XLIII CILAMCE
21–25, November, 2022
FOZ DO IGUAÇU – PR – BRAZIL

Identification of Text Relevance in Service Desk Systems using Machine Learning Classifiers

Marciel M. Degasperi[1], Daniel C. Cavalieri[1], Fidelis Z. Castro[1]

[1]*Dept. of Control and Automation Engineering, Federal Institute of Education, Science and Technology of Espírito Santo at Serra*
Morada de Laranjeiras, Serra, 29166-630, Espirito Santo, Brazil
marciel.deg@gmail.com, daniel.cavalieri@ifes.edu.br, fidelis@ifes.edu.br

Abstract. Service Desk systems form a wide source of useful information for organizations, which consists of historical support requests. Such information can serve as a reference for responding future requests. Standardized search tools, such as keyword searches in support request histories, are infeasible in large datasets and may provide answers unrelated to a problem of interest. This manuscript aims to compare the performance of machine learning algorithms in classifying support requests as relevant or not. We define as relevant the support requests that have the potential to serve as a basis for responding to others. We will develop a filter to remove non-relevant information from the dataset of historical support requests to provide a finite low-cardinality set of recommendations for future support and assistance. In the performed tests, Naive-Bayes, Adaptive Boosting, Random Forest, Stochastic Gradient Descent, Logistic Regression, Support Vector Machine, and Light Gradient Boosting Machine classifiers were used. The classifier with the best performance (Random Forest) presented maximum average accuracy close to 80%, and recall, F1-score, and AUC values all greater than 80%.

Keywords: Machine Learning, Natural Language Processing, Service Desk Systems, Classification.

1 Introduction

The Service Desk systems act as a link between companies and customers through a system that enables the compilation and resolution of problems, as well as records of the solution presented and correlation with other situations [1]. With this, an extensive database is created for the company, managing problems, solving them at their root, and reducing operational costs. Thus, Service Desk systems can centralize various information for several service areas, becoming a pivotal point in administration and problem solving [2].

With the growth of the service base over time, the search for information becomes complex. A simple way to get a set of responses to a user query is to determine which documents in a collection contain a particular set of query keywords. However, this may not be enough to satisfy the user, as the presence of non-relevant documents among the documents returned by a query is practically inevitable. In this scenario, the main objective of these systems should be to retrieve as many relevant documents as possible and as few non-relevant documents as possible [3]. In this scenario, there is a need to use support tools to analyze attendances to help classify and recover information in the data set. As these data are basically composed of descriptive texts, there is a need to propose and implement computational systems based on natural language processing.

A characteristic of this type of system is that not all the calls made have information that can be reused to assist in handling new calls. This data harms search algorithms, whether a simple keyword search algorithm or complex algorithms with neural networks. This work aims to test the sensitivity of classical classification algorithms in identifying the ”relevance” of texts, that is, to identify which texts have knowledge that can be reused. The motivation is that irrelevant services can be previously discarded. Thus, a new dataset can be formed, with only relevant calls, with a smaller volume of data, allowing complex data extraction algorithms to work on a reduced portion of the data, reducing its computational cost.

2 Brief Literature Review

2.1 Vectorization Algorithms

While capable of performing many multimedia tasks, computers are still machines executing instructions about numbers in the binary system. Thus, to enable the classification of texts with computational resources, it is necessary to select strategies that can represent sets of words in the document in numerical form without losing essential characteristics. These strategies are encapsulated in vectorization algorithms [4].

Term Frequency times Inverse Document Frequency (TF-IDF)

One of the procedures generally adopted for text vectorization is the representation using the *bag-of-words* approach. In this approach, each document is represented as a frequency vector of words that occur in the document. When the term is present in the document, the value is 1; otherwise, 0 [5].

A common way of categorizing texts is by identifying the occurrence of keywords that characterize documents on a specific topic. For example, baseball articles would tend to have many occurrences of words like "ball", "bat", "throw", "run", and so on. The algorithm *Term Frequency times Inverse Document Frequency*, or simply TF-IDF, is a statistical measure that indicates the importance of a word concerning the document [5]. Thus, it is possible to replace each word of a text with a number that indicates its importance in the document context.

2.2 Classification Algorithms

Classification algorithms are methods used to identify new classes of observations based on training data. Algorithms learn from datasets or observations and then classify new observations into various classes or groups [6]. We present a brief review on the classification algorithms used in this manuscript below.

