Journal of Information and Data Management, 2024, 13:1, doi: 10.5753/jidm.2024.XXXXXX

A new hyperparameter optimization approach for classification models applied to bank account churn data

Gabriela M. Xavier [**Federal University of Paraná** | ***gabrielaxavier@ufpr.br***]
Alexandre C. Choueiri [**Federal University of Paraná** | ***alexandrechecoli@ufpr.br***]

Department of Production Engineering, Federal University of Paraná, Av. Cel. Francisco H. dos Santos, 100 - Jardim das Américas, Curitiba - PR, 81530-000, Brazil.

Received: DD Month YYYY • **Accepted:** DD Month YYYY • **Published:** DD Month YYYY

Abstract In this work we present a new approach for the hyperparameter optimization stage of machine learning classification models applied to banking data. The function, called F_3, considers a weighting between the classical optimization function (F_1: which considers the maximization of accuracy) and another optimization function that considers the difference between the models' accuracies (namely F_2). Optimization functions were compared for different machine learning classification models, including an ensemble of classifiers, and applied to bank account churn data to highlight the possibilities for companies in the financial sector to prevent losses by predicting and mitigating customer churn in advance. The proposed optimization function presented better results, surpassing the accuracy of the classical approach in the analyzed situations.

Keywords: Classification, Ensemble, Bank Churn, Hyperparameters

1 Introduction

With the advent of Industry 4.0, the manipulation of raw data to generate information and value is becoming an increasingly common practice in organizations. Technologies such as *Big Data*, cloud computing, the Internet of Things (IoT), and *blockchain* are examples of applications that are beginning to reshape the traditional way financial processes are executed: payment methods, fraud prevention systems, and risk assessment are some examples that gain a new dimension of analysis by incorporating these technologies [Li *et al*., 2020].

Financial sector companies, specifically banks, have increasingly turned to Artificial Intelligence (AI) and *Machine Learning* (ML) as everyday tools, both to gain a more significant understanding of raw data and to assist in decision-making [Lagasio *et al*., 2022]. These tools come to enhance a sector that is already in notable expansion: according to a report from the Central Bank of Brazil, there has been substantial growth in the *fintech* sector (financial technology companies that use technology to create innovative solutions), with a marked acceleration in the expansion rate and a high volume of applications for the establishment of new companies under review [Banco Central do Brasil, 2018].

Competitiveness in the sector is amplified by the innovations promised by *Open Banking*: a system that enables the portability of customers' data (credit, salary, loans) between institutions. With the consolidation of this tool, institutions will no longer have a monopoly on customers' data, allowing competitors to conduct more precise studies, increasing the chances of selling products and even converting accounts to new institutions.

These dynamics reinforce the demand for strategies and tools that enable financial institutions to retain their existing customers effectively [de Lima Lemos and Tabak, 2022]. Some research highlights the cost savings resulting from preventing customer churn, as acquiring new customers can be considerably more expensive, potentially up to five times the investment needed to retain and satisfy existing customers [Xiao *et al*., 2014; Sharma and Panigrahi, 2013].

Banks operate on the premise that a customer's decision to stay with or leave the bank can be somewhat predicted based on a specific set of customer characteristics (variables). Some of these include recent account activities, geographic location, age group, possession of a credit card, and income bracket.

Using this information to predict customer churn in relation to the financial institution is a fundamental task in the field of data mining, known as classification. Classification algorithms aim to assign a category or class to a dataset based on its specific characteristics (the algorithm learns the relationship between the characteristics - customer information - and the associated class - leaving or staying with the bank).

Focusing efforts on developing predictive models to anticipate customer churn can help banks take proactive and preemptive actions to reverse potential turnover and mitigate revenue losses [de Lima Lemos and Tabak, 2022].

In this paper, we conduct a preliminary comparative study of various classification algorithms for the problem of customer churn in financial institutions (decision trees, *Support Vector Machine* (SVM), *K-nearest neighbors* (KNN), and logistic regression). In addition to these comparisons, we create an *ensemble* (combination of models) [Mienye and Sun, 2022] with all classifiers and develop new functions for hyperparameter optimization, which are compared to the traditional one used by the community (maximizing test effectiveness).

This paper is organized as follows: in the next Section (2), all the theoretical background of the work is developed. In Section 3, we describe the data separation protocol, the

functioning of the classifier *ensemble*, and the proposed new optimization functions. The results are presented in Section 4, and the final comments and conclusions are described in Section 5.

2 Theoretical framework

This Section is divided into 3 parts: in the first, we describe the classification task, as well as some common concepts that will be used throughout the work, such as supervised learning, effectiveness measures, and *overfitting*. Next, the classification models used in the study are detailed, namely: decision trees, KNN, SVM, and logistic regression, all used for the classification task. Finally, Subsection 2.3 describes how it is possible to combine various models through classifier ensembles.

2.1 Classification: Common Concepts

Supervised learning can be described as the process of estimating a function f that, when applied to a dataset (X), produces the result Y. In data mining terminology, the set X is referred to as independent variables, or *features*, and the results Y as dependent variables, or *target* [Norvig and Russell, 2013]. When the learning process involves only the set X, it is called **unsupervised learning**. The classification task is one of the two that comprise supervised learning.

For a better understanding of the following sections, Figure 1 will be used as a reference for describing the steps of a supervised machine learning process.

2.1.1 The classification task

Classification is one of the four basic tasks of data mining. Along with regression, it is part of the so-called predictive tasks. A predictive task involves determining the relationship (function) between a set of input features (X) and an outcome (Y). When the outcome Y is numerical, the task is called **regression**; when it is a class, it is called **classification** [Pang-Ning Tan, 2005]. Once the relationship that describes Y from X is established, it can be used to predict values: when the set X exists, but Y is unknown.

An example of a classification task highly relevant to the banking sector is establishing the relationship between customer churn (Y) and a set of their characteristics (X). With this relationship, it is possible to predict whether a customer will leave the bank or not based on their characteristics. For instance, the vector X can be given by:

$$\begin{aligned} X &= [\text{age, time at the bank, active credit, active credit amount}] \\ &= [25, 3, \text{YES}, 500] \end{aligned}$$

And the response value indicates whether the customer has churned or not:

$$Y = [\text{YES}]$$

The classification algorithms learn the relationship between X and Y (step III of Figure 1) through a database (step I of Figure 1) of these values.

To assess the effectiveness of learning, a portion of the data is separated from the training step (step II of Figure 1), and then its Y values are removed, and the model must predict these values based solely on X [Michelucci *et al*., 2021]. Finally, we can verify the number of correct and incorrect classifications made by the model. For the calculation, we use:

1. True Positive (TP): correct prediction, the customer churned and the algorithm predicted it.
2. True Negative (TN): correct prediction, the customer did not churn and the algorithm predicted it.
3. False Positive: error, the customer did not churn and the algorithm predicted that they would churn.
4. False Negative: error, the customer churned and the algorithm predicted that they would not churn.

