The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 2024

Volume 32, Pages 208-229

IConTES 2024: International Conference on Technology, Engineering and Science

Machine Learning Approaches for Bank Customer Churn Analysis

Yasemin Bozkurt
Kutahya Dumlupinar University

Hasan Temurtas
Kutahya Dumlupinar University

Cigdem Bakir
Kutahya Dumlupinar University

Abstract: With the development of technology, rapid developments are experienced in the banking sector, as in many other sectors. However, with these developments, many problems arise, especially customer losses. Customer losses are a common problem not only in banks but also in many institutions such as insurance, production and logistics, etc. Customer churn is when people performing banking activities begin to abandon these services. It also causes financial and prestige loss for banks. It is possible to minimize or eliminate this problem by providing good service to customers. However, in order to achieve this, it is necessary to accurately analyze the trends that customers care about when providing banking services and the characteristics that cause customer loss. For these reasons, this study aims to estimate the probability of a bank customer leaving the bank from which he receives service. In this study, a data set containing customer data of a bank was used. The data set has various features that include customers' demographic information, financial situations, and interactions with the bank. Using common machine learning algorithms such as Logistic Regression, Decision Trees, Support Vector Machines (SVM), Gradient Boosting, K-Nearest Neighbors (KNN) and Multi-Layer Perceptron (MLP), the reasons for bank customers leaving were determined and the results were evaluated. Cross-validation method was used to evaluate the performance of each proposed machine learning algorithm. Additionally, hyperparameter optimization was used to increase the classification accuracy of the proposed methods. The best parameters were determined with the GridSearchCV hyperparameter optimization method. The performances of the proposed machine learning algorithms are compared. When the findings were evaluated, it was observed that the Gradient Boosting method achieved the highest accuracy rate.

Keywords: Machine learning, Gradient boosting, Gridsearchcv, Hyperparameter optimization

Introduction

In today's competitive banking environment, customer loyalty and customer relationship management are critical for financial institutions. Customer churn is a major concern for banks, and financial institutions are constantly developing strategies to understand and predict customer behavior in order to retain and grow their customer base (Amuda&Adeyemo, 2019). In this context, machine learning and data analytics techniques offer powerful tools for predicting and managing bank customer churn.

Imani and Arabnia conducted a study to predict customer losses in the telecommunications sector using SVM, RF (Random Forest), Decision Trees, ANN (Artificial Neural Networks), LightGBM, XGBoost and CatBoost methods (Imani&Arabnia, 2023). SMOTE technique was used for the unbalanced class problem and the success of the models used with different evaluation criteria such as F1, precision and recall was compared. When ROC analysis was performed, both XGBoost and CatBoost were more successful than other methods. As a result of

Optuna hyperparameter optimization, CatBoost achieved 93% success. The success of the proposed models is increased with hybrid data sampling techniques. Approximately 91% F1 Score was obtained with the LightGBM gradient boosting technique. However, on larger and unbalanced data sets, the system can be further tested for success with deep learning methods such as Long Short Term Memory (LSTM). Additionally, the negative effects of missing predictions in the data on classification success can be reduced.

It is very risky for businesses to leave customers from the businesses they receive service from for different reasons. Customers who are not satisfied with the service do not recommend these businesses to other users. These problems cause great financial losses and loss of prestige of businesses. Determining customer losses is very important for businesses to minimize these problems. Khattak et al., have carried out studies to detect customer churn using combined deep learning models (BiLSTM-CNN) (Khattak etc., 2023). In this study, it is aimed to increase the accuracy in the detection process of customer losses with the combined deep learning model they proposed. Their proposed model provided approximately 81% accuracy, 66% precision, 65% F1-Score, and 64% recall. However, a two-class data set was used in this study. Analysis is needed for multi-class data sets. Additionally, studies can be carried out to select features and determine feature types.

Liu et al. (2024) used the Extreme Gradient Boosting Tree (XGBT) method to retain customers and determine customer churn in businesses. Hyperparameters and weights of XGBT trees are optimized with Bayes theorem. The XGBT method performed with Bayesian optimization is called BO-XGBT. By adjusting the class weights through the study conducted on real datasets, the computational cost was greatly reduced and the classification accuracy was increased. Additionally, the working time is more optimum than other methods. Thus, the problem of misclassification is reduced. In the study, the relationship between customer tendencies and the cost of retaining customers can be determined on different data sets. Therefore, it can be used in different areas such as credit fraud and fraud detection, apart from modeling customer loss prediction. Jajam et al. (2023) Arithmetic Optimization Algorithm (AOA) and Ensemble Deep Learning SBLSTM (Stacked Bidirectional Long Short-term Memory) –RNN (Rcurrent Neural Network)-IGSA (Improved Gravitational Search Optimization Algorithm) for customer churn prediction (CCP) and customer satisfaction. They proposed the model. The classification accuracy was increased by eliminating the IGSA optimization developed by hyperparameter tuning with the proposed AOA-SBLSTM-RNN model, and the results of both methods were compared. Three different health data were used to measure the success of these two proposed models. The results of the AOA-SBLSTM-RNN deep learning model were analyzed with different evaluation criteria and achieved 97.67% and 97.89% success in the 1st and 2nd dataset, respectively. In this study, customer turnover was estimated efficiently and effectively. Further analysis can be done with time series methods. Xiahou and Harada (2022) have developed methods based on K-Means and SVM method to determine e-commerce customer losses of B2C E-commerce organization, because in e-commerce organizations, preventing customer losses and retaining customers is very important, especially in determining and implementing marketing strategies. For this reason, in this study, the temporal characteristics of customers' e-commerce shopping were modeled with k-means and SVM. Additionally, the proposed models were compared with logistic regression. However, expanding the customer segmentation applied in this study can also strengthen customer relationships.

The use of artificial intelligence and machine learning methods in CCP (Customer Churn Prediction) systems has increased the performance of businesses. Faritha et al. (2022). proposed an artificial intelligence-based CCP model for early prediction of customer churn in the telecommunications industry.The model called AICCP (AI-based CCP)-TBM (Telecommunication Business Markets) identifies customers in the telecom sector. It aims to prevent customer losses in the Telecom sector by identifying subscribed and non-subscribed customers. In addition, the AICCP-TBM model Chaotic Salp Swarm Optimization-based Feature Selection (CSSO-FS) determined the best features in three datasets. The AICCP-TBM model they applied on these datasets achieved 97.25%, 97.5% and 94.33% success, respectively. The proposed model can be used for CCP systems in real-time cloud data on large datasets.

Sana et al. (2022) developed a model based on machine learning techniques (Naïve Bayes, Random forest, Gradient boosting, KNN, Feed Forward Neural Networks, Logistic Regression, Recurrent Neural Networks) for CCP to identify customers who are likely to leave in the telecommunications industry. For this model, they used various data transformation (Box-Cox, Log, Z-Score, Rank) and feature selection methods on CRM (Customer Relationship Management) data. Grid search method was applied to determine and optimize the most optimum and best hyperparameters through univariate feature selection. Research was conducted on the TCI data set with sensitivity, recall, AUC, F-Score evaluation criteria. CCP prediction was increased by 26.2% and 17% in terms of AUC and F-Score, respectively. The proposed model can be tested on other telecommunications data and larger data sets.

In our study, the use of various 6 different machine learning methods on a bank customer churn data set is examined. Our aim is to identify the factors affecting bank customer churn and to determine the most effective model to predict customer churn using these factors. In our study, hyperparameter optimization was used to increase the success of all machine learning methods. By performing GridSearchCV optimization hyperparameter fine-tuning, the classification success, i.e. accuracy, of all proposed machine learning methods has been increased. The best parameter values for all methods proposed in this study were determined and presented in the application section. The accuracy of the proposed models with different evaluation criteria was analyzed and the success of the proposed models was compared. This study can help banks improve their strategies for managing customer relationships and allow for more effective measures to protect their customer base.

Method

In this section, all methods used in the study are shown.

Logistic Regression

Logistic Regression is a widely used machine learning technique for classification problems. Its main purpose is to estimate the probability of the dependent variable (outcome) based on the independent variables (predictor). It is often used in binary classification problems, meaning that the outcome can be split between two categories (such as pass or fail). Logistic regression transforms a linear combination of input features into a probability value using the sigmoid function (De Caigny et al., 2018). This probability value expresses the probability of belonging to a particular class. The training process occurs by comparing the probabilities predicted by the model with the observed labels. The model uses the maximum likelihood method to learn the parameters that best fit the data (Table 1) (Jain et al., 2020).

Table 1. Logistic regression advantages and dezavantages

Advantages	**Disadvantages**
Simplicity: The simplicity of the model makes it easy to implement and interpret.	Sensitivity to outliers: Logistic regression may be less flexible in tolerating outliers than nonlinear models.
Good performance: Linear models can work effectively even on large data sets.	Multi-class classification: Logistic regression focuses on binary classification by default and cannot be directly extended for multi-class classification problems.
Good interpretability: Coefficients help us understand the impact on each feature.	Assumption of linear relationship between independent variables: Logistic regression is based on the assumption of a linear relationship between independent variables. If such a relationship does not exist, the model may give misleading results.

Decision Tree

Table 2. Decision tree advantages and dezavantages

Advantages	**Disadvantages**
Understandability: Decision trees can be easily interpreted by users because the rules they create are clear and understandable.	Tendency to Overfit: They tend to fit too much to the data set, which can trigger overfitting and decreased generalization performance.
Easy Applicability: They do not require complex data preprocessing and can easily work with categorical/numeric data.	Sensitivity: They can be sensitive to small data changes, which can affect the stability of the model.
Determining Variable Importance Level: They can be used to evaluate the importance of each attribute in classification.	Complexity: As the dataset complexity increases, the constructed tree becomes more complex, which can reduce the understandability of the model.

Decision trees are a machine learning technique that identifies patterns in data sets using a tree structure to solve a classification or regression problem (Nie et al., 2011). Essentially, a decision tree creates a tree structure that represents a set of decision rules. Each node tests a specific range of values of an attribute and is divided into different branches based on this test result. This process is used to assign data points to a particular class or estimate a value (Table 2) (Hoppner et al., 2020).

Support Vector Machines (SVM)

SVM is a powerful and popular classification method used in machine learning and data mining (Zhao et al., 2005). This method sets decision boundaries to classify data and tries to minimize classification error. SVM is a technique that is generally effective on high-dimensional datasets and is used to solve nonlinear classification problems. SVM is a supervised learning algorithm that finds the best separation line (hyperplane) to separate data into two classes. SVM uses support vectors to find this hyperplane. This hyperplane performs the classification process by maximizing the margin between two classes. Support vectors are the data points closest to the classification boundary, and these points play a critical role in the classification decision. While SVM directly determines a hyperplane for linear separable data, for non-linear separable data it transforms the data into a higher dimensional space using kernel functions and makes distinctions there (Table 3) (Xia &J in, 2008).

