the percentage of customers who discontinued the service and switched to competitors. Positive Predictive Value (PPV) identifies the percentage of customers who haven't churned but have shown signs of disinterest or dissatisfaction at a specific time. These customers may opt to leave the service in the future. Service providers should implement new strategies to minimize churn because acquiring new customers is considerably more expensive than retaining existing ones. Negative Predictive Value (NPV) indicates the percentage of satisfied customers likely to remain loyal to the service over a certain period. They actively engage in purchasing products, receiving services, and conducting transactions, potentially becoming repeat customers. Tables 4 and 5, and 6 reveal that the Oracle meta-classifier outperforms other classifiers in terms of TPR, TNR, PPV, and NPV. False Positive Rate (FPR) represents the proportion of customers wrongly identified as churners. False Negative Rate (FNR) indicates the ratio of actual churners not correctly identified. False Discovery Rate (FDR) includes all clients except potential churners, while False Omission Rate (FOR) includes all clients except potential loyal and repeat customers. FPR, FNR, FDR, and FOR depict error ratios, with Tables 4 and 5, and 6 demonstrating that Oracle has lower error ratios compared to others. Accuracy (ACC) confirms whether all churners and non-churners are correctly classified. Oracle exhibits the highest accuracy rate among all mentioned algorithms. Additionally, the Matthews Correlation Coefficient (MCC) is a more reliable metric for assessing classifier competence in identifying observed and predicted binary classifications. Oracle achieves MCC rates of 73.29%, 54.20%, and 68.37% across transaction, telecommunication, and customer churn datasets, indicating superior performance compared to other classifiers by a significant margin.

7 Contributions and further directions

In this study, we introduced the Oracle meta-classifier to predict customer churn intention using transaction, telecommunication, and customer churn datasets. Oracle is a static and robust meta-classifier known for its high accuracy, especially when coupled with hyperparameter optimization and feature selection processes. We implemented Oracle within the "DESlib" library, known for its high test coverage. For hyperparameter extraction, we utilized GridSearchCV, recognized as a stable tool for tuning. We carefully selected the best parameters for establishing classifiers, employing precise GridSearchCV methodology. GridSearchCV stands out as the most effective tool for hyperparameter tuning, systematically exploring various parameter combinations to determine the optimal set. Its outputs are consistent across runs, ensuring reliability. The detailed and explicit outputs of GridSearchCV significantly enhance clas-

Table 4 Summary of the predictions for classifiers in transaction dataset

Classifiers	FS+HPO	TPR	TNR	PPV	NPV	FPR	FNR	FDR	FOR	ACC	MCC
Decision Tree	SKB+GS	0.850	0.671	0.958	0.333	0.328	0.149	0.041	0.666	0.836	0.390
	PFS+BS	0.844	0.668	0.953	0.329	0.332	0.154	0.053	0.673	0.824	0.384
Random Forest	SKB+GS	0.821	0.914	0.996	0.142	0.085	0.178	0.003	0.857	0.825	0.319
	PFS+BS	0.818	0.909	0.992	0.137	0.089	0.182	0.006	0.869	0.820	0.305
XGBoost	SKB+GS	0.873	0.808	0.973	0.439	0.191	0.126	0.026	0.560	0.865	0.530
	PFS+BS	0.869	0.796	0.970	0.434	0.197	0.131	0.033	0.568	0.864	0.527
AdaBoost	SKB+GS	0.880	0.719	0.951	0.490	0.280	0.119	0.048	0.509	0.8587	0.515
	PFS+BS	0.875	0.711	0.948	0.488	0.283	0.125	0.051	0.513	0.852	0.509
Extra Treess	SKB+GS	0.867	0.740	0.962	0.419	0.259	0.132	0.037	0.580	0.8533	0.482
	PFS+BS	0.858	0.737	0.956	0.413	0.264	0.139	0.040	0.597	0.848	0.479
Stacking	SKB+GS	0.882	0.765	0.962	0.490	0.234	0.117	0.038	0.509	0.868	0.541
	PFS+BS	0.881	0.760	0.957	0.486	0.240	0.123	0.044	0.511	0.858	0.536
Oracle	SKB+GS	0.907	0.991	0.998	0.598	0.008	0.092	0.001	0.401	0.917	0.732
	PFS+BS	0.902	0.987	0.990	0.591	0.009	0.096	0.002	0.437	0.901	0.728