Based on these parameters, we can calculate the prediction effectiveness:

$$A = \frac{TP + TN}{T} \tag{1}$$

Where T represents the total number of classifications made at the end of the execution [Michelucci *et al*., 2021].

With the completion of training and the verification of accuracies, a decision must be made: is the model at a satisfactory performance level (step V of Figure 1)? If not, one option may be to return to the training process, changing some of its parameters, as training with different parameters generates different learning and models. This new model may have better performance. This cycle (V, IV, and III of Figure 1) is called **hyperparameter optimization**.

A pathology that can occur during the hyperparameter optimization stage is called *overfitting*, and it is intrinsically linked to the complexity of the generated model. More complex models can better learn the relationship between X and Y, generating better predictions. If the model is too complex, however, this excessive learning can be explained by overadaptation to the data. When this happens, the model will not perform well in predicting outputs whose inputs were not in the training set [Domingos, 2012; Beckmann and Schüssler, 2016; Peng and Lee, 2021]. *Overfitting* is thus an inability to generalize the learning obtained to a dataset never seen before [Pang-Ning Tan, 2005].

When a set of parameters that generates a model with good performance (in accuracy) and does not suffer from *overfitting* is found, the next step is to train it with the entire dataset and implement it for final use (stage VI). This is a roadmap of the macro-steps and best practices to be taken when training classification models. Each macro-step can be performed in various ways. In this work, for example, data separation was performed using the *k-fold* method and not the *holdout*, as illustrated in Figure 6.

2.2 Classification Algorithms

Classification algorithms are supervised machine learning methods that aim to categorize or classify data into different groups or categories, as explained in detail in Subsection 2.1.1. Some examples include logistic regression, SVM, decision trees, random forest, KNN, naive bayes, among others.

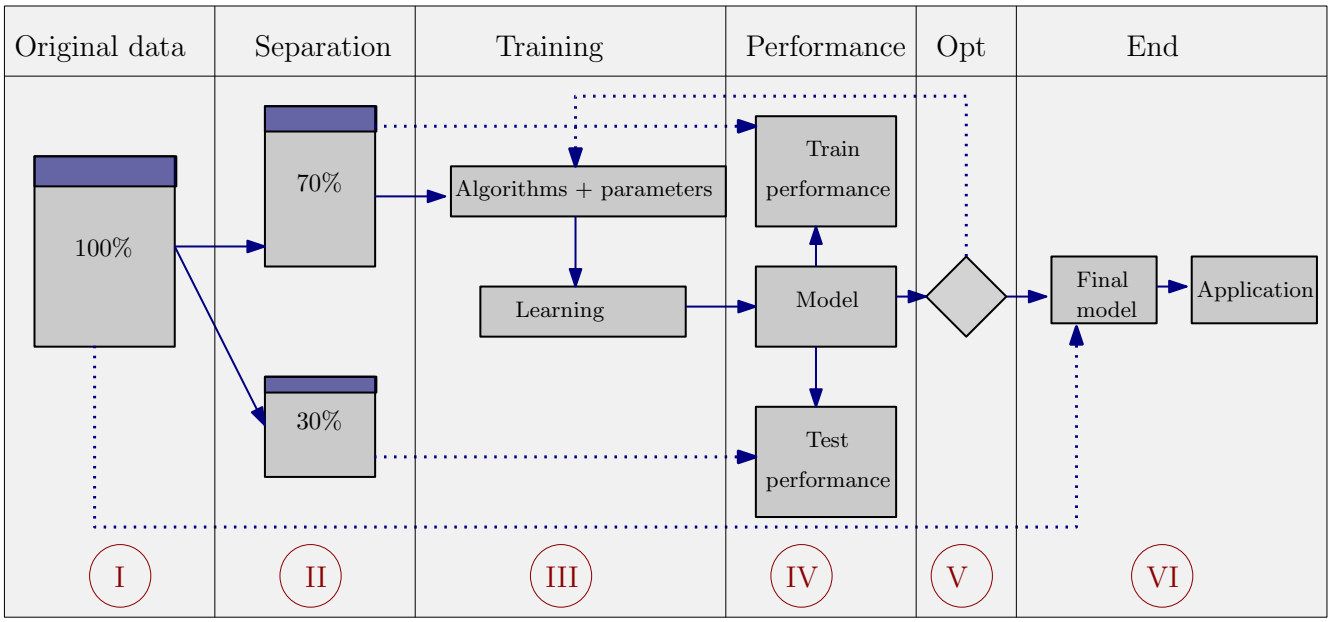


Figure 1. Supervised machine learning process

In the following subsections, the algorithms used in this work are explained.

2.2.1 Decision Trees

Decision trees are commonly created through a recursive process, following a strategy that starts from the top to the bottom where the beginning consists of an attribute that divides the data and then a subdivision occurs considering another attribute [Chen and Hung, 2009]. These attributes that divide the data are called nodes. The three main nodes of a tree are [Pang-Ning Tan, 2005]:

1. Root Node: The root node is the starting point of the decision tree and represents the first decision or feature used to split the data.
2. Internal Nodes: Internal nodes are intermediate nodes that represent intermediate decisions or features used to split the dataset into smaller subsets.
3. Leaf Nodes: Leaf nodes are terminal and represent the final decisions or output classes.

Figure 2 visually exemplifies a decision tree, where: Attribute 01 represents the root node, Attribute 02 is an internal node, and Classes 1 and 2 are the leaf nodes that signify the final classification of the algorithm. Some algorithms can be used to improve the accuracy of a decision tree by selecting the best attributes to partition the data. Some examples include: Hunt, ID3, C4.5, CART [Pang-Ning Tan, 2005].

2.2.2 KNN

The KNN algorithm was proposed in 1967 and can be used as a classification algorithm. This method consists of selecting the k nearest neighbors (considering some distance) regarding the input data, and performing a vote among these elements to estimate the class of the input.