Table 3. SVM advantages and dezavantages

Advantages	**Disadvantages**
High Accuracy: SVM offers high accuracy rates, especially on large and complex data sets. By maximizing the margin between classes, the model better separates the data and minimizes classification errors.	Computational Cost: SVM has high computational cost when working on large datasets and high-dimensional data. Training time can be long, which can be a limiting factor in big data applications.
Generalization Ability: SVM is resistant to overfitting problems. Thanks to the margin maximization strategy, the model can also show high performance on new and unprecedented data.	Hyperparameter Tuning: The performance of SVM depends on careful tuning of various hyperparameters. This can make the process of optimizing the model complex and time-consuming.
Flexibility: Thanks to kernel functions, SVM can solve both linear and non-linear classification problems. This makes it possible for the model to adapt to different types of data sets.	Significance: SVM can be difficult to explain how the model works and how the decision boundaries are determined. This can reduce the transparency and understandability of the model.

Gradient Boosting Classifier

Table 4. Gradient boosting classifier advantages and dezavantages

Advantages	**Dezavantages**
High Accuracy: Gradient Boosting often achieves high accuracy rates because it corrects errors at each step and improves the overall performance of the model.	Precise Parameter Tuning: Careful tuning of hyperparameters is necessary to optimize the performance of the model. Incorrect adjustments may reduce the performance of the model.
Flexibility: Can be used for both regression and classification problems.	Computational Cost: Gradient Boosting is computationally intensive and can be time consuming because it creates many trees.
Overfitting Control: Overfitting can be controlled thanks to parameter settings. In particular, hyperparameters such as learning rate and tree depth can be optimized to increase the generalization ability of the model.	Error Propagation: If the model makes large errors initially, these errors can propagate to subsequent trees, negatively impacting overall performance.
Feature Importance: Gradient Boosting is powerful in evaluating feature importance, which helps understand which features are more critical to the model.	Complexity: It is more complex than some other ensemble methods (e.g. Bagging) and requires more detailed knowledge of hyperparameter settings.

Gradient Boosting is a machine learning ensemble method that aims to create a strong learner by sequentially combining multiple weak learners (usually decision trees). This method gradually improves the model's performance by trying to correct errors at each step. Gradient Boosting in turn performs an optimization process on the outputs of tree-based models and repeats this process to reduce errors at each stage, increasing the accuracy of the final model (Table 4) (Raeisi&Sajedi, 2020). Gradient Boosting focuses on fixing errors step by step. The process can be summarized as follows (Gregory, 2018):

1. Creating the Initial Model: The process usually begins with the creation of a simple model (for example, a decision tree).
2. Calculation of Errors: Differences (residuals) between the predictions of the initial model and the actual values are calculated.
3. Training New Models: These residuals are estimated with a new decision tree. The new tree learns from these mistakes in the best possible way.
4. Model Update: The new tree is added to the existing model and the model's predictions are updated.
5. Iteration: Steps 2-4 are repeated for a certain number of iterations (or until errors are sufficiently reduced).

K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a simple but powerful classification and regression algorithm used in machine learning. Developed by Evelyn Fix and Joseph Hodges in 1951, KNN is known for working effectively especially on small and medium-sized data sets. KNN takes into account the labels or values of its nearest *k* neighbor to classify a particular data point or predict a value. Euclidean distance is usually used to determine neighbors, but other distance measurements can also be used (Table 5). The KNN algorithm works by storing training data in memory and refers to this training data when classifying or predicting a new data point. Its operation can be summarized as follows (Sjarif et al., 2019):

1. Parameter Selection: kkk value is determined. This indicates how many neighbors will be taken into account.
2. Distance Calculation: Distances between the new data point and all points in the training data set are calculated.
3. Determining Neighbors: The closest neighbor is selected.
4. Classification/Prediction:

- Classification: The class of the new data point is determined by looking at which neighbors' labels are most common.
- Regression: The value of the new data point is estimated by averaging the values of the neighbors.

Table 5. KNN classifier advantages and dezavantages

Advantages	**Dezavantages**
Simplicity: The KNN algorithm is quite simple and understandable. It does not require complex mathematical modeling.	Memory Usage: Memory usage can be a problem in large data sets as it needs to store the entire training data in memory.
No Training Period: There is no training phase; This is done by storing the training data in memory and therefore training time is negligible.	Computational Cost: In large data sets, classifying a new data point can take a lot of time. Since distance must be calculated with all training data, it is costly.
Adaptability: It can easily adapt to different data types and distributions.	Sensitivity to Records: Noisy and redundant data can negatively impact the performance of KNN.
Flexibility: It can be used in both classification and regression problems.	Attribute Scaling: Different scales of attributes in the data may affect distance calculations. Therefore, the data needs to be scaled beforehand.
Effective on Small Data Sets: KNN can provide very effective results on small and medium-sized data sets.	Complexity Parameter k: Choosing the appropriate k value is critical. While too small k value may cause overfitting, too large k value may lead to underfitting.

Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is the simplest and most widely used type of artificial neural networks, one of the basic building blocks of deep learning (Ismail et al., 2015). MLP is a feed-forward neural network model that contains multiple layers and multiple neurons (perceptrons) in each layer. MLP is an artificial neural network consisting of an input layer, one or more hidden layers, and an output layer. Each neuron takes inputs from the previous layer, weights those inputs, and calculates its output through an activation function. This process ensures the network's ability to learn and ensures the flow of information from inputs to outputs.

- Input Layer: This is the first layer that receives input data. Each neuron corresponds to an input feature.
- Hidden Layers: One or more layers located between the input layer and the output layer. These layers enable the network to learn complex relationships.
- Output Layer: It is the last layer and produces the final predictions or classifications of the network.

The working principle of MLP consists of two main steps: feedforward and backpropagation (Tang et al., 2020):

1. Feedforward: Input data is passed through the layers of the network and each neuron produces an output by weighting the inputs it receives and applying an activation function. This output is given as input to the next layer and the process continues until it reaches the output layer.
2. Backpropagation: The error between the model's predictions and the actual values is calculated. This error is propagated backwards to adjust the weights of the network. The goal is to minimize the error function, and this is accomplished using optimization algorithms such as gradient descent.

Commonly used activation functions in MLP are:

- Sigmoid: Compresses the output between 0 and 1, especially used in the output layer.
- ReLU (Rectified Linear Unit): If the input is positive, it leaves it as is, if it is negative, it resets it. It is widely used in deep networks.
- Tanh: Compresses the output between -1 and 1, reducing the gradient vanishing problem.

Multi-Layer Perceptron (MLP) is one of the cornerstones of artificial neural networks and forms the basis of deep learning models. It is capable of learning and modeling complex data relationships, thanks to feedforward and backpropagation processes between input, hidden and output layers. However, it has disadvantages such as high computational costs and hyperparameter tuning difficulties. Nevertheless, it has been used successfully in a wide range of applications due to its flexibility and generalization capacity.

Table 6. MLP classifier advantages and dezavantages

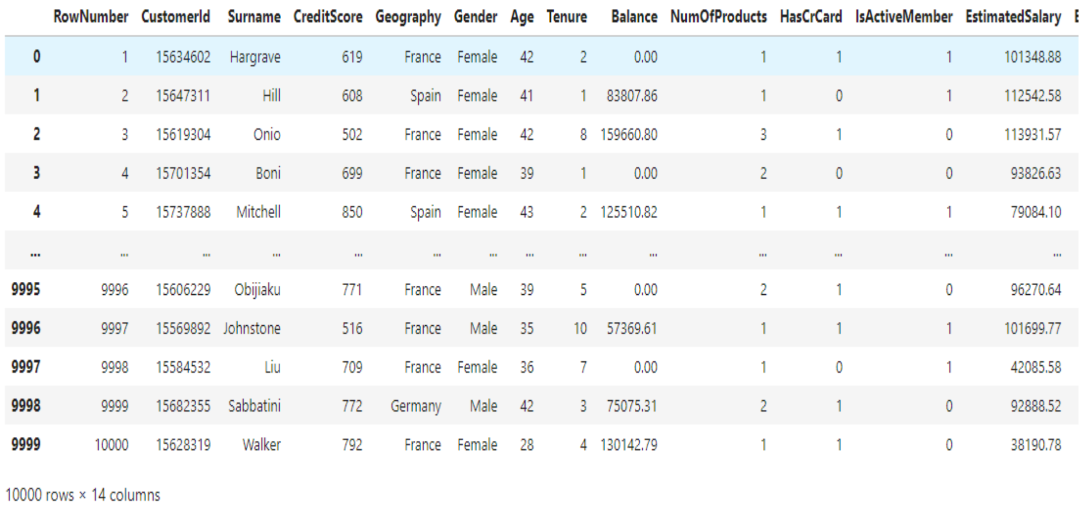
Advantages	**Dezavantages**
Flexibility and Power: MLP has the ability to learn and model complex data sets. In particular, its multi-layered structure can capture complex relationships in data sets.	Computational Costs: Training MLPs can be quite time-consuming and computationally intensive, especially for large data sets and networks with large numbers of layers and neurons.
Generalization Capacity: A sufficiently large and deep MLP can generalize data effectively and perform well on previously unseen data points.	Hyperparameter Tuning: The performance of MLPs depends on many hyperparameters such as number of layers, number of neurons, learning rate, and activation functions. Optimizing these hyperparameters can be complex and time-consuming.
Universal Approximation: Theoretically, a sufficiently large MLP can approximately model any continuous function with arbitrary accuracy.	Overfitting: A very large network may lose its ability to generalize by overfitting the training data. To avoid this situation, regularization methods and cross-validation techniques are used.

Experimental Study

The data set used in this study represents the customer base of a bank in Figure 1 (https://www.kaggle.com/datasets/saurabhbadole/bank-customer-churn-prediction-dataset/data). The data set includes various characteristics of customers, such as demographics (age, gender, geographic location), financial status (credit score, balance, estimated income), and interactions with bank products. It contains a total of 10,000 customers and 14 attribute columns. The data set is as follows (Figure 1): The data set contains the following columns:

1. RowNumber: Shows the row number that specifies the row for customers in the data set.
2. CustomerId: Shows the id number of the customers registered in the data set.

3. Surname: Shows the surname of the customers registered in the bank.
4. CreditScore: A numerical value that represents the customer's credit history. A high credit score generally indicates financial stability and credit availability.