Naive-Bayes

In probability and statistics, Bayes' theorem describes the probability of an event based on a priori knowledge that may be related to the event. Naive-Bayes classifier is a direct application of Bayes' Theorem. The term "naive" refers to the central premise of the algorithm that the considered attributes are uncorrelated with each other [7].

In text classification, the *Naive-Bayes* classifier is a popular method due to its computational efficiency and good performance in prediction [8]. In this classifier, each document is seen as a collection of words, where the occurrence position of each word is not considered. Because it is very simple and fast, it has a relatively higher performance than other classifiers. Also, Naive-Bayes only needs a small number of test data to complete classifications with reasonable accuracy [9].

Adaptive Boosting

"*Boosting*" is a technique proposed in the 90's, where several inaccurate classifiers are combined to produce a robust new algorithm. Among the various existing boosting implementations, *Adaptive Boosting*, or *AdaBoost*, is the most commonly used [10].

The basic concept behind *AdaBoost* is to define classifier weights and train the data sample at each iteration in a way that guarantees accurate predictions of unusual observations. Each iteration tries to provide an optimal fit for these examples, minimizing the training error so that the next iteration does a more accurate classification [6].

Random Forest

A decision tree is a predictor that predicts the label associated with an instance traveling from a tree's root node to a leaf. At each node on the path from the root to the leaf, the successor child is chosen based on the division of the input space. Typically, the split is based on one of the characteristics or a predefined set of rules [11]. Decision trees can be used to discover features and extract patterns in large databases important for discrimination and predictive modeling. These characteristics and their intuitive interpretation are why decision trees are used extensively for exploratory data analysis [12].

In order to minimize the known limitations on Decision Trees, *Random Forest* was proposed. It is a combination of decision trees, so each tree is built in a random subspace of the feature space. Trees in different subspaces generalize their classification in a complementary way, and their combined classification is better than the individual classification [13].

Stochastic Gradient Descent

In mathematics, "*Gradient Descending*" is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. The idea is to take repeated steps in the opposite direction of the function's gradient at the current point, as this is the steepest descent direction [14].

The *Stochastic Gradient Descent*, often abbreviated as SGD, can be considered a stochastic approximation of gradient descent optimization. This algorithm replaces the actual gradient (calculated from the entire dataset) with an estimate of the same (calculated from a randomly selected subset of data), thus reducing computational load [15].

Logistic Regression

Logistic regression is a supervised classification algorithm. In a classification problem, the output variable can only take on discrete values for a given set of inputs. The logistic regression model builds a regression model to predict the probability that a given data input belongs to the category numbered 1. Just as linear regression assumes that the data follow a linear function, logistic regression models the data using the sigmoid function [16].

Logistic regression is a statistical technique that aims to produce, from a set of data, a model that allows the prediction of values taken by a categorical variable from a series of explanatory variables [17].

Support Vector Machine

The *Support Vector Machine* classifier, or SVM, takes a set of data as input and predicts, for each given input, which of two possible classes the input belongs to. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or another [18].

An SVM model is a representation of examples as points in space, mapped in such a way that the examples in each category are divided by a clear space that is as wide as possible. The new examples are then mapped to the same space and predicted to belong to a category based on which side of the space they are placed [11].

Light Gradient Boosting Machine

Gradient Boosting Decision Tree is a popular machine learning algorithm with some effective implementations. However, its efficiency and scalability are still unsatisfactory for large datasets.

The purpose of *LightGBM* is to reduce the number of data instances and the number of resources, thus speeding up the training process. For this, two techniques were created: Gradient-based One-Side Sampling (GOSS), which proposes to reduce the volume of data by performing random sampling on instances with small gradients while keeping all instances with large gradients, and the Exclusive Feature Bundling (EFB), which bundles exclusive features. With these approaches, LightGBM achieves results up to 20 times faster than its predecessor, with virtually the same accuracy. [19].

2.3 Evaluation Metrics

Classifiers are generally trained to minimize errors. This error is evaluated using one or several metrics, the choice of which has been an ongoing debate in research and the industry for several decades [20].

Among the metrics used in evaluating classifiers, we can mention Accuracy, Precision, Recall, F1-score, ROC, and AUC. Accuracy is defined as the proportion of true instances retrieved, both positive and negative, among all retrieved instances. Precision measures the number of correct instances retrieved divided by all instances retrieved. The Recall measures the number of correct instances retrieved divided by all correct instances. The F1-score is the harmonic mean between precision and recall [21]. ROC (Receiver Operating Characteristics Curve) plots are two-dimensional plots that display the relationship between true positive and false positive instances. Finally, the AUC (Area Under Curve) is the area under the ROC curve, a numerical representation of the same indicator [22].