Figure 3 represents how this classification occurs. The light green squares represent one class of data, and the dark

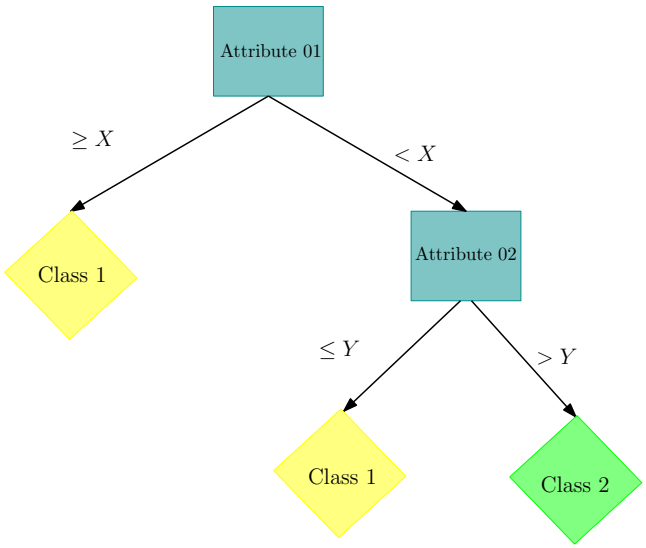


Figure 2. Structure of a decision tree.

green squares represent a second class, while the input data is represented by the yellow square. The first circle around the squares represents the selection of the three nearest neighbors ($k = 3$), and the second circle represents the selection of $k = 7$ neighbors. With $k = 3$, the algorithm would classify the input data with the class of the dark green squares (2 dark green x 1 light green). With $k = 7$, the situation is reversed, and the estimated class would be that of the light green squares (4 light green x 3 dark green). The step of determining the value of k is crucial for the good performance of the algorithm [Zhang, 2022].

A large volume of research related to this selection focuses on the study of distance functions and similarity metrics. The development of techniques aimed at improving KNN classification, especially concerning the selection and construction of distance measures, has become a focus of great interest in research [Hastie and Tibshirani, 1995; Peng *et al.*, 2001; Domeniconi *et al.*, 2002; Vincent and Bengio, 2002; Zhang, 2022].

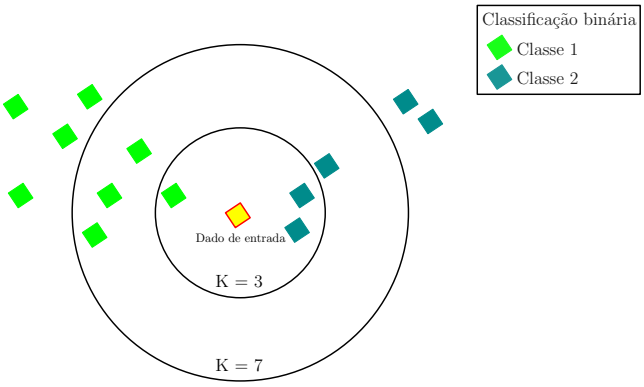


Figure 3. KNN as described in this subsection.

2.2.3 *Support Vector Machine* (SVM)

The idea of SVM was introduced through Vapnik's statistical learning theory in his book *"The Nature of Statistical Learning Theory"* published in 1995 [Sain, 1996]. Classification with this algorithm is based on the principle of achieving the best possible separation between classes. If they are distinguishable, the solution is determined to maximize the separation between them (increasing the distance of the support vectors to the hyperplane, as represented by the dashed arrows in Figure 4) [Nascimento *et al*., 2009]. Figure 4 represents a classification using SVM for data where Y_i has only two distinct values (binary) and is linear.

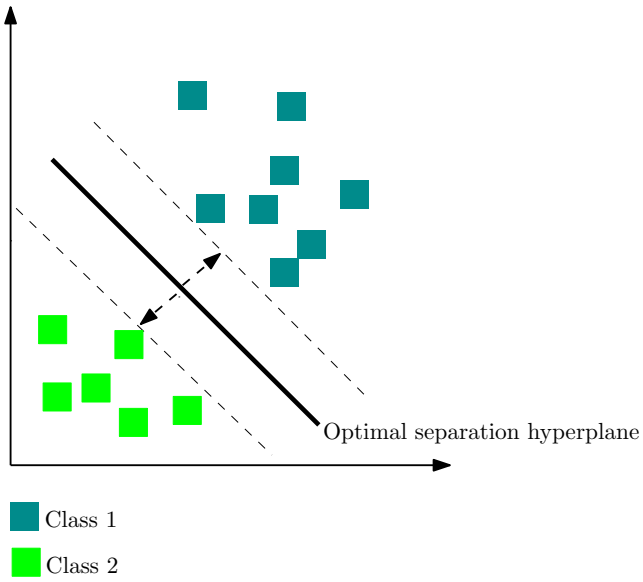


Figure 4. Classification using SVM.

SVM has (i) linear, (ii) quadratic, (iii) polynomial, and (iv) radial basis function kernels to be used in more complex applications.

2.2.4 Logistic Regression

In 1993, logistic regression emerged with the initial proposal to address binary problems. This statistical technique is employed to estimate the probability of an event occurring, ranging between 0 and 1 [de Menezes *et al*., 2017].

Based on linear regression, logistic regression starts as linear and then adds a layer of nonlinear mapping, summing the features linearly and then using the sigmoid function to make predictions [Gai and Zhang, 2023]. The sigmoid function is defined by Equation 2:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

The interpretations of the results from the equation are as follows:

- If Z is greater than 0, then g(Z) will be greater than 0.5, indicating a higher probability of the event occurring (class 1).
- If Z is less than 0, then g(Z) will be less than 0.5, indicating a lower probability of the event occurring (class 0).

2.3 Ensemble of Classifiers

An ensemble of classifiers consists of a predictive model that combines the predictions of a set of individual classifiers to obtain more robust, stable, and accurate results [Yu *et al*., 2019; Cui *et al*., 2021]. The classification of an attribute vector, X, is done by all models in the ensemble (in the case of Figure 5, 3 models). Each model generates a classification, so the final classification is made by the class that occurs most frequently among the models (voting).

As mentioned in Dong *et al*. [2019], ensemble models can be classified as parallel or sequential. Figure 5 represents the execution flow of an ensemble with three classification algorithms in parallel, meaning with independent classifications. In the case of sequential classification, subsequent models are trained to correct the errors of the previous model, and the model predictions are thoughtfully combined to form the final prediction. Among the most popular methods are *Boosting* (sequential) [Freund *et al*., 1996] and *Bagging* (parallel) [Breiman, 1996]. All show consistent improvements in prediction accuracy compared to individual classifiers [Kim *et al*., 2006].

3 Development

In this section, we will describe the dataset used, the machine learning models applied, the functions for hyperparameter optimization, and the creation of the ensemble. We will use Figure 6 to explain each step, as well as the division and use of the data.

3.1 Dataset

The dataset used is available in the Kaggle repository `https://www.kaggle.com/datasets/lkritika/bank-customer-churn-prediction`. The dataset has ten thousand rows and twelve columns, each described in Table 1. The target variable, Y, is "Exited". The goal is to predict whether the customer would leave the institution based on their behavior considering the variables X.

Initially, the data were divided into training and validation sets, 80% and 20%, respectively. The target variable distribution remained similar in all sets, as shown in the charts in Figure 7.