	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	E
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	0	96270.64	
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	1	101699.77	
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	1	42085.58	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.78	

10000 rows × 14 columns

Figure 1. Bank dataset analysis for customer churn prediction

5. Geography: The geographical region where the customer lives. This information can be used to analyze whether there are differences in bank abandonment rates or behavior of customers in different regions.
6. Gender: The gender of the customer. It can be used to examine the impact of gender on customer abandonment rates.
7. Age: A numerical value representing the customer's age. Age can be a significant factor in customer abandonment rates.
8. Tenure: A numerical value that expresses how long the customer has been in the bank. It can have a significant impact on customer loyalty.
9. Balance: The balance in the customer's bank account. This provides information about the customer's financial situation and may influence the abandonment decision.
10. NumOfProducts: Number of products purchased by the customer from the bank. It can indicate the customer's level of interaction with the bank.

11.HasCrCard: A binary variable indicating whether the customer has a credit card. This may reflect customer shopping habits and propensity to use bank products.

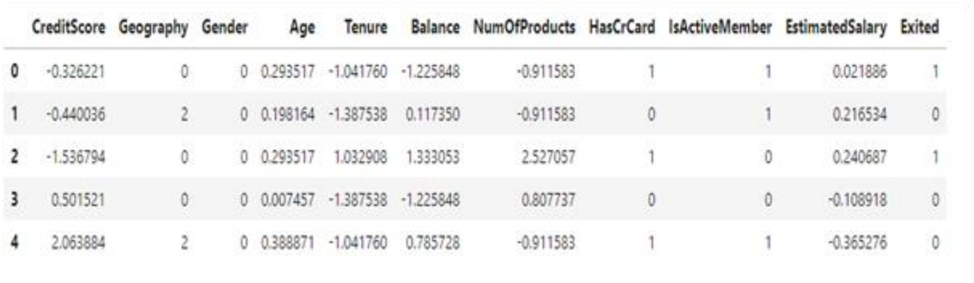
12 IsActiveMember: A binary variable indicating whether the customer is an active member. Active membership can indicate the customer's level of interaction with the bank.

13. EstimatedSalary: The customer's estimated annual salary. This provides additional information about the customer's financial situation.
14. Exited: A binary variable indicating whether the customer has left the bank. This represents the target variable in the data set and is the main focus to be analyzed.

The preprocessing steps performed on the data set are:

1. Examining and Filling in Missing Values: First, it was checked whether there were missing values in the data set. When missing values were detected, the missing values in the columns containing the missing values were filled in with an appropriate strategy. This is usually accomplished using the mean, median, or most frequent value.
2. Removal of Unnecessary Columns: Columns that were considered unnecessary for the analysis (such as Customer Identification Number and Customer Surname) were removed from the data set. These columns contain information that has no impact on the results of the analysis or is semantically unimportant.
3. Coding of Categorical Variables: Categorical variables were converted into numerical values for the modeling process. In this step, categorical variables such as geographic region and gender were converted to numerical values using methods such as label coding or one-hot coding.
4. Feature Scaling: Finally, numerical features were scaled to improve model performance. Scaling is usually done using Standardization or Normalization methods. In this way, the range of values between different features is balanced and the model works better.

These preprocessing steps make the dataset usable by machine learning models and increase the accuracy of the model. After the pre-processing steps, the final version of the data set is in Figure 2:



	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	-0.326221	0	0	0.293517	-1.041760	-1.225848	-0.911583	1	1	0.021886	1
1	-0.440036	2	0	0.198164	-1.387538	0.117350	-0.911583	0	1	0.216534	0
2	-1.536794	0	0	0.293517	1.032908	1.333053	2.527057	1	0	0.240687	1
3	0.501521	0	0	0.007457	-1.387538	-1.225848	0.807737	0	0	-0.108918	0
4	2.063884	2	0	0.388871	-1.041760	0.785728	-0.911583	1	1	-0.365276	0

Figure 2. Bank dataset pre-processing for customer churn prediction

Examining the distribution of data in the data set and the relationships between variables constituted one of the critical steps in the data analysis process. Understanding the distribution of the data helped to see the overall structure and characteristics of the data set, while understanding the relationships between variables guided the modeling process.

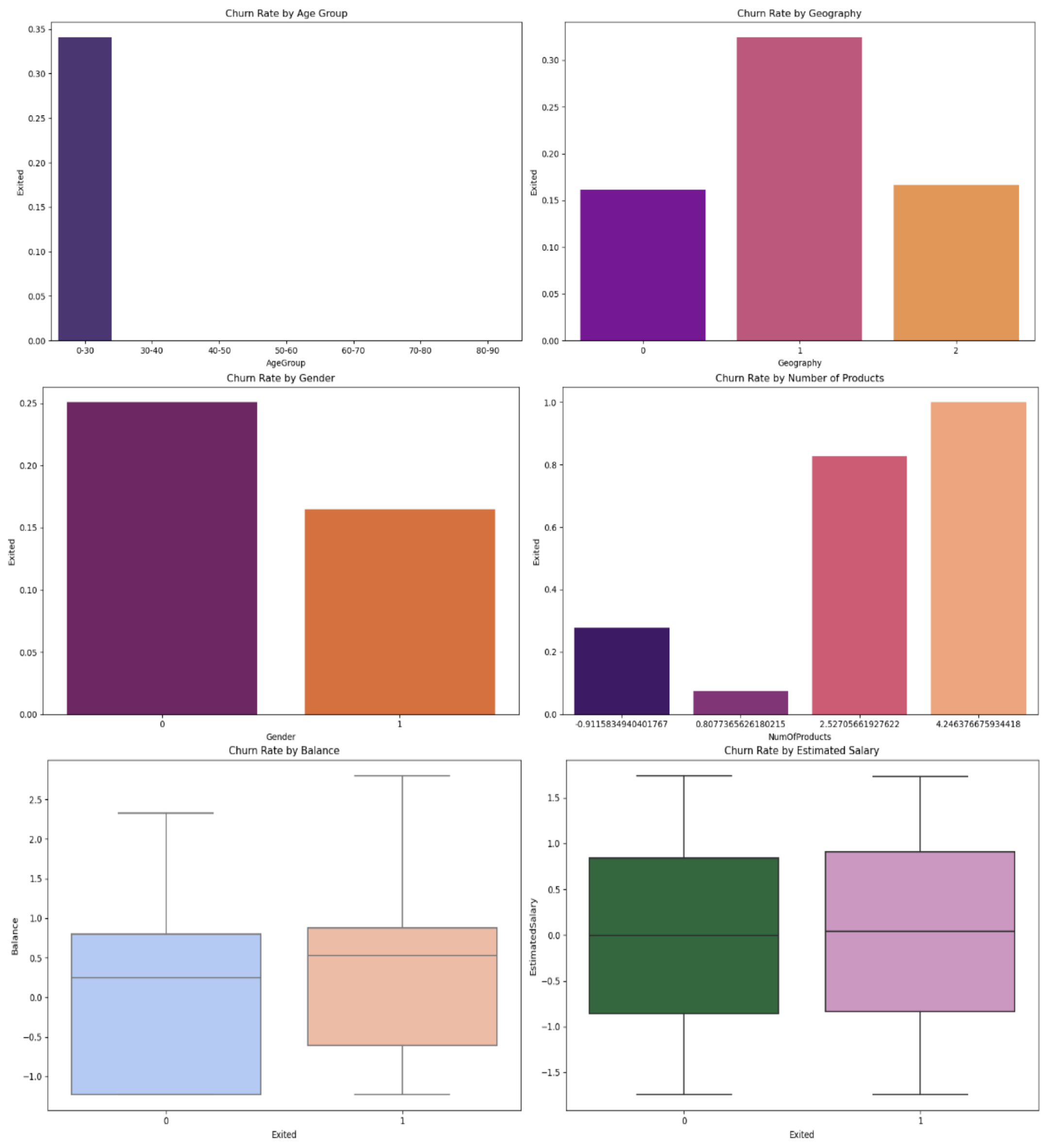


Figure 3. Bank dataset features for customer churn prediction

First, various graphical methods were used to understand the basic statistical properties of the data set. Histograms were created to examine the distribution of numerical variables. For example, the histogram showing the distribution of customer age revealed how the ages of customers were distributed and whether there were concentrations in certain age ranges. Histograms helped quickly understand the overall distribution of the data set by visualizing the frequencies of certain values in the data set.

Column charts were used to analyze the distribution of categorical variables in Figure 3. These charts show how many data points are in each category. For example, when analyzing the geographical distribution of customers, the number of customers in each geographic region was visualized. This analysis helped understand in which regions customers are concentrated and customer behavior in certain regions.

While analyzing the data set, a correlation matrix was used to better understand the relationships between variables and this matrix was displayed graphically. The correlation matrix measures the linear relationship between each pair of variables and usually takes values between -1 and 1. A value of -1 indicates a completely negative correlation between two variables; A value of 1 indicates a completely positive correlation. A value of 0 indicates that there is no linear relationship between the two variables. By visualizing this matrix, relationships between variables in the data set can be detected more easily and quickly.

A heatmap was used to graphically represent the correlation matrix. The heat map represents correlation values with color tones, so it can be quickly visually discerned which pairs of variables have strong or weak correlations. The color tones on the chart indicate positive correlations, generally with warmer (red) colors, and negative correlations with cooler (blue) colors. Increasing color intensity indicates increasing strength of the correlation.

Numerical variables of the data set were used to create the correlation matrix. In this way, the relationships between variables such as credit score, age, balance, number of products, credit card ownership status, active membership status and estimated salary were examined. Looking at the correlation matrix, it was found that some variables had high correlations with each other, while others showed low correlations. This information guided the study on which variables should be included in the model when establishing a logistic regression model.