Table 5 Summary of the predictions for classifiers in telecommunication dataset

Classifiers	FS+HPO	TPR	TNR	PPV	NPV	FPR	FNR	FDR	FOR	ACC	MCC
Decision Tree	SKB+GS	0.891	0.487	0.833	0.634	0.512	0.098	0.166	0.365	0.795	0.287
	PFS+BS	0.888	0.481	0.828	0.627	0.518	0.099	0.169	0.371	0.788	0.278
Random Forest	SKB+GS	0.895	0.479	0.831	0.640	0.520	0.095	0.168	0.360	0.795	0.390
	PFS+BS	0.893	0.472	0.830	0.439	0.522	0.097	0.172	0.369	0.788	0.389
XGBoost	SKB+GS	0.896	0.534	0.841	0.647	0.447	0.094	0.152	0.359	0.759	0.403
	PFS+BS	0.891	0.529	0.839	0.643	0.453	0.095	0.160	0.368	0.754	0.339
AdaBoost	SKB+GS	0.893	0.564	0.853	0.650	0.436	0.096	0.146	0.349	0.749	0.431
	PFS+BS	0.889	0.558	0.847	0.648	0.441	0.098	0.155	0.356	0.743	0.429
Extra Treess	SKB+GS	0.894	0.575	0.855	0.657	0.440	0.095	0.153	0.358	0.765	0.400
	PFS+BS	0.890	0.566	0.849	0.651	0.446	0.096	0.158	0.362	0.766	0,396
Stacking	SKB+GS	0.898	0.600	0.873	0.670	0.145	0.079	0.141	0.299	0.768	0.439
	PFS+BS	0.892	0.596	0.871	0.666	0.148	0.082	0.146	0.301	0.754	0.424
Oracle	SKB+GS	0.900	0.914	0.919	0.895	0.025	0.089	0.080	0.105	0.909	0.542
	PFS+BS	0.900	0.906	0.915	0.892	0.028	0.090	0.081	0.107	0.903	0.533

sifier accuracy. To forecast customer churn intention, we employed Decision Tree, Random Forest, XGboost, AdaBoost, and Extra Trees classifiers. We also utilized the SelectKBest feature selection method with the f_classif feature selection approach to enhance classifier proficiency. StratifiedKFold was used for cross-validation to mitigate overfitting. Combining various classifiers within a meta-classifier framework yields the highest prediction accuracy. To bolster the superiority of Oracle, we conducted a series of experiments employing advanced techniques such as Permutation feature selection in conjunction with BayesSearchCV hyperparameters Bayesian optimization. The outcomes of these experiments unequivocally affirm the superiority of Oracle when compared to other classifiers as well. Additionally, we compared Oracle with the Stacking ensemble meta-classifier to demonstrate Oracle's robustness. Results indicate that Oracle outperforms individual classifiers and even

Table 6 Summary of the predictions for classifiers in customer churn dataset

Classifiers	FS+HPO	TPR	TNR	PPV	NPV	FPR	FNR	FDR	FOR	ACC	MCC
Decision Tree	SKB+GS	0.501	0.499	0.891	0.108	0.500	0.498	0.108	0.891	0.501	0.000
	PFS+BS	0.495	0.498	0.887	0.105	0.501	0.502	0.111	0.900	0.499	0.001
Random Forest	SKB+GS	0.498	0.495	0.522	0.471	0.504	0.501	0.477	0.528	0.497	0.006
	PFS+BS	0.493	0.492	0.502	0.470	0.506	0.504	0.481	0.534	0.400	0.008
XGBoost	SKB+GS	0.504	0.501	0.521	0.484	0.498	0.496	0.478	0.515	0.503	0.005
	PFS+BS	0.497	0.500	0.518	0.482	0.503	0.508	0.483	0.523	0.486	0.006
AdaBoost	SKB+GS	0.500	0.497	0.598	0.399	0.502	0.499	0.401	0.600	0.499	0.001
	PFS+BS	0.491	0.494	0.593	0.397	0.507	0.504	0.402	0.602	0.495	0.003
Extra Treess	SKB+GS	0.499	0.496	0.523	0.472	0.503	0.500	0.476	0.527	0.498	0.003
	PFS+BS	0.493	0.491	0.517	0.466	0.505	0.500	0.478	0.540	0.492	0.006
Stacking	SKB+GS	0.502	0.515	0.814	0.091	0.484	0.497	0.085	0.908	0.503	0.010
	PFS+BS	0.501	0.512	0.809	0.089	0.894	0.503	0.087	0.909	0.501	0.011
Oracle	SKB+GS	0.819	0.865	0.875	0.806	0.134	0.180	0.124	0.193	0.841	0.683
	PFS+BS	0.818	0.857	0.871	0.802	0.138	0.182	0.127	0.198	0.839	0.688