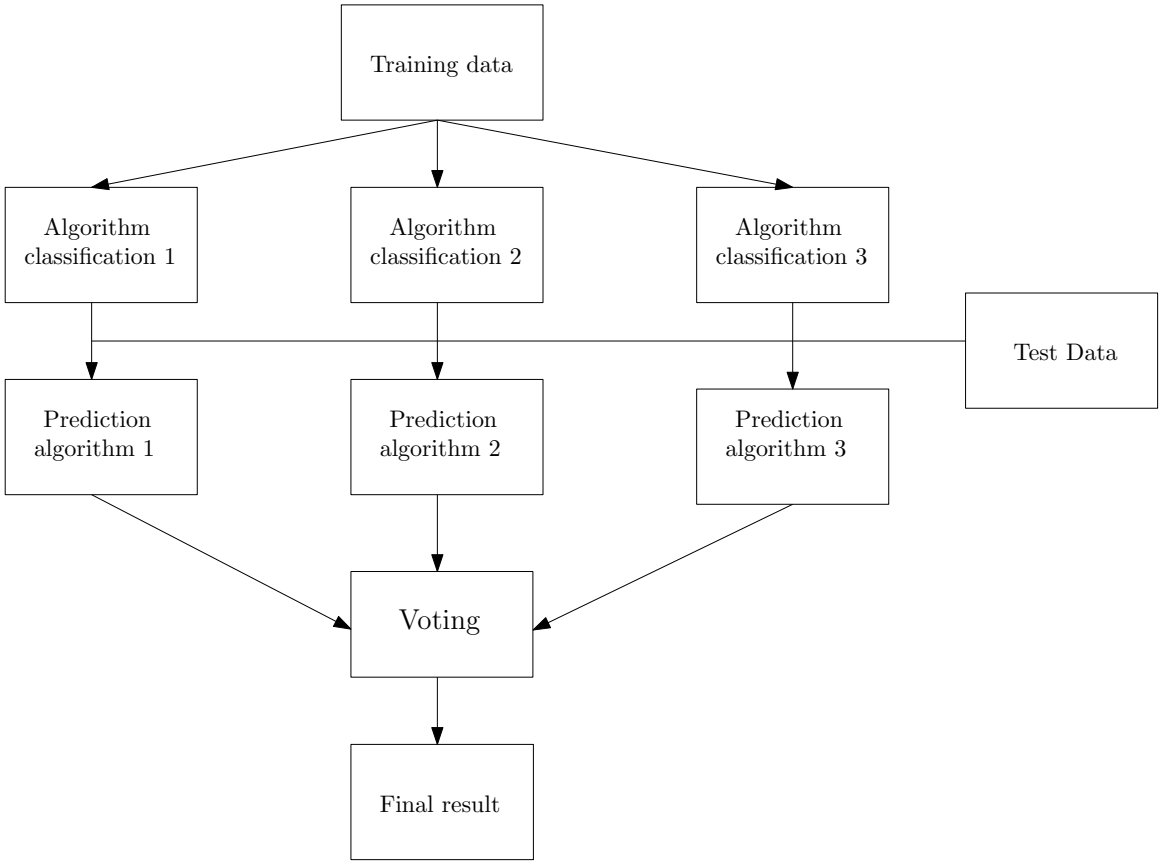


Figure 5. Flow of a classification ensemble

Table 1. Table describing database variables

Column name	Column description	Data type
RowNumber	Database record number	*Integer*
CustomerID	Unique identification code for the customer at the bank	*Integer*
CreditScore	Credit score	*Integer*
Geography	Customer geographic location	*String*
Gender	Customer gender	*String*
Age	Customer age	*Integer*
Tenure	Years of relationship with the bank	*Integer*
Balance	Average customer account balance	*Decimal*
HasCrCard	Does the customer have a credit card? Yes or No	*Boolean*
IsActiveMember	Is the customer active in the organization? Yes or No	*Boolean*
EstimatedSalary	Estimated value of the client's salary	*Decimal*
Exited	Did the customer leave the bank? Yes or No	*Boolean*

3.2 Hyper Parameter Optimization

This stage is represented by "STAGE A" in Figure 6 and aimed to find the best sets of parameters for each model under different optimization functions with the aid of cross-validation (*k-fold cross validation*). The training dataset (80%) was divided into K parts (where $K = 5$) of approximately equal sizes. The model was trained 5 times, each time using $K - 1$ splits as training data and the remaining split as test data. The training accuracy (A_{tr}) and test accuracy (A_{ts}) of the model were calculated as the average of each accuracy obtained in each iteration.

3.2.1 Optimization Functions

As previously described, in this work we proposed 2 new objective functions for the hyperparameter optimization step, which will be compared with the traditional function used by the machine learning community (maximization of test accuracy), here called F_1:

$$\max F_1 = A_{ts} \tag{3}$$

where A_{ts} is the test accuracy.

We developed the new functions based on potential deficiencies that can occur in F_1, and we will illustrate their functionality through the Figure 8a.

The graph shows the evolution of training and testing accuracies (Accuracy axis) for a decision tree model, considering