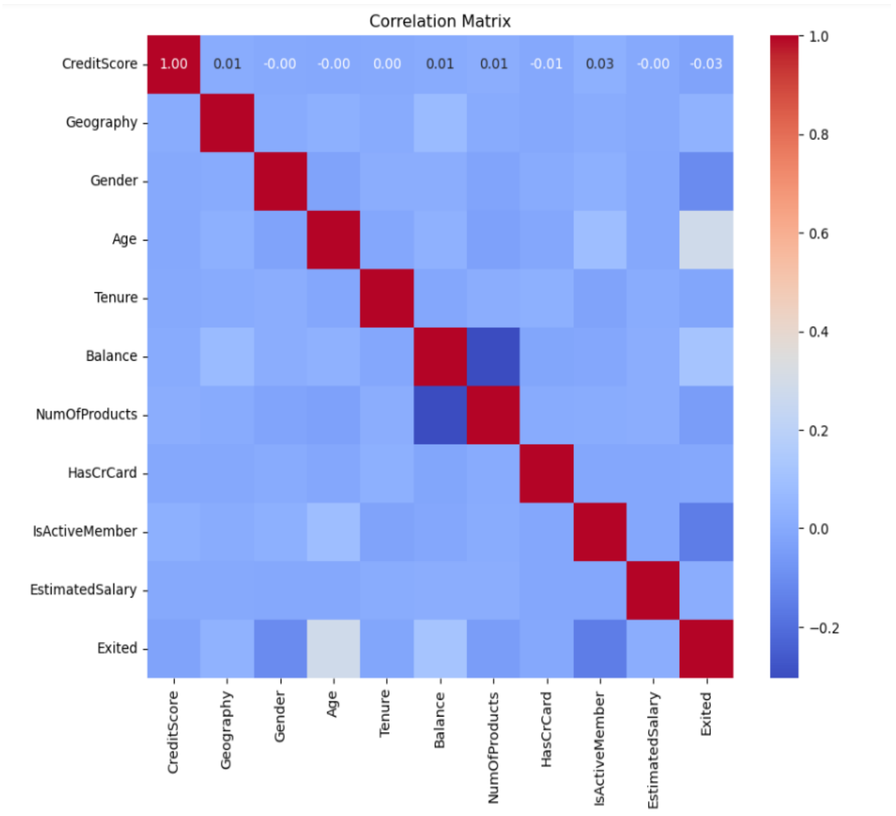


Figure 4. Correlation analysis for customer churn prediction

For example, a negative correlation has been observed between customer age and credit score (Figure 4). This may indicate that your credit score decreases with age. Similarly, a positive correlation was found between customer balance and estimated salary, which may indicate that as salary increases, bank balance also increases. This information from correlation analysis helped us understand the structure of our data set and make more informed decisions when building our logistic regression model.

By analyzing the distribution of data in the data set and the relationships between variables with these methods, the general structure of the data set was better understood. This understanding allowed more informed decisions to be made during the modeling process and provided important information in preparation for advanced analysis methods such as logistic regression model. Understanding the distribution and correlations of the data played a critical role in both the data cleaning and preprocessing steps and the model building and evaluation stages.

Logistic Regression

While applying the logistic regression model, various factors were taken into consideration to improve model performance. Among these, trying to get the best performance by trying different parameter combinations using GridSearchCV, making appropriate parameter adjustments to prevent problems such as overfitting and underfitting, reliably evaluating the performance of the model with the cross-validation method, selecting parameter values appropriately for the problem and data set, unbalanced classes. There are factors such as addressing and taking precautions if necessary. These factors are very important to increase the accuracy of the model and obtain reliable results.

As a result of the hyperparameter grid search, it was determined with which hyperparameters the best performance of the model was achieved. According to the hyperparameter grid search results for the Logistic Regression model, the hyperparameter combinations that achieved the best performance were determined. During this analysis process, different regularization forces (C), penalty norms (penalty) and optimization algorithms (solver) were evaluated in order to optimize the accuracy of the model. The grid search result graph visualizes the model's accuracy scores for each hyperparameter combination. Logistic Regression Hyperparameter Grid Search chart values and results of the data set are as shown in the chart below:

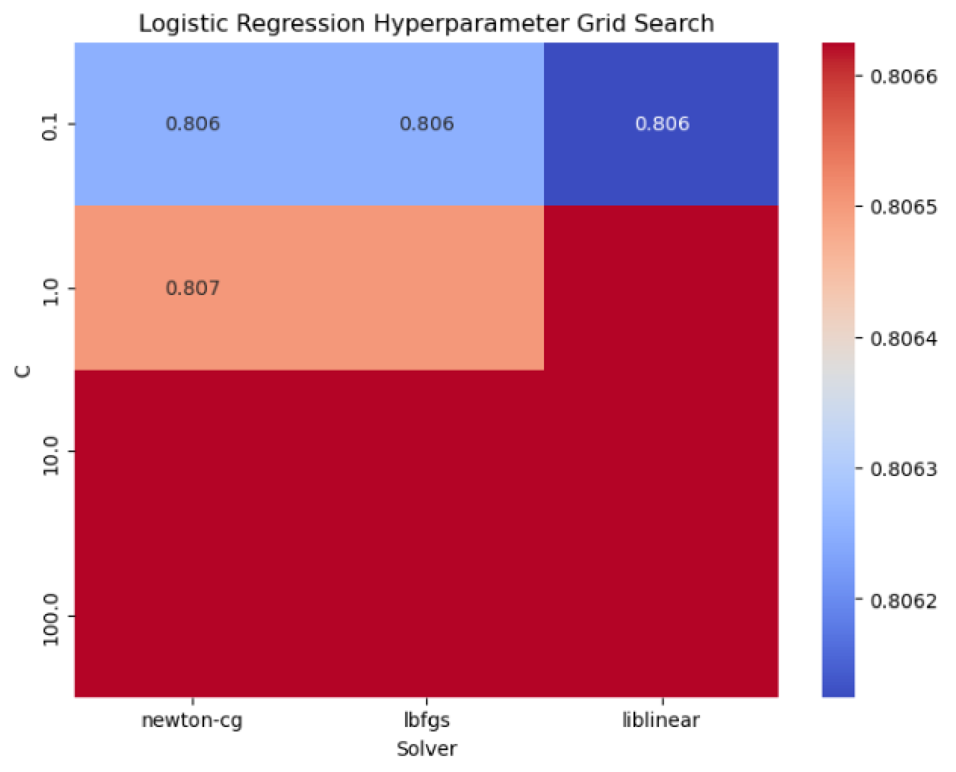


Figure 5. Logistic regression hyperparameters gridseacrh CV

Figure 5 shows that the highest accuracy scores are concentrated in certain combinations of hyperparameters. In particular, it has been observed that the combination where the 'C' value is '1' and the 'penalty' value is 'l2' has the highest accuracy. This finding shows that the model achieves its best performance with these hyperparameters. Another important finding is that the performance of the model increases significantly when 'liblinear' is used as the 'solver' parameter. This solver is known for performing well, especially on small datasets and low-dimensional data. The hyperparameter grid search plot reveals that the C value inversely determines the strength of regularization, with medium-sized C values (0.1, 1, 10) providing higher accuracy. It has been observed that extremely large or extremely small C values may negatively affect the performance of the model. Therefore, the C value must be chosen carefully. When looking at the penalty parameter, it is seen that the l2 norm generally provides higher accuracy. Different performances were observed between L1 (Lasso) and L2 (Ridge) norms, and it was understood that the L2 norm was more effective in terms of regularization. This shows that the l2 norm is more successful in reducing the overfitting problem of the model.

As a result of the hyperparameter grid search, it was determined with which hyperparameters we achieved the best performance of our model. In particular, it was determined that the model reached the highest accuracy when C value was used as 1, l2 as penalty and liblinear as solver. These combinations of hyperparameters can be used to optimize our model and obtain more reliable results. This process contributes to more effective and efficient use of the model in real-world applications. As a result, the best hyperparameter combinations for the Logistic Regression model were determined thanks to the hyperparameter grid search chart. These combinations can be used to increase the model's accuracy and optimize prediction performance. Retraining the model with the best hyperparameters will enable more accurate and reliable predictions.

Figure 6 showing the best hyperparameters of our Logistic Regression model and the accuracy values obtained with these hyperparameters is an important tool to evaluate the performance of the model. This graph visualizes the impact of different combinations of C (inverse regularization strength), Threshold (decision making threshold), Tol (tolerance) values on accuracy. First, the C value represents the regularization parameter of the logistic regression model. This value determines how complex or simple the model will be. Looking at the results in the graph, it can be seen that C varies between 0.6 and 0.7. In this range, it can be said that the model is optimized and performs well. The Threshold value determines how classification decisions are made in the logistic regression model. According to the data in the chart, the threshold value varies between 0.2 and 0.3. This value indicates how precise or flexible the model will classify. The Tol value is a tolerance value that determines when the model converges. According to the data in the chart, the tol value varies between 0.0 and 0.1. This value indicates how accurate the model is in the optimization process.

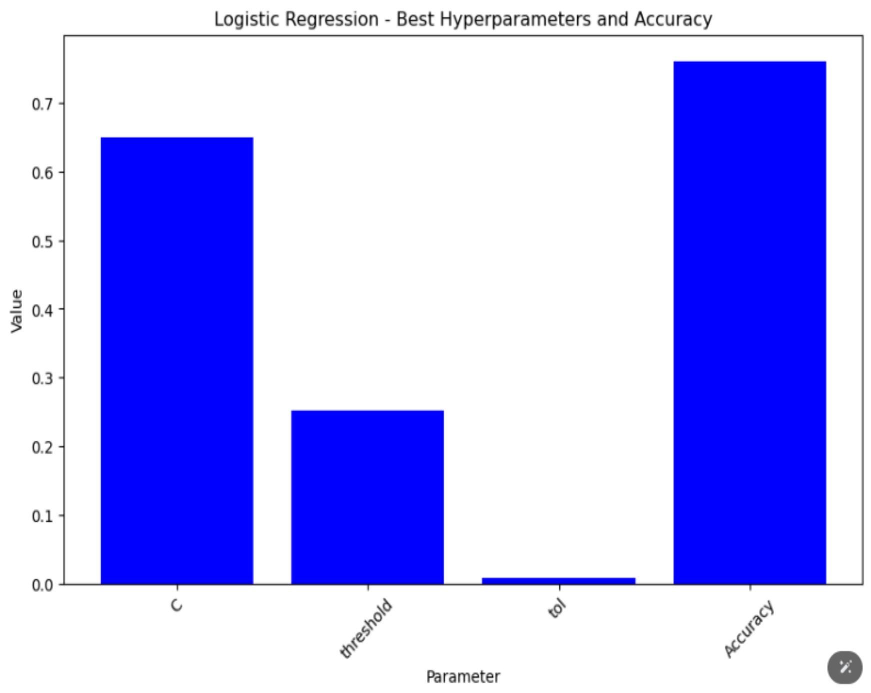


Figure 6. Logistic regression best hyperparameters and accuracy

Decision Tree

The optimized decision tree model is evaluated on the test data set and the results are visualized. Additionally, the performance and accuracy values of the decision tree model according to its parameters are examined comparatively. Decision Tree Hyperparameter Grid Search chart values and results for the data sets are as shown in Figure 7:

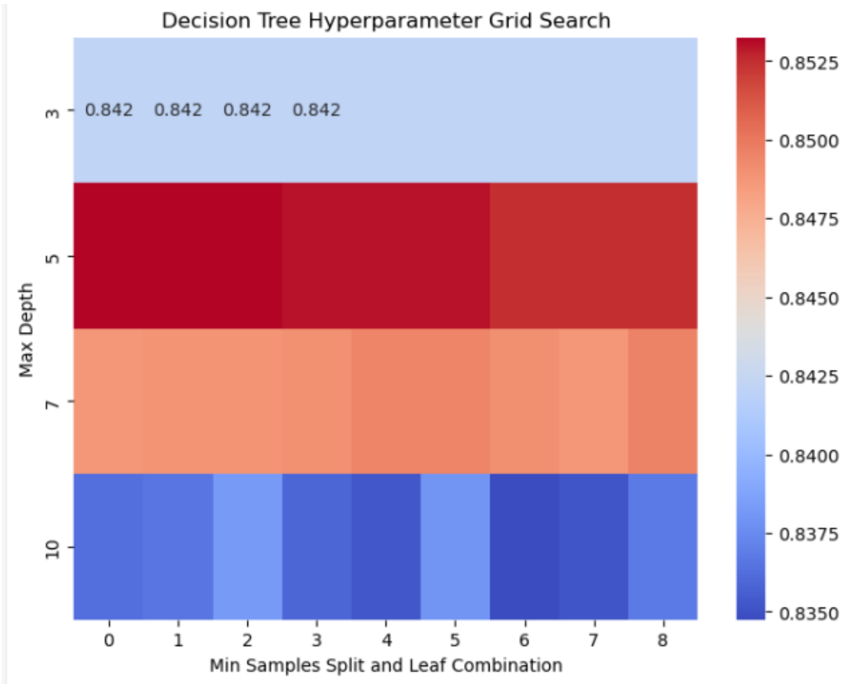


Figure 7. Decision tree hyperparameters GridSeacrh CV

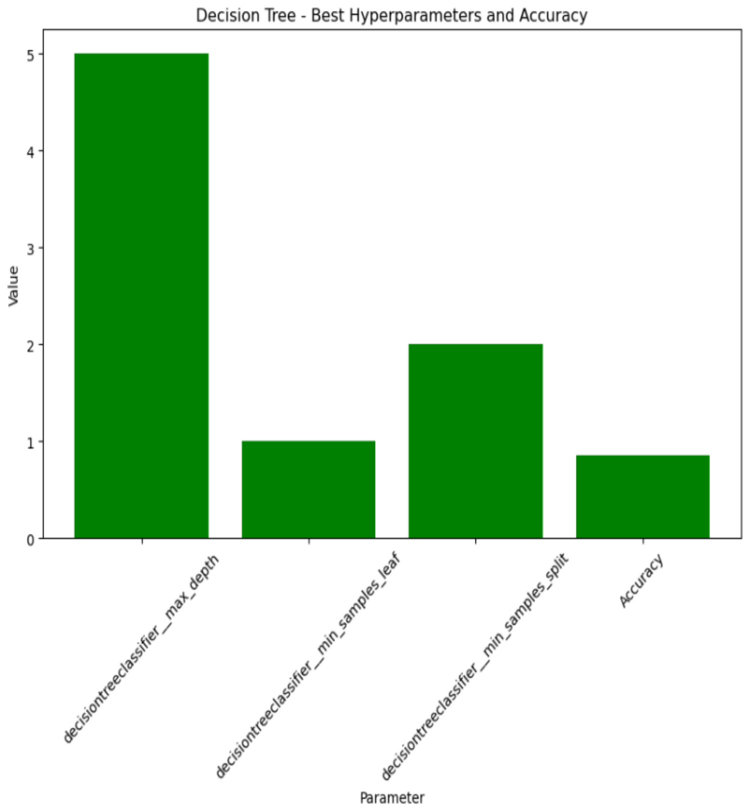


Figure 8. Decision tree hyperparameters and accuracy

- Max Depth: Determines the maximum depth of the tree. As tree depth increases, the model becomes more complex and can better fit the training data, but it can also become prone to overfitting. By choosing various depths such as 5, 10, 15, 20, 25, it was aimed to test the performance of the model at different complexity levels.
- Min Samples Split: Specifies the minimum number of samples required to split an internal node. If this value is low, the model may make more divisions and its complexity increases. By selecting values of 2, 5, 10, 15, 20, the effect of division operations on the model performance was examined.
- Min Samples Leaf: Determines the minimum number of samples that should be present in a leaf node. If this value is low, the model may make more divisions. By choosing values of 1, 2, 5, 10, 15, the effect of the minimum number of samples in leaf nodes was examined.
- Criterion: Determines the criterion to be used in divisions. There are two options, 'gini' and 'entropy'. Both criteria were selected and it was tested which criterion provided better performance.

If we interpret the Figure 8 in detail, the maximum depth values on the X axis vary as 5, 10, 15, 20 and 25. In general, a moderate maximum depth (for example, 10) performs better. The combined effect of min_samples_split and min_samples_leaf values on the y axis was examined. It is observed that the performance improves when these two parameters, which vary along the y-axis in the graph, take higher values.

The color tones in the graph visually present the performance of different hyperparameter combinations. Dark colors represent higher performance and light colors represent lower performance. This visual analysis helps determine which hyperparameter values should be selected to achieve the best performance. The best parameter combinations obtained as a result of Grid Search were used to increase the accuracy of the model. Optimal parameters provide better performance by balancing the complexity and generalization ability of the model. In conclusion, the Decision Tree Hyperparameter Grid Search chart shows how different parameter combinations affect model accuracy. By analyzing this graph, we can select the parameters where the model performs best and minimize the risk of overfitting. According to the graphical results obtained from the code, the accuracy of the model was maximized by selecting the optimum values of the max_depth and min_samples_split parameters.

Support Vector Machines (SVM)

The model is trained with the best hyperparameter values obtained after the Grid Search process, and its accuracy is measured on the test data set. This metric shows how well the model performs under the conditions where it achieves its best performance. The results obtained provide important information about the generalization ability and accuracy rate of the model. This analysis is a critical step to evaluate how well the model works with real-world data. The SVM model was created with StandardScaler using the 'make_pipeline' function. This ensures standardization of features across the dataset and helps the model perform better. GridSearchCV is trained to determine the best parameters of the model using the specified hyperparameter grid. 5-fold cross validation determined by the 'cv' parameter was used. Then, the best model was evaluated with the test data and the test accuracy score was calculated. This value indicates the generalization ability of the model. Finally, the accuracy scores of the hyperparameter combinations were visualized using a heat map. This heatmap shows the average accuracy scores obtained for different C and gamma values, using color coding.

Figure 9 shows the average accuracy scores obtained for different C and gamma values. Each cell corresponds to a specific combination of C and gamma values. The graph visualizes the accuracy scores of these combinations using color coding. Dense colors represent higher accuracy scores, while lighter colors represent lower accuracy scores. Therefore, it is possible to get an idea about which combinations of hyperparameters achieve the best performance. For example, when the C value is 1 and the gamma value is 'scale' (with a high accuracy score), the RBF kernel may be the best preferred configuration. However, the heat map must be taken into account to observe the interaction of different C and gamma values. For example, one can examine how accuracy scores change if gamma is 'scale' and 'auto' for a given C value. Such analyzes are important to understand which hyperparameters and their values of the model provide the best performance.

- When C Value is 1: The highest accuracy scores were obtained with the 'scale' gamma value. This combination allows the model to perform well on both training and testing data.
- When C Value is 10: Performance decreases slightly, which indicates that the model may be slightly overfitting.
- When C Value is 0.1: The accuracy score is lower, which indicates that the model is not complex enough and the accuracy rate decreases.

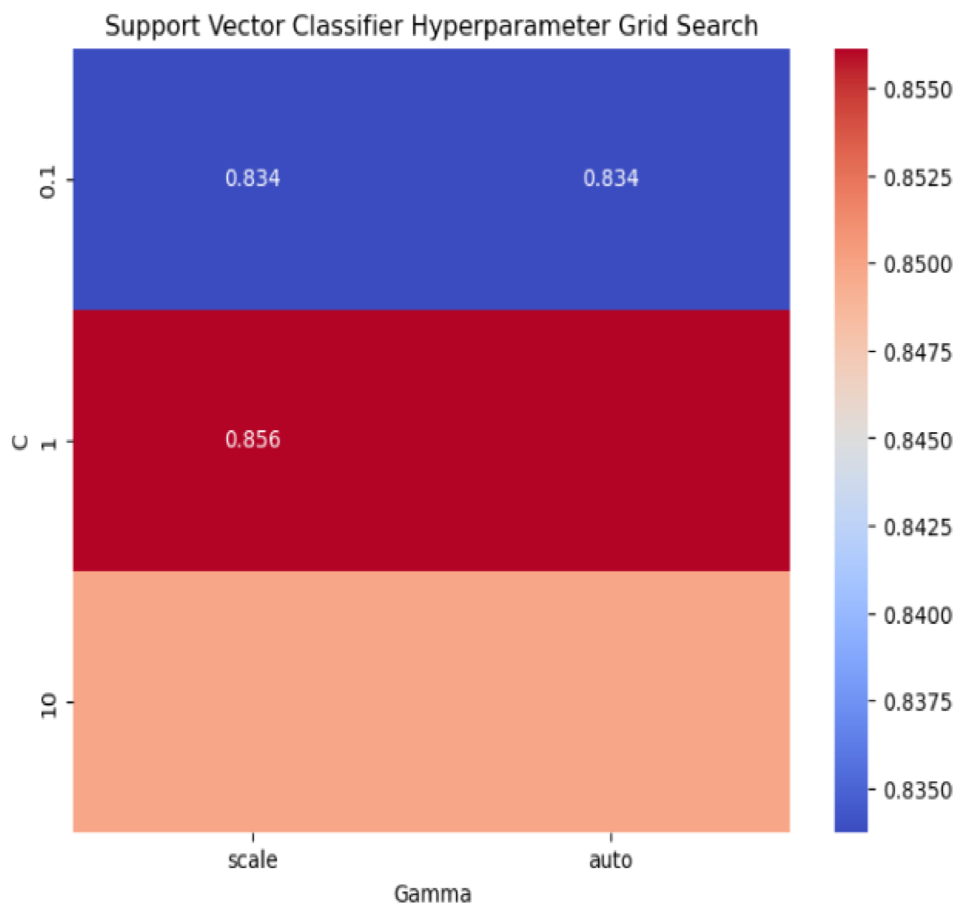


Figure 9. SVM hyperparameters GridSeacrh CV

When the 'C' value is selected as 1, the model performs well in balancing errors and correct classifications. When the 'Gamma' value is set to 'scale', this parameter is automatically scaled according to the characteristics of the data set. When 'rbf' (radial basis function) is selected as the 'kernel' type, the model can better learn complex relationships between classes. The best CV score was found to be 0.8561. This score expresses how well the model performs on the training data. This score, obtained by 5-fold cross-validation, also provides information about the generalization ability of the model.