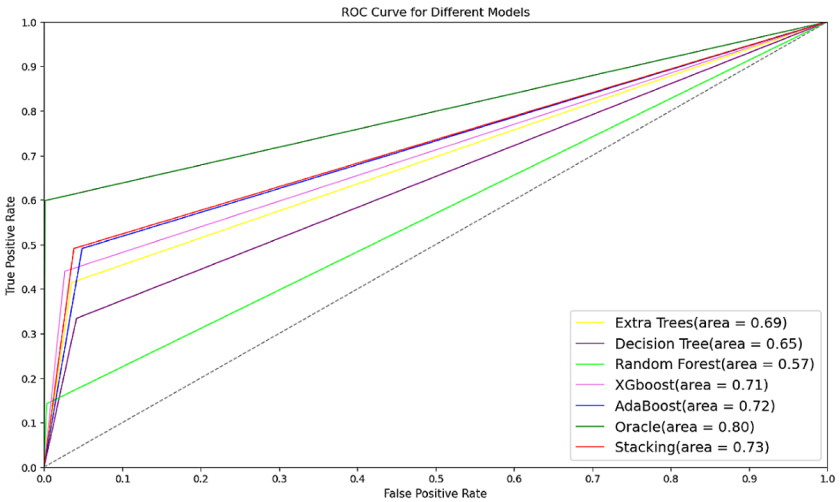

Fig. 5 The ROC Curve of the selected classifiers using the transaction dataset

the Stacking ensemble in predicting customer churn rates accurately. For example, as shown in Fig. 5, the superiority of the Oracle meta-classifier over the other selected classifiers is evident. The area under the ROC curve for Oracle is larger compared to the other classifiers, indicating that Oracle outperforms them in customer churn prediction. Oracle demonstrates superior accuracy and precision in predicting customer churn behavior compared to both individual classifiers and meta-classifiers like Stacking. Specialists have the option to employ more sophisticated gradient boosting methods on Decision Trees, such as LightGBM and Catboost, to enhance the efficiency and effectiveness of churn prediction. Additionally, they can utilize techniques like Lasso CV or Ridge Classifier CV for cross-validation. Furthermore,

experts can leverage the Oracle meta-classifier not only for feature selection but also for addressing various classification challenges, including confidence estimation and identifying missing features. The versatility of the Oracle algorithm extends beyond churn prediction; it can be applied in decision-making contexts such as conversion rate prediction and business intelligence. Given its precision and adaptability, Oracle holds potential for practical applications in diverse prediction domains, including e-marketing, e-retailing, the stock market, and even healthcare.

8 Managerial implication

Given the rapidly evolving nature and expanding scope of the internet services industry, there has been a significant increase in the discernible shifts in customers' expectations, desires, and decision-making processes. This behavior is primarily driven by the widespread accessibility of diverse products, services, and information. Within this dynamic environment, the bargaining power of customers and their ability to switch between services with just a few clicks has continued to grow. Consequently, customers can swiftly transition from one service provider to another without encountering significant entry barriers. Simultaneously, the proliferation of service providers and the range of services they offer have increased over time. This escalation poses a threat from new market entrants, particularly as dissatisfied customers readily shift to competitors if their service expectations are not met. Consequently, the internet services industry has become highly competitive, marked by disruptions, threats, and intense competition fueled by innovative products, new entrants, and existing rivals. As the usage of internet services among customers continues to rise, datasets related to transactions, telecommunications, and customer churn hold valuable insights. These datasets comprise pertinent information about customers' demographic profiles and payment histories, facilitating the detection of customer behavior and consumption patterns. Given the importance of these datasets in predicting customer churn rates, service providers must meticulously observe and analyze customer interactions to mitigate churn, boost revenue, and maintain competitiveness. Analyzing and scrutinizing transactional, telecommunication, and customer churn datasets offer a more effective approach to predicting churn rates compared to other datasets. This is because these datasets reflect direct interactions with the service, akin to examining a shopping basket. In the realm of service-oriented businesses, customers represent the most valuable asset for every enterprise. Consequently, service providers must cultivate strong relationships with their customers by delivering tailored services based on their preferences to ensure satisfaction and retention. Predicting customer churn rates enables firms to devise practical strategies aimed at retaining customers. For instance, Total Positive Rate (TPR) identifies the percentage of customers who have not churned, suggesting the importance of maintaining their loyalty by offering superior and expedited services. Total Negative Rate (TNR) highlights the rate of customers who have churned, indicating the necessity of reaching out to them to understand their reasons for leaving and resolving any issues to encourage their return. Positive Predictive Value (PPV) denotes the proportion of customers who have not churned but are dissatisfied with the services. Offering incentives such as