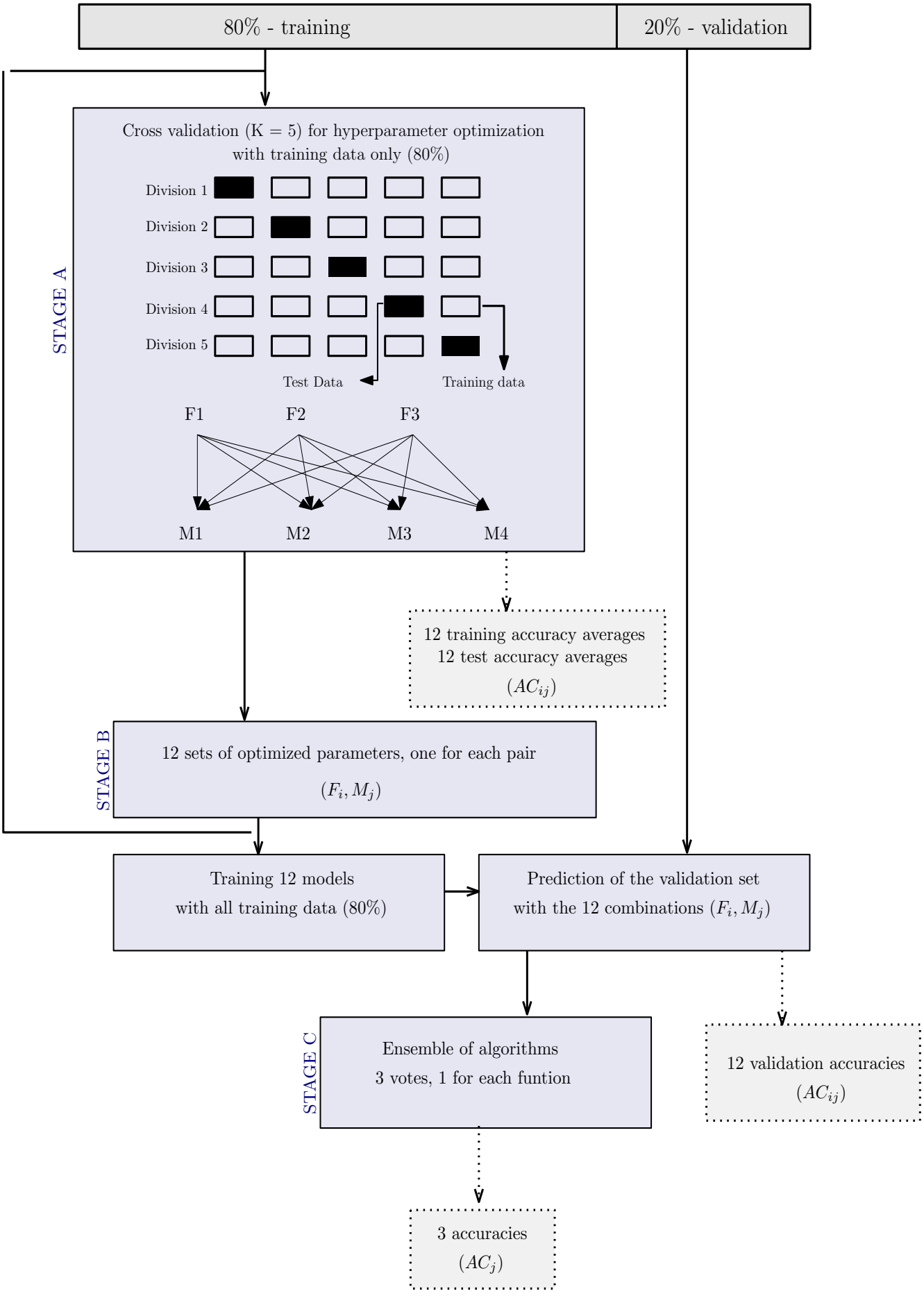


Figure 6. Development flow

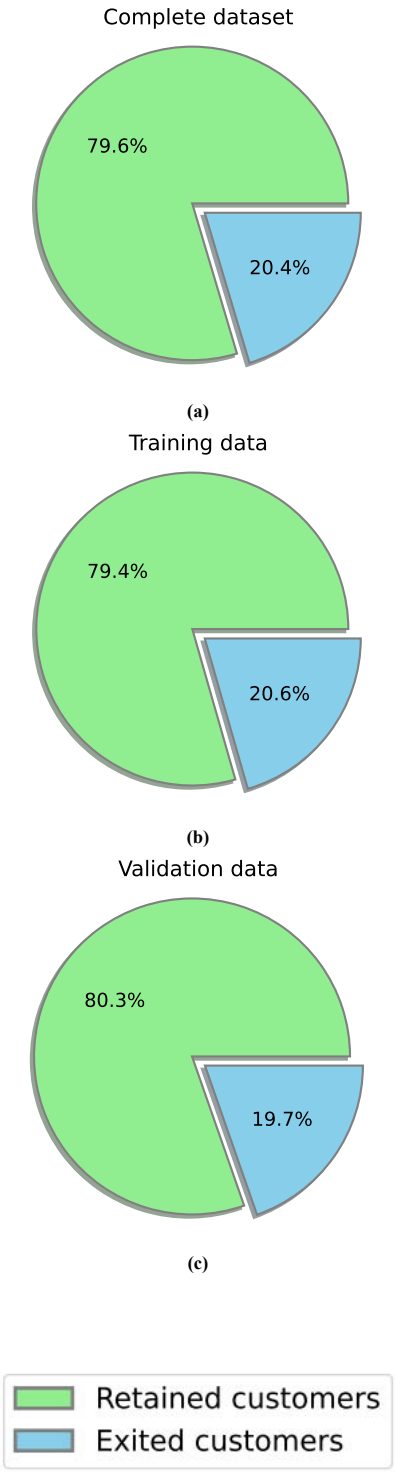


Figure 7. Distribution of the target variable in the different data sets

an increase in the allowed depth of the tree (Depth axis). It is known that the deeper it is, the more adjusted the tree becomes to the data set, consequently closer to *overfitting*. The behavior can be observed by the two curves: as the depth increases, both accuracies also increase, however, there is a point at which the training accuracy reaches 100%, and it is at this moment that we say that the *overfitting*, being reflected in a continuous decrease in test accuracy. As previously stated, this is the expected behavior of overfitted models. Therefore, it makes sense to use the metric of maximizing test accuracy as this is the inflection point of the function and indicates *overfitting*.

However, we want to understand what happens to the metric in cases where the curves do not have this well-defined behavior, for example, in cases where the test curve is somewhat stabilized for a certain period of time before decreasing, as shown in Figure 8b. The maximum test accuracy value may reflect a certain noise in the model training, as the values are already stabilized on average. Therefore, considering only the maximum accuracy can still lead to *overfitting* the model. For this reason, we propose a second evaluation function.

The Function (F_2) considers both test and training accuracy values, as shown in Figure 9a.

The objective is to minimize the absolute value of the difference between accuracies to map the point with the smallest difference between training and testing accuracies. From the example in Figure 9a, the selected point would no longer be the same as F_1, which would already prevent overfitting in models with more stable accuracy behavior (as in Figure 8b).

Let A_{ts} be the test accuracy; function F_2 is then:

$$\min F_2 = |A_{tr} - A_{ts}| \tag{4}$$

To apply this function it was necessary to add the condition:

$$A_{ts} \neq A_{tr} \tag{5}$$

This restriction requires that the data be distinct because if the data for A_{ts} and A_{tr} were the same, this would result in zero differences and the function would not return useful results.

There is still a situation that can affect the performance of F_2, as shown in Figure 9b: if the difference in accuracies is very small, the optimization will give priority to these values, even in cases where the accuracy of testing is low. As shown in Figure, the first difference point would be selected. Therefore, we propose a third optimization function.

The third optimization function (F_3) considers both the difference between accuracies (F_2) and the maximization of test accuracy (F_1). The objective is to find the smallest difference where the test accuracy is maximum. Figure 10 represents the components of function F_3.