The test accuracy score was found to be 85.75%. This score is the accuracy rate achieved by the model on test data. The model trained with the best parameters determined during training performed with an accuracy rate of 85.75% on the test data. This shows that the model can also perform well on real-world data.

Gradient Boosting Classifier

Gradient Boosting Classifier (GBC) was optimized using grid search method for various hyperparameters. In this analysis, the effects of three main hyperparameters such as learning rate, number of estimators and maximum depth were evaluated. In particular, a heatmap was created to visualize the effect of the learning rate and the number of trees on the performance of the model (Figure 10).

The best hyperparameters were determined as n_estimators (number of trees) 200, learning_rate (learning rate) 0.1 and max_depth (maximum depth) 3. This combination achieved the highest average accuracy score (0.869) during the cross-validation (CV) process. These parameters were determined as the combination that achieved the highest average accuracy score during the cross-validation (CV) process. The model trained with the best hyperparameters was also evaluated on the test data set and a result of 86.75% was obtained for test accuracy. This result demonstrates the performance of the model on real-world data and is consistent with the high accuracy score obtained during cross-validation. The graph showing the best parameters and accuracy value is as follows:

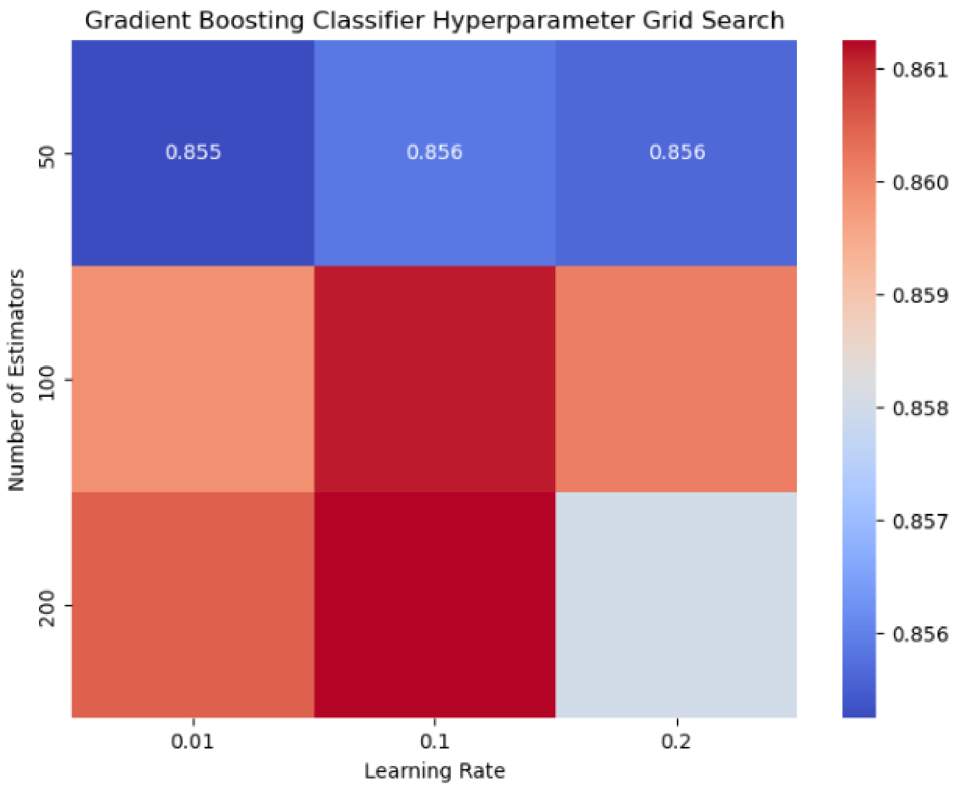


Figure 10. Gradient boosting classifier hyperparameters GridSeacrh CV

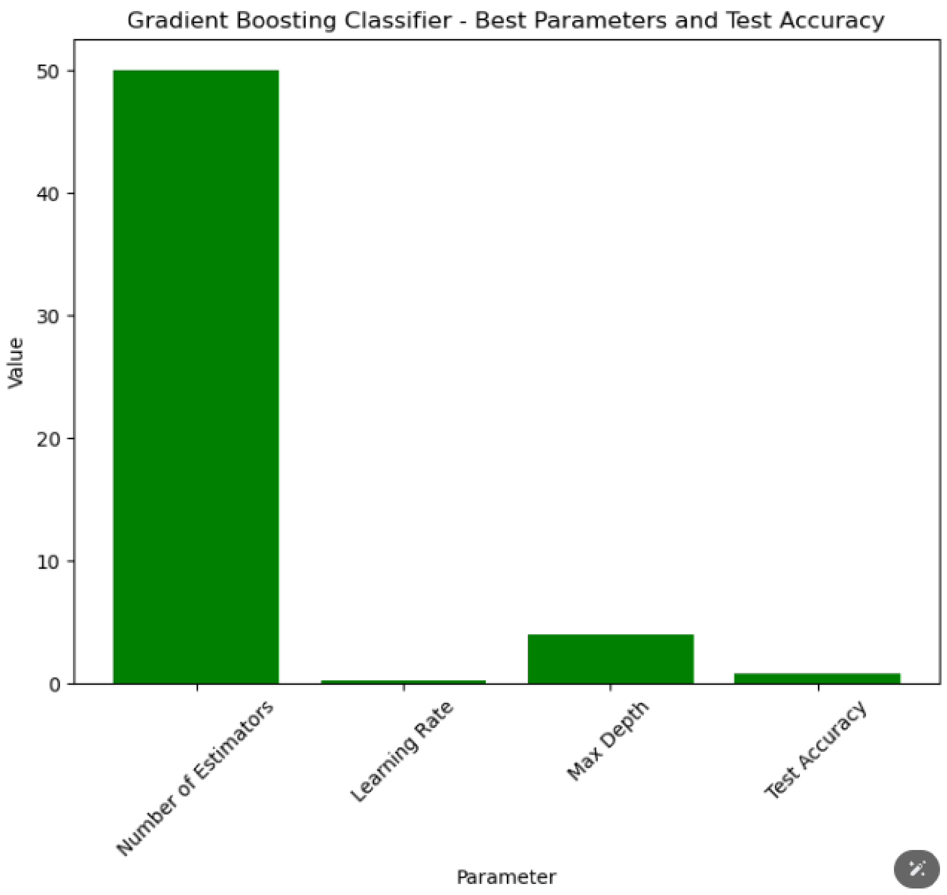


Figure 11. Gradient boosting classifier best parameters and test accuracy

Heatmap visualizes the average accuracy scores obtained during the grid search process, based on the learning rate and number of trees parameters. This visualization helps us understand how hyperparameter combinations affect model performance (Figure 11).

- Higher accuracy scores were obtained with a learning rate between 0.1 and 0.2.
- It has been observed that accuracy scores generally increase as the number of trees increases (from 50 to 200). This shows that more trees make the model more powerful and generalizing.

Learning Rate:

- 0.01: Low learning rate is generally associated with low accuracy scores. This causes the model to learn slowly and therefore not generalize well enough.
- 0.1: It is seen as the optimal learning rate. In this range, the model learns fast enough while also showing good generalization capacity.
- 0.2: Although high accuracy scores were obtained, it was observed that a learning rate of 0.2 could lead to overfitting in some cases.

Number of Trees:

- 50: Low number of trees is generally associated with low accuracy scores. This indicates that the model is not complex enough and fails to capture the complex structure of the data.
- 100 and 200: Higher tree numbers significantly increased the performance of the model. In particular, the highest accuracy scores were achieved with 200 trees.

Best Combination:

- n_estimators=200 and learning_rate=0.1: This combination represents the region on the heatmap where the highest accuracy scores were obtained. This indicates that these parameters are optimal and maximize the generalization ability of the model.

Heatmap analysis is an important tool in determining optimal hyperparameter combinations for the Gradient Boosting Classifier. This analysis shows which hyperparameters are more effective to maximize the performance of the model. The model, with a learning rate of 0.1 and number of trees of 200, achieved high accuracy scores on both cross-validation and the test data set, confirming that this combination gave the best results. These findings provide valuable information on how to optimize the Gradient Boosting Classifier model on a specific data set and serve as a guide for future work.

K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm is a method that attracts attention with its simplicity and flexibility in the field of machine learning. It can provide effective results especially on small and medium-sized data sets. However, it has disadvantages such as computational cost and memory usage in large data sets. With the selection of appropriate hyperparameters and correct scaling of the data, KNN can be used successfully in many application areas. Selection of the most appropriate algorithm for a particular problem should be made by taking into account the advantages and disadvantages of KNN. Here, a heatmap was created that visualizes the results of the Grid Search process performed to optimize the hyperparameter settings of the K-Nearest Neighbors (KNN) algorithm. This map is used to visualize how different combinations of hyperparameters affect the performance of the model .

Figure 12, the horizontal axis represents the weights hyperparameter, and the vertical axis represents the n_neighbors (number of neighbors) hyperparameter. Each cell shows the average cross-validation score for the corresponding hyperparameter combination. This value reflects how well the model performs with a particular combination of hyperparameters.

- Horizontal Axis: Weights

1. uniform: Gives equal weight to all neighbors.

2. distance: Gives neighbors weight inversely proportional to their distance; that is, closer neighbors are given more weight.

- Vertical Axis: Number of Neighbors

o 3, 5, 7: Specifies how many neighbors the model will take into account. Lower values may cause the model to be more complex and overfitting, while higher values may cause it to be simpler and more generalizing (underfitting).

- Color Scale and Scores

o Color scale represents average cross-validation scores. Darker colors (especially dark red) indicate lower validation scores, and lighter colors (toward yellow) indicate higher validation scores.

o The numerical value in each cell indicates the average cross-validation score obtained with the corresponding hyperparameter combination.