free services or discounts may prevent their departure. Negative Predictive Value (NPV) represents the percentage of satisfied and loyal customers, who can be incentivized to remain through rewards or bonuses for continued service usage. Additionally, involving them in feedback sessions and encouraging them to share suggestions for service improvement can enhance customer satisfaction and loyalty.

9 Discussion and conclusion

Electronic service platforms offer a wide array of purchase options, giving customers the ability to easily switch to competitors if unsatisfied. This increases customer bargaining power, pressuring firms to prioritize customer satisfaction and loyalty. Many businesses are shifting from product-centric to customer-centric approaches due to intense competition. Customer retention strategies, like Customer-Churn Prediction (CCP), are crucial in this environment. Retaining existing customers is prioritized as acquiring new ones is more costly. Predicting customer behavior and buying patterns offers competitive advantages, such as identifying bestselling products and understanding customer churn. Businesses utilize classifiers to recognize customer needs and consumption patterns, aiming to satisfy and retain current customers while attracting new ones. This study focused on forecasting customer churn intentions through the implementation of an innovative and robust meta-classifier. To accomplish this, we analyzed three distinct datasets: transaction records, telecommunications data, and customer churn information. Using a combination of Decision Tree, Random Forest, XGBoost, AdaBoost, and Extra Trees as the primary supervised classifiers across these datasets, we conducted separate cross-validation and evaluation procedures. Additionally, we employed SelectKBest and permatuation feature selection to identify the most effective features for achieving optimal accuracy. Hyperparameter optimization was carried out using GridSearchCV and BayesSearchCV. Subsequently, we integrated the refined classifiers into a new meta-classifier tailored to each dataset. The experimental findings highlight the superior accuracy of our proposed meta-classifier compared to traditional classifiers and even stacking ensemble methods. These predictive insights provide valuable assistance to businesses in identifying potential churners and implementing proactive strategies to retain customers, thereby bolstering customer retention rates and ensuring long-term business viability.

Author contributions The authors declare that all listed authors have approved the manuscript before submission, including the names and order of authors. All authors are responsible for correctness of the statements provided in the manuscript.

Funding The authors did not receive support from any organization for the submitted work.

Declarations

Ethical approval The authors declare that this study does not involve human or animal participants.

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Achal FT, Ahmmed MS, Aurpa TT (2023), April Severity Detection of Problematic Smartphone Usage (PSU) and its Effect on Human Lifestyle using Machine Learning. In 2023 IEEE 8th International Conference for Convergence in Technology (I2CT) (pp. 1–6). IEEE

Ahn Y, Kim D, Lee DJ (2019) Customer attrition analysis in the securties industry: a large-scale field study in Korea. Int J Bank Mark 38(3):561–577

Alexandropoulos SAN, Aridas CK, Kotsiantis SB, Vrahatis MN (2019) Stacking strong ensembles of classifiers. In Artificial Intelligence Applications and Innovations: 15th IFIP WG 12.5 International Conference, AIAI 2019, Hersonissos, Crete, Greece, May 24–26, 2019, Proceedings 15 (pp. 545–556). Springer International Publishing

Altmann A, Toloşi L, Sander O, Lengauer T (2010) Permutation importance: a corrected feature importance measure. Bioinformatics 26(10):1340–1347

Anshari M, Almunawar MN, Lim SA, Al-Mudimigh A (2019) Customer relationship management and big data enabled: personalization & customization of services. Appl Comput Inf 15(2):94–101