That is, a function with two objectives, represented as:

$$(\max F_1) \wedge (\min F_2)$$

To place both objectives in one function, it is enough to invert the sign of one of the components, for example:

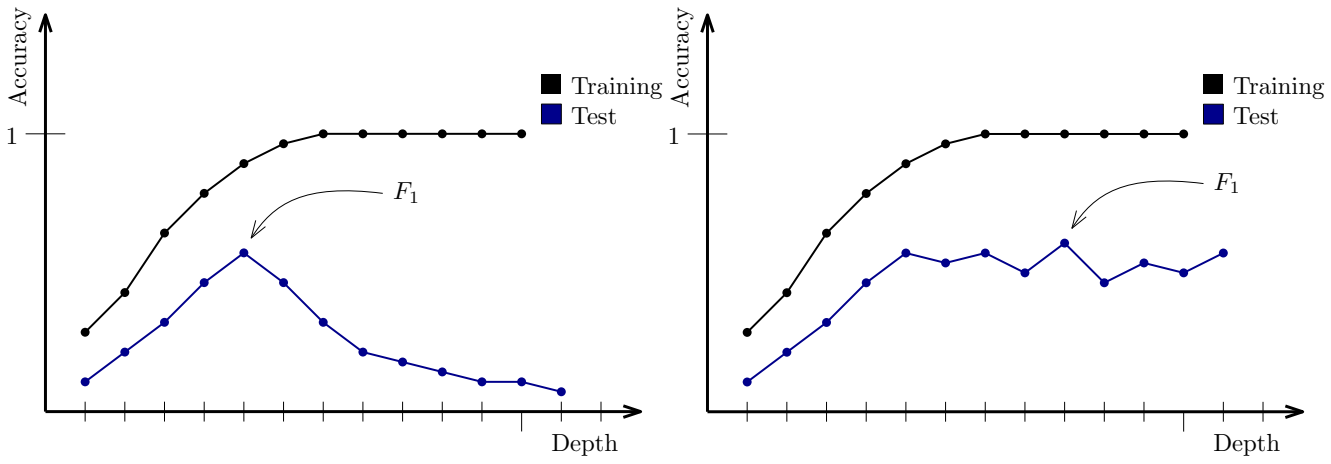
$$\max F_1 - F_2$$

Thus, the smaller F_2 is the better for the maximization function.

Furthermore, we apply a convex combination of elements in order to generalize the function, and subsequently allow the possibility of adapting (and optimizing) the weights given to each component.

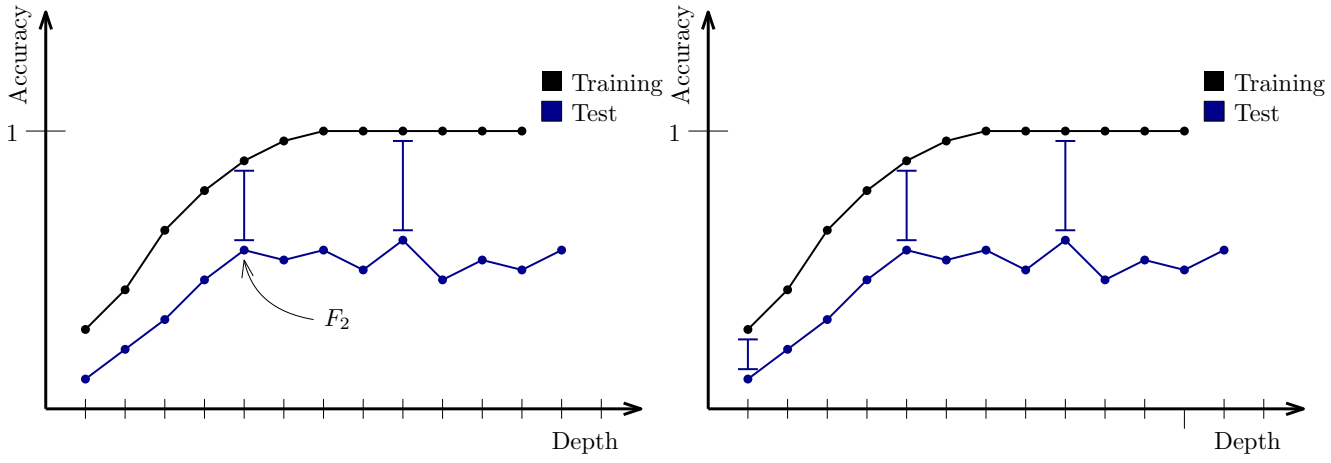
$$\max F_3 = \alpha_1 F_1 - \alpha_2 F_2 \tag{6}$$

with $\alpha_1 + \alpha_2 = 1$.



(a) Traditional **(b)** Unusual

Figure 8. Overfitting possibilities



(a) Function F_2 **(b)** Function F_2 problem

Figure 9. Overfitting possibilities

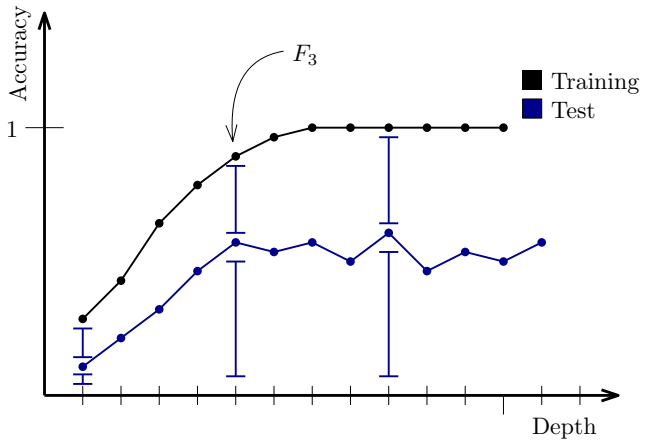


Figure 10. F3 function graph

3.3 Training models with full data (training)

This stage is represented in Figure 6 by "STAGE B". After obtaining the optimized hyper parameter sets for each model and for each function, the training data (same set used in *Cross Validation*) is used for training with the 12 distinct parameter sets obtained.

These sets will be represented by pairs (Function, Model), with the notation (F_i, M_j), where,

$$i \in \{1, 2, 3\}, \quad j \in \{1, 2, 3, 4\} \tag{7}$$

The j components are:

1. Decision tree model
2. KNN model
3. SVM model
4. Logistic regression model

3.4 Validation prediction and ensemble

After training the complete training data, 12 prediction vectors, YP_{ij}, (prediction result for the i-th function and j-th model) were obtained for the validation data (20%). From this, 12 accuracy metrics are extracted for this set, one for each element (F_i, M_j).

The ensemble classification is perform by voting on the *outputs* of the 4 models used. As the data is binary, the criterion of summing the classifications of the models was considered. Thus, the conditions for ensemble classification are:

$$\begin{cases} 1 \text{ if sum} \geq 3 \\ 0 \text{ if sum} < 2 \\ \text{class of the model with the highest accuracy, if sum} = 2 \end{cases}$$

The last case considers a situation of a tie in the voting, so that the model with the greatest accuracy is considered to break the tie. In this way, the three ensemble prediction vectors were constructed and the three final accuracies were extracted.