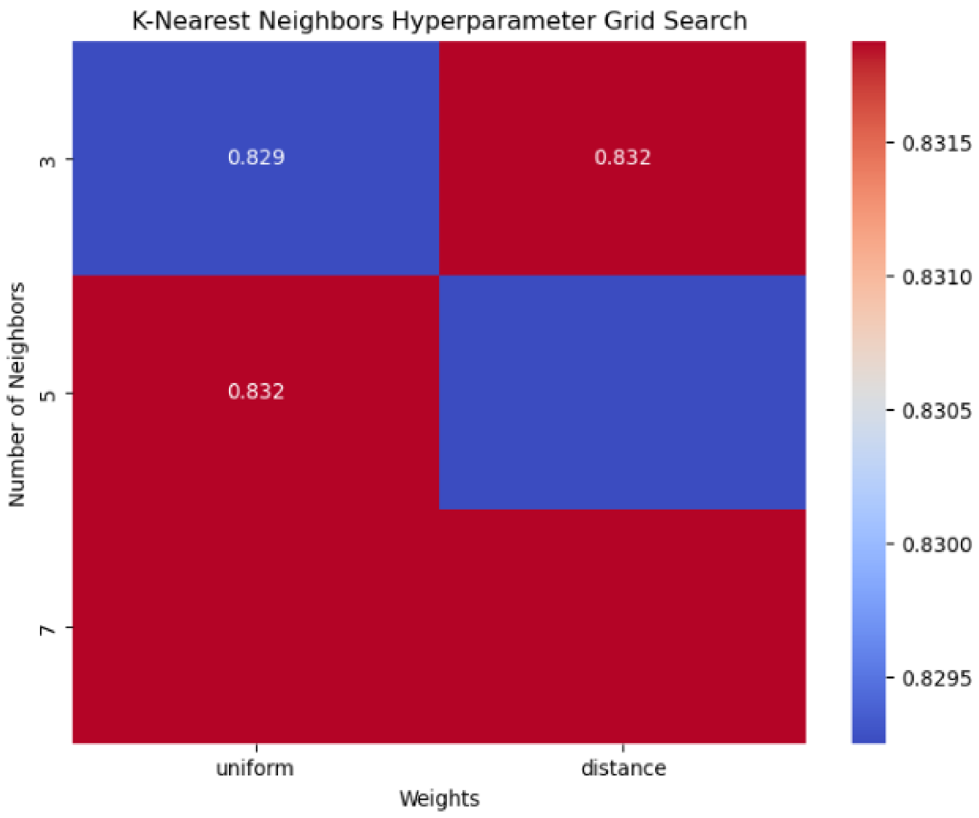


Figure 12. KNN hyperparameter GridSearchCV

In general, we can observe that the 'distance weighting method provides higher validation scores than the 'uniform' method. This suggests that giving more weight to closer neighbors can improve the performance of the model. When the number of neighbors was 5 or 7, higher validation scores were obtained, especially with the distance weighting method. This indicates that the number of neighbors must be carefully selected to achieve the optimal performance of the model. It is observed that the highest scores are obtained in combinations of n_neighbors = 5 or 7 and weights = distance. This indicates that the best combination of hyperparameters lies at these values.

This heatmap was used to visualize how the hyperparameters (number of neighbors and weighting method) of the K-Nearest Neighbors model affect the model performance. The best validation scores were obtained in combinations where the number of neighbors was 5 or 7 and the weighting method was distance. This analysis highlights the importance of choosing the right hyperparameters to improve the performance of the model. Such visualizations are a critical tool for optimizing the model and achieving the best results.

Multi-layer Perceptron

Grid search was used to optimize the hyperparameters of the MLP model. The determined hyperparameters are in Figure 13:

- hidden_layer_sizes: Number of neurons in the hidden layer (determined as 50 and 100).
- activation: Activation function (Only 'relu' is used in our study).
- solver: Optimization algorithm (Only 'man' is used in our study).
- alpha: L2 regularization parameter (set as 0.0001 in our study).
- learning_rate: Learning rate (specified as 'constant' in our study).

As a result of the grid search, it was seen that the best hyperparameters (hidden_layer_sizes: (100,), activation: 'relu', solver: 'adam', alpha: 0.0001, learning_rate: 'constant') were determined and the cross-validation (CV) score of the model was measured as 0.875. . The model obtained with these hyperparameters showed successful performance on the dataset. In addition, the Heatmap method enabled the effect of certain hyperparameter combinations to be easily seen on model performance. While creating the heatmap, cross-validation scores were calculated for each hyperparameter combination and the best combination was determined using these scores. This heatmap is created only for hidden_layer_sizes and activation hyperparameters. Other hyperparameters (solver, alpha, learning_rate) were kept constant. The heatmap's axes are as follows:

- On Y Axis: hidden_layer_sizes (Numbers of Neurons in the Hidden Layer)
- On X Axis: activation (Activation Function)

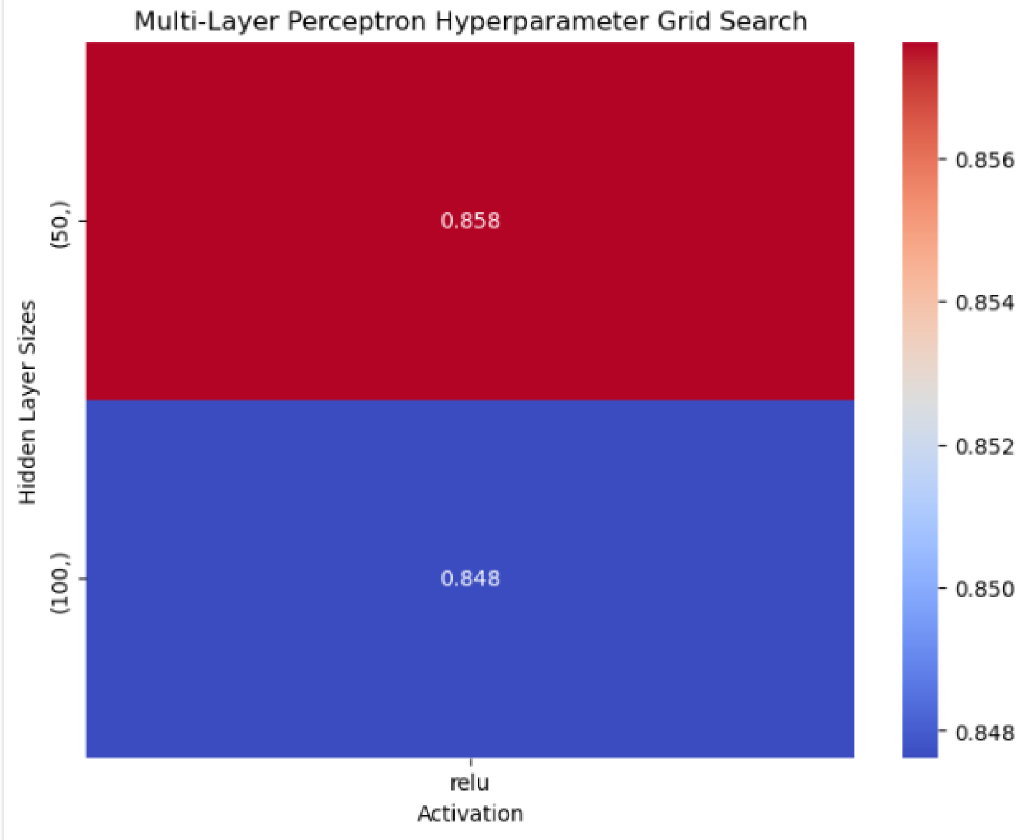


Figure 13. MLP hyperparameter GridSearchCV

However, since only the 'relu' activation function is used in this example, the X-axis represents a single value.

- Scores of hidden_layer_sizes and activation combinations: Average cross-validation scores were calculated for each combination and these scores were visualized through colors on the heatmap.

- Best hyperparameter combination: The cell representing the highest score in the heatmap shows the best hyperparameter combination. In this case, (100,) hidden_layer_sizes and 'relu' activation function worked best.

In this example, the grid search process used to determine the best hyperparameter combination of the MLP model and the results of this process are visualized with a heatmap. The obtained results clearly show how the MLP model can achieve higher performance with the correct hyperparameters. The power, flexibility, and generalization capacity of MLP make it an ideal choice for solving many machine learning problems. However, determining the correct hyperparameters is critical to maximize model performance, and visualization tools such as heatmap provide great convenience in this process.

Results and Discussion

The graph was created to compare the performances of different machine learning algorithms (Logistic Regression, Gradient Boosting Classifier, K-Nearest Neighbors, Multi-Layer Perceptron, Support Vector Classifier and Decision Tree Classifier) and the best parameters of these algorithms (Figure 14). Each bar represents a machine learning algorithm. The height of the bars indicates the accuracy score obtained by each algorithm on the test data set. A high bar indicates that the corresponding algorithm is performing better. Each line indicates the best parameter of the corresponding algorithm. For example, a dashed line represents the n_neighbors (number of neighbors for K-Nearest Neighbors) parameter for which the respective algorithm performs best. The dashed line visually indicates the value of the corresponding parameter. The x-axis represents each algorithm. The name of each algorithm is indicated in labels below the axis. The y-axis represents test accuracy. Accuracy is the ratio of correctly classified samples to the total number of samples. High accuracy indicates how well a model performs on the test data set. This graph is very useful for comparing the performance of different machine learning algorithms. To determine which algorithm performs better in which situations, attention should be paid to both test accuracy and best parameters. For example, one algorithm may achieve higher accuracy than others, but at a certain parameter setting this performance may decrease. This type of analysis can be extremely valuable for optimizing the performance of a machine learning application.