Asthana P (2018) A comparison of machine learning techniques for customer churn prediction. Int J Pure Appl Math 119(10):1149–1169

Bachmann P, Meierer M, Näf J (2021) The role of time-varying contextual factors in latent attrition models for customer base analysis. Mark Sci 40(4):783–809

Calderón-Monge E, Ramírez-Hurtado JM (2022) Measuring the consumer engagement related to social media: the case of franchising. Electron Commer Res 22(4):1249–1274

Calvano E, Polo M (2021) Market power, competition and innovation in digital markets: a survey. Inf Econ Policy 54:100853

Çelik O, Osmanoglu UO (2019) Comparing to techniques used in customer churn analysis. J Multidisciplinary Developments 4(1):30–38

Claesen M, Simm J, Popovic D, Moor B (2014), September Hyperparameter tuning in python using optunity. In Proceedings of the international workshop on technical computing for machine learning and mathematical engineering (Vol. 1, No. 3)

Coussement K (2014) Improving customer retention management through cost-sensitive learning. European Journal of Marketing

Cruz RM, Sabourin R, Cavalcanti GD (2018) Dynamic classifier selection: recent advances and perspectives. Inform Fusion 41:195–216

Devriendt F, Berrevoets J, Verbeke W (2021) Why you should stop predicting customer churn and start using uplift models. Inf Sci 548:497–515

Dey R, Mathur R (2023), May Ensemble Learning Method Using Stacking with Base Learner, A Comparison. In International Conference on Data Analytics and Insights (pp. 159–169). Singapore: Springer Nature Singapore

Effrosynidis D, Arampatzis A (2021) An evaluation of feature selection methods for environmental data. Ecol Inf 61:101224

Ehsani F, Hosseini M (2023a) Investigation to determine elements influencing customer's satisfaction in the B2C electronic retailing marketplaces. EuroMed J Bus 18(3):321–344

Ehsani F, Hosseini M (2023b) Customer segmentation based on location and timing dimensions using Big Data from Business-to-customer retailing marketplaces. Big Data 11(5):1–16

Elmi AH, Abdullahi A, Barre MA (2024) A machine learning approach to cardiovascular disease prediction with advanced feature selection. Indonesian J Electr Eng Comput Sci 33(2):1–1x

Fisher A, Rudin C, Dominici F (2019) All models are wrong, but many are useful: learning a variable's importance by studying an entire class of prediction models simultaneously. J Mach Learn Res 20(177):1–81

Hand DJ (2018) Statistical challenges of administrative and transaction data. J Royal Stat Soc Ser A: Stat Soc 181(3):555–605

Isa SIHS, Nayan SM (2020) WOW your customers: Tips to retain customers. J Undergrad Social Sci Technol, 2(2)

Jan ZM, Verma B (2020) Multiple elimination of base classifiers in ensemble learning using accuracy and diversity comparisons. ACM Trans Intell Syst Technol (TIST) 11(6):1–17

Johny CP, Mathai PP (2017) Customer churn prediction: a survey. Int J Adv Res Comput Sci, 8(5)

Karthikeyan S, Kathirvalavakumar T, Prasath R (2023), June Classification of the Class Imbalanced Data Using Mahalanobis Distance with Feature Filtering. In International Conference on Mining Intelligence and Knowledge Exploration (pp. 45–53). Cham: Springer Nature Switzerland

Lalwani P, Mishra MK, Chadha JS, Sethi P (2022) Customer churn prediction system: a machine learning approach. Computing 104:1–24

Lee HK, Kim SB (2018) An overlap-sensitive margin classifier for imbalanced and overlapping data. Expert Syst Appl 98:72–83

Liu W, Fan H, Xia M, Xia M (2022) A focal-aware cost-sensitive boosted tree for imbalanced credit scoring. Expert Syst Appl 208:118158

Loo BP, Ngan YL (2012) Developing mobile telecommunications to narrow digital divide in developing countries? Some lessons from China. Telecomm Policy 36(10–11):888–900

Mahajan D, Gangwar R (2017) Improved customer churn Behaviour by using SVM. Int J Eng Technol, 2395–0072

Mallett S, Halligan S, Collins GS, Altman DG (2014) Exploration of analysis methods for diagnostic imaging tests: problems with ROC AUC and confidence scores in CT colonography. PLoS ONE, 9(10), e107633