4 Results

In this Section we present the results of our analysis for the optimization functions, the machine learning models and the created ensemble. The results are organized according to the final accuracies obtained.

The hyperparameter optimization considered *grid-search* (scanning through different parameter combinations). The parameters swept for each model are presented in Table 2.

The optimal parameter sets for each (F_i,M_j) are presented in Tables 3, 4, 5 and 6.

Table 7 was obtained from "STAGE A" of Figure 6 (Subsection 3.2). In the hyperparameter optimization phase, several training and testing accuracies were found, however, to understand the model's ability to assertively interpret new input data, we considered the average of the test data accuracies in Table 7. The values in Table 7 are graphically represented in Figure 11.

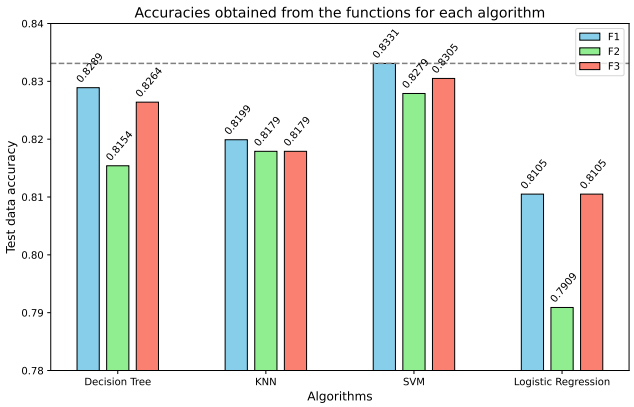


Figure 11. Test data accuracy graph

Note that the blue bars of the accuracy values considering the parameters provided from F_1 reached the highest values for all the analyzed algorithms. Furthermore, it is noted that the highest point on the graph was given by the SVM algorithm with the F_1 parameter set.

The accuracies obtained for the F_2 parameter set (represented by the orange bar) were lower than the others, with three of the four algorithms showing this behavior. Therefore, we can conclude that this function did not surpass the traditional one.

The Table 8 and Figure 12 were constructed based on the results obtained from "STAGE B" and "STAGE C" of Figure 6, both described in Subsection 3.4

Validation data was used as *inputs*. Training with 80% of the data occurred based on the parameters selected by each function, which were later used for prediction. Note that the red bar (F_3) presented superior results in three of the four models, in one of them there was a tie and the highest point on the graph was reached by the "ensemble" using the model

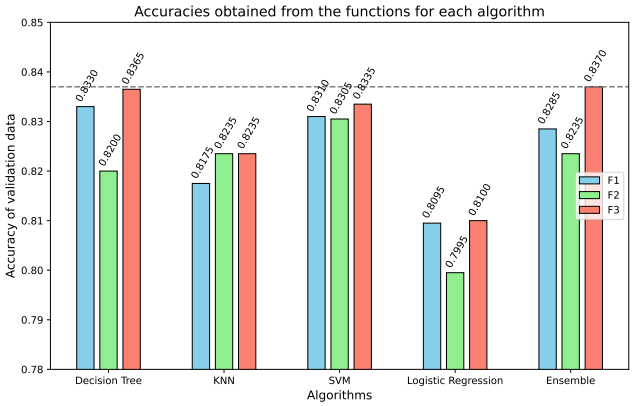


Figure 12. Graph of validation data accuracy

prediction based on the optimization of the parameters using F_3.

Additionally, note that the *ensemble* algorithms showed solid overall performance, with accuracies similar to Decision Tree and SVM. This suggests that combining multiple classifiers can improve performance over using just a single classifier.

Therefore, from Figure 12 we can conclude that the Decision Tree (AD) and SVM models (alone) showed better performance, both with higher values with the parameters provided by F_3. Considering model hybridization, *ensemble* presented the highest value of all accuracies, when individual models were trained using the F_3 function.

5 Discussion

This study highlighted the growing importance of predicting customer abandonment in financial institutions, in a scenario marked by rapid technological evolution and intensified competition. Predicting customer actions allows proactive actions to be taken, aiming to avoid a breakdown in the customer/bank relationship.

To do this, we use the subcategory of Artificial Intelligence: Machine Learning. Four algorithms were used, namely: Decision Trees, Nearest Neighbor, *Support Vector Machine* and Logistic Regression. The models were trained and tested with the *Cross Validate* method, and then the accuracies of the validation data were removed, with the set of parameters chosen according to the applied optimization function. Finally, the *ensemble* using the four algorithms was created with the prediction vector Y obtained with the validation data and the accuracies were compared.

The results of the comparative study between different classification models for the account abandonment problem provided *insights* that can help companies. We observed that certain algorithms demonstrated better performance in predicting customer abandonment, which can be attributed to their specific capabilities in dealing with the characteristics of the analyzed data. Furthermore, the specific contributions of this work, such as the creation of an ensemble of classifiers and the development of new functions for hyperparameter optimization, offer a different approach that has shown promise in dealing with this challenge. Therefore, it is noted that the application of the *ensemble* strategy together with training using F_3 surpassed the accuracy of all other algorithms and

Model	Parameter	Values swept
Decision tree	*criterion*	“gini”, “entropy”
	splitter	“best”, “random”
	min_samples	2, 3, 4
	max_depth	1 até 19
KNN	*n_neighbors*	3, 5, 7, 9, 11, 13, 15
	metric	”euclidean”, ”manhattan”, ”minkowski”
	weights	”uniform”, ”distance”
SVM	*kernel*	”linear”, ”poly”, ”rbf”, ”sigmoid”
	decision_function_shape	”ovo”, ”ovr”
	C	1, 2, 3
Logistic Regression	*penalty*	”l1”, ”l2”
	max_iter	100, 200, 300, 400, 500, 600, 700, 800, 900, 1000
	C	0.001, 0.01, 0.1, 1, 10, 100

Table 2. Parameter relationship

Function	Parameters used			
	criterion	**splitter**	**min_ss**	**max_depth**
F_1	gini	best	2	4
F_2	gini	random	2	3
F_3	gini	best	2	3

Table 3. Parameters used **decision tree**

Function	Parameters used		
	n_neighbors	**weights**	**metric**
F_1	15	distance	euclidean
F_2	15	uniform	manhattan
F_3	15	uniform	manhattan

Table 4. Parameters used **[nearest neighbor]**

Function	Parameters used		
	C	**kernel**	**decision_fs**
F_1	3	rbf	ovo
F_2	3	poly	ovo
F_3	1	rbf	ovo

Table 5. Parameters used **SVM**

Function	Parameters used		
	max_iter	**C**	**penalty**
F_1	200	1.000	l1
F_2	600	0.001	l1
F_3	800	10.000	l1

Table 6. Parameters used **Logistic Regression**

functions in all situations analyzed.