2.4 Cross-Validation

When deploying a predictive model, it is vital to understand its prediction accuracy in future tests. Therefore, the results must reflect reasonable estimates and accurate confidence intervals. Cross-validation is a widely used approach for these two tasks [23].

Cross-validation operates by splitting the dataset into two segments: one used to train and the other used to validate the model. The training and validation sets must be crossed over in successive rounds so that each data point has a chance to be validated. The basic form of cross-validation is k-fold cross-validation. First, the data is partitioned into k segments or similarly sized folds. Then k training and validation iterations are performed such that within each iteration a different fold of the data is kept for validation. At the same time, the remaining $k - 1$ folds are used for learning. Upon completion, k samples of the performance metric will be available for each algorithm. Different methodologies, such as averaging, can be used to obtain an aggregate measure of these samples, or these samples can be used in a statistical hypothesis test to show that one algorithm is superior to another [24].

3 Methodology

3.1 Development Tools

In this article, we used the Python programming language in the Google Collaboratory development environment. We also use implementations of the sklearn package, which provides a set of tools for machine learning and statistical modelings such as classification, regression, clustering, and dimensionality reduction.

3.2 Dataset

The database to be used, entirely in Portuguese, was collected from a relational database of the *Conexos Help Desk* system, internal software of the company *Conexos Consultoria e Sistemas LTDA*. This system has been operating since 2004, containing a considerable collection of services the company provides to these customers. The flow of interactions in this system is similar to the flow of online forums: from a customer request, which can be a question, a problem report, or a request for an increase in the system, there may be several responses from the provider of services, as well as several replicas and rejoinders until the customer has the request resolved, at which time the service is completed. This format allows the generation of calls with complex flows, where oscillations between different subjects often occur, making their classification difficult. Thus, for tests, only calls were selected where the flow of care is composed of a question from the customer, a response from the service provider, and the end of the service. Given these conditions, data were collected from the consultations carried out between 2019 and 2020, totaling 11449 consultations. Following the Brazilian guidelines of the General Data Protection Law (*Lei Geral de Proteção de Dados - LGPD*), the database was anonymized to make it impossible to identify confidential data.

Table 1 brings some statistical data on the words that make up the dataset. Among the notable characteristics, there is a strong occurrence of repetition of words in the texts.

Table 1. Dataset metrics

Indicator	Value
Total of Calls	11449
Total of Words	953728
Average of Words per Call	83.30
Distinct Words	29408
Average of Distinct Words per Call	28.90

Word Cloud is an intuitive tool that allows a quick analysis of word occurrences in a document. A Word Cloud is a visual representation of the frequency of words, so the more commonly the term appears within the text being analyzed, the larger the word will appear in the generated image [25]. Figure 1 shows the Word Cloud for the dataset.

3.3 Text Preprocessing

Text preprocessing is an essential step in any Natural Language Processing System. This step involves its transformation before its analysis. It includes strategies such as removing irrelevant content for the task, clustering semantically related terms, and increasing the amount of semantic information captured [26].

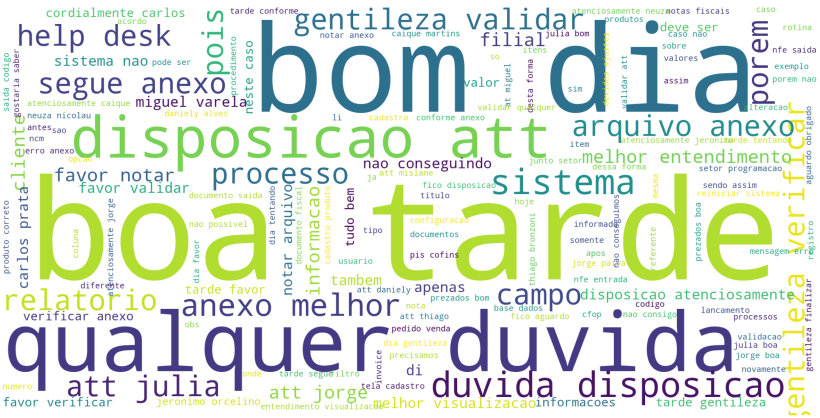

Figure 1. Dataset Word Cloud.

In Figure 2, the Word Cloud is represented after the removal of greetings and standard introductory and closing terms. We can notice a better distribution of occurrences in terms with contextual meaning, such as the words “processo”, “sistema”, “relatório”, and others.

Figure 2. Dataset Word Cloud after data pre-processing.