Mehralian MM (2022) Identifying and Explaining the Effective Factors of Digital Marketing Strategies in Consumers' Emotional States and Sales Rates: A Mixed Methods Research. In 20th International Conference of the Business and Strategic Management

Mkansi M (2022) E-business adoption costs and strategies for retail micro businesses. Electron Commer Res 22(4):1153–1193

Pan H, Zhou H (2020) Study on convolutional neural network and its application in data mining and sales forecasting for E-commerce. Electron Commer Res 20(2):297–320

Rane NL, Achari A, Choudhary SP (2023) Enhancing customer loyalty through quality of service: effective strategies to improve customer satisfaction, experience, relationship, and engagement. Int Res J Modernization Eng Technol Sci 5(5):427–452

Rao C, Xu Y, Xiao X, Hu F, Goh M (2024) Imbalanced customer churn classification using a new multi-strategy collaborative processing method. Expert Syst Appl 247:123251

Santos MS, Abreu PH, Japkowicz N, Fernández A, Soares C, Wilk S, Santos J (2022) On the joint-effect of class imbalance and overlap: a critical review. Artif Intell Rev 55(8):6207–6275

Seturi M (2024) Exploring the importance of building strong customer relationships. Technol Audit Prod Reserves, 1(4 (75))

Sharaff A, Gupta H (2019) Extra-tree classifier with metaheuristics approach for email classification. In Advances in Computer Communication and Computational Sciences: Proceedings of IC4S 2018 (pp. 189–197). Springer Singapore

Shuai Y, Zheng Y, Huang H (2018), November Hybrid software obsolescence evaluation model based on PCA-SVM-GridSearchCV. In 2018 IEEE 9th international conference on software engineering and service science (ICSESS) (pp. 449–453). IEEE

Tsai C-F, Chen M-Y (2010) Variable selection by association rules for customer churn prediction of multimedia on demand. Expert Syst Appl 37(3):2006–2015

Vafeiadis T, Diamantaras KI, Sarigiannidis G, Chatzisavvas KC (2015) A comparison of machine learning techniques for customer churn prediction. Simul Model Pract Theory 55:1–9

Varadarajan R, Welden RB, Arunachalam S, Haenlein M, Gupta S (2022) Digital product innovations for the greater good and digital marketing innovations in communications and channels: evolution, emerging issues, and future research directions. Int J Res Mark 39(2):482–501

Verbeke W, Dejaeger K, Martens D, Hur J, Baesens B (2012) New insights into churn prediction in the telecommunication sector: a profit driven data mining approach. Eur J Oper Res 218(1):211–229

Vetter TR, Schober P (2018) Agreement analysis: what he said, she said versus you said. Anesth Analgesia 126(6):2123–2128

Victoria AH, Maragatham G (2021) Automatic tuning of hyperparameters using bayesian optimization. Evol Syst 12(1):217–223

Vijaya J, Sivasankar E (2018) Computing efficient features using rough set theory combined with ensemble classification techniques to improve the customer churn prediction in telecommunication sector. Computing 100:839–860

Vuttipittayamongkol P, Elyan E, Petrovski A (2021) On the class overlap problem in imbalanced data classification. Knowl Based Syst 212:106631

Wang Z, Zhao S, Li Z, Chen H, Li C, Shen Y (2021) Ensemble selection with joint spectral clustering and structural sparsity. Pattern Recogn 119:108061

Wu X, Liao H, Tang M (2023) Decision making towards large-scale alternatives from multiple online platforms by a multivariate time-series-based method. Expert Syst Appl 212:118838
Wyner AJ, Olson M, Bleich J, Mease D (2017) Explaining the success of adaboost and random forests as interpolating classifiers. J Mach Learn Res 18(48):1–33
Zhao Y, Zhang W, Liu X (2024) Grid search with a weighted error function: hyper-parameter optimization for financial time series forecasting. Appl Soft Comput, 111362

Authors and Affiliations

Fatemeh Ehsani[1] · **Monireh Hosseini**[1]

✉ Monireh Hosseini
hosseini@kntu.ac.ir

Fatemeh Ehsani
ehsani@email.kntu.ac.ir

[1] Department of Information Technology, Faculty of Industrial Engineering, K. N. Toosi University of Technology, Tehran, Iran