For future research, it is recommended to use F_3 for training the hyper parameters and in conjunction with Ensemble to facilitate the possibility of the highest possible accuracy value. Considering other parameter selection functions, the SVM or AD models are also suggested for this classification category.

Table 7. Test accuracies obtained from *cross validate*

Algorithm	F1	F2	F3
Decision tree	0.8289*	0.8154	0.8264
Nearest Neighbor	0.8199*	0.8179	0.8179
SVM	0.8331*	0.8279	0.8305
Logistic Regression	0.8105*	0.7909	0.8105*

Table 8. Validation data accuracies

Algorithm	F1	F2	F3
Decision tree	0.8330	0.8200	0.8365*
Nearest Neighbor	0.8175	0.8235*	0.8235*
SVM	0.8310	0.8305	0.8335*
Logistic Regression	0.8095	0.7995	0.8100*
Ensemble	0.8285	0.8235	0.8370*

References

Banco Central do Brasil (2018). *Relatório de Economia Bancária 2018*.

Beckmann, J. and Schüssler, R. (2016). Forecasting exchange rates under parameter and model uncertainty. *Journal of International Money and Finance*, 60:267–288. DOI: https://doi.org/10.1016/j.jimonfin.2015.07.001.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24:123–140.

Chen, Y.-L. and Hung, L. T.-H. (2009). Using decision trees to summarize associative classification rules. *Expert Systems with Applications*, 36(2, Part 1):2338–2351. DOI: https://doi.org/10.1016/j.eswa.2007.12.031.

Cui, S., Wang, Y., Yin, Y., Cheng, T., Wang, D., and Zhai, M. (2021). A cluster-based intelligence ensemble learning method for classification problems. *Information Sciences*, 560:386–409. DOI: https://doi.org/10.1016/j.ins.2021.01.061.

de Lima Lemos, R.A.; Silva, T. and Tabak, B. (2022). Propension to customer churn in a financial institution: a machine learning approach. *Neural Comput & Applic*.

de Menezes, F. S., Liska, G. R., Cirillo, M. A., and Vivanco, M. J. (2017). Data classification with binary response through the boosting algorithm and logistic regression. *Expert Systems with Applications*, 69:62–73. DOI: https://doi.org/10.1016/j.eswa.2016.08.014.

Domeniconi, C., Peng, J., and Gunopulos, D. (2002). Locally adaptive metric nearest-neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285. DOI: 10.1109/TPAMI.2002.1033219.

Domingos, P. (2012). A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87. DOI: 10.1145/2347736.2347755.

Dong, X., Yu, Z., Cao, W., Shi, Y., and Ma, Q. (2019). A survey on ensemble learning. *Frontiers of Computer Science*, 14:241 – 258.

Freund, Y., Schapire, R. E., *et al*. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Gai, R. and Zhang, H. (2023). Prediction model of agricultural water quality based on optimized logistic regression algorithm. *EURASIP Journal on Advances in Signal Processing*, 2023. DOI: 10.1186/s13634-023-00973-9.

Hastie, T. J. and Tibshirani, R. (1995). Discriminant adaptive nearest neighbor classification and regression. In *Neural Information Processing Systems*.

Kim, Y., Street, W. N., and Menczer, F. (2006). Optimal ensemble construction via meta-evolutionary ensembles. *Expert Systems with Applications*, 30(4):705–714. DOI: https://doi.org/10.1016/j.eswa.2005.07.030.

Lagasio, V., Pampurini, F., Pezzola, A., and Quaranta, A. G. (2022). Assessing bank default determinants via machine learning. *Information Sciences*, 618:87–97. DOI: https://doi.org/10.1016/j.ins.2022.10.128.

Li, J.-P., Mirza, N., Rahat, B., and Xiong, D. (2020). Machine learning and credit ratings prediction in the age of fourth industrial revolution. *Technological Forecasting and Social Change*, 161:120309. DOI: https://doi.org/10.1016/j.techfore.2020.120309.

Michelucci, U., Sperti, M., Piga, D., Venturini, F., and Deriu, M. A. (2021). A model-agnostic algorithm for bayes error determination in binary classification. *Algorithms*, 14(11):301.

Mienye, I. D. and Sun, Y. (2022). A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access*, 10:99129–99149. DOI: 10.1109/ACCESS.2022.3207287.

Nascimento, R. F. F., Alcântara, E., Kampel, M., Stech, J. L., Moraes Novo, E., and Fonseca, L. M. G. (2009). O algoritmo support vector machines (svm): avaliação da separação ótima de classes em imagens ccd-cbers-2. *Anais do XIV Simpósio Brasileiro de Sensoriamento Remoto [CDROM]*, pages 25–30.

Norvig, P. and Russell, S. (2013). *Inteligência Artificial*. ELSEVIER EDITORA.

Pang-Ning Tan, Michael Steinbach, V. K. (2005). *Introduction to Data Mining*. Pearson.

Peng, J., Heisterkamp, D., and Dai, H. (2001). Lda/svm driven nearest neighbor classification. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I. DOI: 10.1109/CVPR.2001.990456.

Peng, Y.-L. and Lee, W.-P. (2021). Data selection to avoid overfitting for foreign exchange intraday trading with machine learning. *Applied Soft Computing*, 108:107461. DOI: https://doi.org/10.1016/j.asoc.2021.107461.

Sain, S. R. (1996). The nature of statistical learning theory. *Technometrics*, 38(4):409–409. DOI: 10.1080/00401706.1996.10484565.

Sharma, A. and Panigrahi, D. (2013). A neural network based approach for predicting customer churn in cellular network services. *International Journal of Computer Applications*, 27. DOI: 10.5120/3344-4605.

Vincent, P. and Bengio, Y. (2002). K-local hyperplane and convex distance nearest neighbor algorithms. *Advances in Neural Information Processing Systems*.

Xiao, J., Xiao, Y., Huang, A., Liu, D., and Wang, S. (2014). Feature-selection-based dynamic transfer ensemble model for customer churn prediction. *Knowledge and Information Systems*, 43:29–51. DOI: 10.1007/s10115-013-0722-y.

Yu, Z., Wang, D., Zhao, Z., Chen, C. L. P., You, J. J., Wong, H.-S., and Zhang, J. (2019). Hybrid incremental ensemble learning for noisy real-world data classification. *IEEE Transactions on Cybernetics*, 49:403–416.

Zhang, S. (2022). Challenges in knn classification. *IEEE Transactions on Knowledge and Data Engineering*, 34(10):4663–4675. DOI: 10.1109/TKDE.2021.3049250.