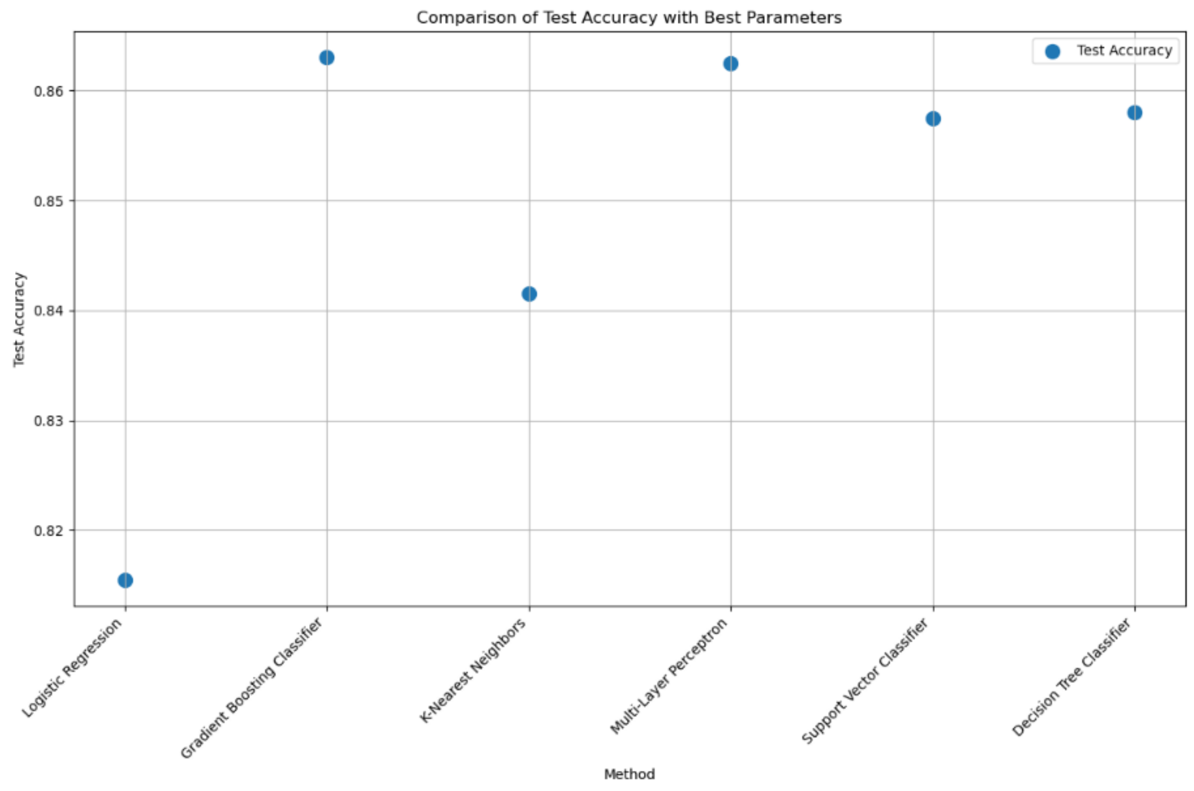


Figure 14. All proposed models test accuracy results for best parameters

Based on Figure 7, the performances of different machine learning methods were compared. The Gradient Boosting Classifier method stands out as the most effective method as it has the highest test accuracy compared to other methods. These results take into account the complexity of the dataset and nonlinear relationships and highlight the efficiency of the algorithm. Furthermore, these results are generally consistent with Cross-Validation (CV) scores, indicating that the models' overall CV performance is a good measure of predicting test accuracy. However, in some cases the CV score may not predict test accuracy well enough, so it is important to consider both metrics when choosing a model. As a result, the Gradient Boosting Classifier method provides the highest test accuracy by best capturing the features of our data set. Therefore, this method is more likely to perform better in real-world applications.

Table 7. All proposed models test accuracy for GridSearchCV

Method	Best Parameters	CV Score	Test Accuracy
Logistic Regression	C : 1 solver: liblinear	0.806625	0.8155
Decision Tree	max_depth: 5 min_samples_split: 2 min_samples_leaf: 1	0.853250	0.8580
Support Vector Classifier	C : 1 gamma: scale kernel: 'rbf'	0.856125	0.8575
Gradient Boosting Classifier	n_estimators: 50 learning_rate: 0.2 max_depth: 4	0.861125	0.8630
K-Nearest Neighbors	n_neighbors: 7 weights: uniform algorithm: auto	0.831875	0.8415
Multi-Layer Perceptron	hidden_layer_sizes:(100,) activation: relu solver: adam alpha: 0.0001 learning_rate: constant	0.853625	0.8625

As a result of the Grid Search study conducted for the data set, the hyperparameters that provide the best performance were determined. These hyperparameters; 'entropy' as the criterion, 10 as the maximum depth, 2 as the minimum number of sample divisions and 1 as the minimum number of leaf samples. The highest accuracy score obtained with this hyperparameter combination was recorded as 85.6%. This result demonstrates the classification success of the model on the dataset and confirms that the determined hyperparameters optimize the overall performance of the model.As a result, the Decision Tree - Best Hyperparameters and Accuracy chart obtained from the code reveals the parameter combinations where the model performs best. Optimizing max_depth and min_samples_split values increases both the accuracy and generalization ability of the model. Careful selection of these parameters maximizes the model's performance on training and test data, resulting in more reliable results.

Conclusion

This study focuses on predicting customer churn in the banking sector. Customer churn is a major concern for financial institutions, and estimating these losses and assessing their impacts are critical to financial sustainability and competitive advantage. In our study, a series of models were developed to predict customer churn using machine learning methods. These models include Logistic Regression, Decision Tree, Support Vector Classifier, Gradient Boosting Classifier, K-Nearest Neighbors and Multi-Layer Perceptron methods. Each of these methods has different features and advantages, and a variety of metrics have been used to evaluate their ability to predict customer churn.

The results obtained show that the Gradient Boosting Classifier method has the highest test accuracy compared to other methods. This suggests that the model's ability to predict customer churn is most effective when it takes into account the complexity of our data set and non-linear relationships. These findings encourage financial institutions to use machine learning models to predict customer churn and develop appropriate strategies. In addition to reducing customer churn, these models can also help financial institutions take preventive steps to increase customer satisfaction. However, this study has some limitations. The focus of our data set on a specific time period and a specific geographic region may limit the generalizability of the results. It should also be noted that the results obtained could be improved by collecting more data and using more complex modeling techniques. This study contributes to research on predicting customer churn in the banking industry, while also helping financial institutions better understand their strategies for maintaining and expanding their customer base. Future studies can further deepen knowledge in this area with broader data coverage and more comprehensive analysis.

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM Journal belongs to the authors.

Acknowledgements or Notes

* This article was presented as an oral presentation at the International Conference on Technology, Engineering and Science (www.icontes.net) held in Antalya/Turkey on November 14-17, 2024.

References

Amuda, K. A., & Adeyemo, A. B. (2019). Customers churn prediction in financial institution using artificial neural network. *arXiv preprint arXiv*:*1912*,11346.

De Caigny, A., Coussement, K., & De Bock, K. W. (2018). A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. *European Journal of Operational Research, 269*(2), 760-772.

Faritha Banu, J., Neelakandan, S., Geetha, B. T., Selvalakshmi, V., Umadevi, A., & Martinson, E. O. (2022). Artificial intelligence based customer churn prediction model for business markets. *Computational Intelligence and Neuroscience, 2022*(1), 1703696.

Gregory, B. (2018). Predicting customer churn: Extreme gradient boosting with temporal data. *arXiv preprint arXiv:1802*, 03396.

Höppner, S., Stripling, E., Baesens, B., vanden Broucke, S., & Verdonck, T. (2020). Profit driven decision trees for churn prediction. *European Journal of Operational Research, 284*(3), 920-933.

Imani, M., & Arabnia, H. R. (2023). Hyperparameter optimization and combined data sampling techniques in machine learning for customer churn prediction: a comparative analysis. *Technologies*, *11*(6), 167.

Ismail, M. R., Awang, M. K., Rahman, M. N. A., & Makhtar, M. (2015). A multi-layer perceptron approach for customer churn prediction. *International Journal of Multimedia and Ubiquitous Engineering, 10*(7), 213-222.

Jain, H., Khunteta, A., & Srivastava, S. (2020). Churn prediction in telecommunication using logistic regression and logit boost. *Procedia Computer Science, 167*, 101-112.

Jajam, N., Challa, N. P., Prasanna, K. S., & Ch, V. S. D. (2023). Arithmetic optimization with ensemble deep learning SBLSTM-RNN-IGSA model for customer churn prediction. *IEEE Access*.

Khattak, A., Mehak, Z., Ahmad, H., Asghar, M. U., Asghar, M. Z., & Khan, A. (2023). Customer churn prediction using composite deep learning technique. *Scientific Reports*, *13*(1), 17294.

Liu, Z., Jiang, P., De Bock, K. W., Wang, J., Zhang, L., & Niu, X. (2024). Extreme gradient boosting trees with efficient Bayesian optimization for profit-driven customer churn prediction. *Technological Forecasting and Social Change*, *198*, 122945.

Nie, G., Rowe, W., Zhang, L., Tian, Y., & Shi, Y. (2011). Credit card churn forecasting by logistic regression and decision tree. *Expert Systems with Applications, 38*(12), 15273-15285.

Raeisi, S., & Sajedi, H. (2020, October). E-commerce customer churn prediction by gradient boosted trees. In 2020 *10th International Conference on Computer and Knowledge Engineering (ICCKE)* (pp.55-59). IEEE.

Sana, J. K., Abedin, M. Z., Rahman, M. S., & Rahman, M. S. (2022). A novel customer churn prediction model for the telecommunication industry using data transformation methods and feature selection. *Plos one, 17*(12), e0278095.

Sjarif, N., Rusydi, M., Yusof, M., Hooi, D., Wong, T., Yaakob, S., ... & Osman, M. (2019). A customer Churn prediction using pearson correlation function and K nearest neighbor algorithm for telecommunication industry. *International Journal of Advance Soft Computing Applications, 11*(2).

Tang, Q., Xia, G., Zhang, X., & Long, F. (2020, March). A customer churn prediction model based on XGBoost and MLP. *In 2020 International Conference on Computer Engineering and Application (ICCEA)* (608-612). IEEE.

Xia, G. E., & Jin, W. D. (2008). Model of customer churn prediction on support vector machine. *Systems Engineering-Theory & Practice*, *28*(1), 71-77.

Xiahou, X., & Harada, Y. (2022). B2C E-commerce customer churn prediction based on K-means and SVM. *Journal of Theoretical and Applied Electronic Commerce Research*, *17*(2), 458-475.

Zhao, Y., Li, B., Li, X., Liu, W., & Ren, S. (2005). Customer churn prediction using improved one-class support vector machine. In *Advanced Data Mining and Applications*: *First International Conference, ADMA 2005* (Vol 1., pp. 300-306). Berlin Heidelberg: Springer.

Author Information

Yasemin Bozkurt
Kutahya Dumlupinar University
Engineering Faculty, Computer Engineering Department,
Kutahya, Türkiye

Hasan Temurtas
Kutahya Dumlupinar University
Engineering Faculty, Computer Engineering Department,
Kutahya, Türkiye

Çigdem Bakir
Kutahya Dumlupinar University
Engineering Faculty, Software Engineering Department,
Kutahya, Türkiye
Contact e-mail: *cigdem.bakir@dpu.edu.tr*

To cite this article:

Bozkurt, Y., Temurtas, H., & Bakir, C. (2024). Machine learning approaches for bank customer churn analysis *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 32*, 208-229.