3.4 Training dataset

For training the classification algorithms, a portion of data must be previously classified, so the algorithms can be based during training and validate the obtained results. For this, a subset of 1084 attendances was classified to compose the training dataset. The unbalanced distribution between classes in the dataset makes the classifier models tend towards classes with many training samples [27]. This way, the calls were selected to compose a perfectly balanced dataset.

Table 2 shows statistical information about the subset for training. A notable feature is that, on average, relevant calls have a more significant number of words than non-relevant calls.

Table 2. Descriptive information of the training dataset.

Indicator	Total	Positives	Negatives
Number of Calls	1084	542	542
Total Words	88244	52347	35897
Words per Service	81.41	96.58	66.23
Distinct Words	8355	6167	5194
Distinct Words per Call	28.92	29.88	27.97

4 Results

A subset of the original dataset was used for the tests, composed of 1084 manually classified attendances. This subset was carefully balanced, with 542 services classified as relevant and the others as not relevant. This decision was taken because there are known problems of some classification algorithms with unbalanced datasets, as can be seen in the works of Chawla and Sylvester [28], Pozzolo et al. [29], Sundarkumar and Ravi [30], and others. The TF-IDF vectorizer and a series of classic classifiers were used in the tests, further detailed in the next step.

The results will be presented below using the classifiers Adaptive Boosting, Light Gradient Boosting Machine, Logistic Regression, Naive-Bayes, Random Forest, Stochastic Gradient Descent, and Support Vector Machine. Table 3 shows the classifiers' metrics values, obtained through cross-validation experiments with $k = 10$, performed on the data set.

Table 3. Classic classifier metrics when applied to the dataset.

Classifier	Accuracy	Precision	Recall	F1-Score	AUC
AdaBoost	0.747 ± 0.044	0.758 ± 0.049	0.733 ± 0.074	0.742 ± 0.048	0.824 ± 0.033
LightGBM	0.791 ± 0.025	0.785 ± 0.043	0.806 ± 0.053	0.794 ± 0.024	0.867 ± 0.036
Logistic Regression	0.789 ± 0.021	**0.785 ± 0.029**	0.799 ± 0.052	0.790 ± 0.024	**0.872 ± 0.030**
Naive-Bayes	0.767 ± 0.034	0.746 ± 0.035	0.812 ± 0.067	0.776 ± 0.037	0.861 ± 0.028
Random Forest	**0.793 ± 0.036**	0.768 ± 0.046	**0.847 ± 0.043**	**0.804 ± 0.031**	0.871 ± 0.035
SGD	0.781 ± 0.026	0.771 ± 0.043	0.808 ± 0.053	0.787 ± 0.024	0.858 ± 0.026
SVM	0.788 ± 0.018	0.781 ± 0.044	0.808 ± 0.044	0.792 ± 0.015	0.864 ± 0.030

The experiment's classifiers presented similar metrics, emphasizing the *Random Forest* classifier that obtained the best results, reaching values of accuracy and F1-Score of 0.793 and 0.804, respectively.

Figure 3 shows the ROC curves of the algorithms used. In the same way as the other metrics, the data were collected in cross-validation, with $k = 10$.

5 Conclusions

The definition of content "relevance", despite the apparent simplicity behind binary classification, is an abstract concept. The initial analysis of the dataset showed that the characteristics that identify relevance are subtle and involve more than just the presence or not of keywords in the text.

Some classical classifiers were applied to identify the concept of "relevance" of the dataset texts. The classical classification algorithms applied showed an efficiency superior to 70% for identifying the relevant context. The Random Forest and LightGBM classifiers stood out in the classification. The Random Forest classifier presented accuracy and F1-score values of 0.793 and 0.804, respectively, while the LightGBM classifier presented values for the same statistics of 0.791 and 0.794, respectively.

The high variance between runs, evidenced by the high value of standard deviation identified in the cross-validation, can indicate two distinct scenarios: (1) the dataset used for training is insufficient, or (2) the classifiers are inadequate to process the data in the context of the problem. To validate scenario 1, the training dataset will be increased, and new tests will be executed. For scenario 2, tests must be carried out using other classifiers that can be sensitive to other text characteristics.

For the task of identifying the relevance of texts, other computational strategies must be studied. In future works, we suggest using recurrent neural networks and deep learning, such as Word2vec and LSTM networks.

Acknowledgements. This research was supported/partially supported by Conexos Consultoria e Sistemas LTDA, which provided access to the data used in this work. We thank our teachers and colleagues from Federal Institute of Education, Science and Technology of Espírito Santo (Ifes) who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

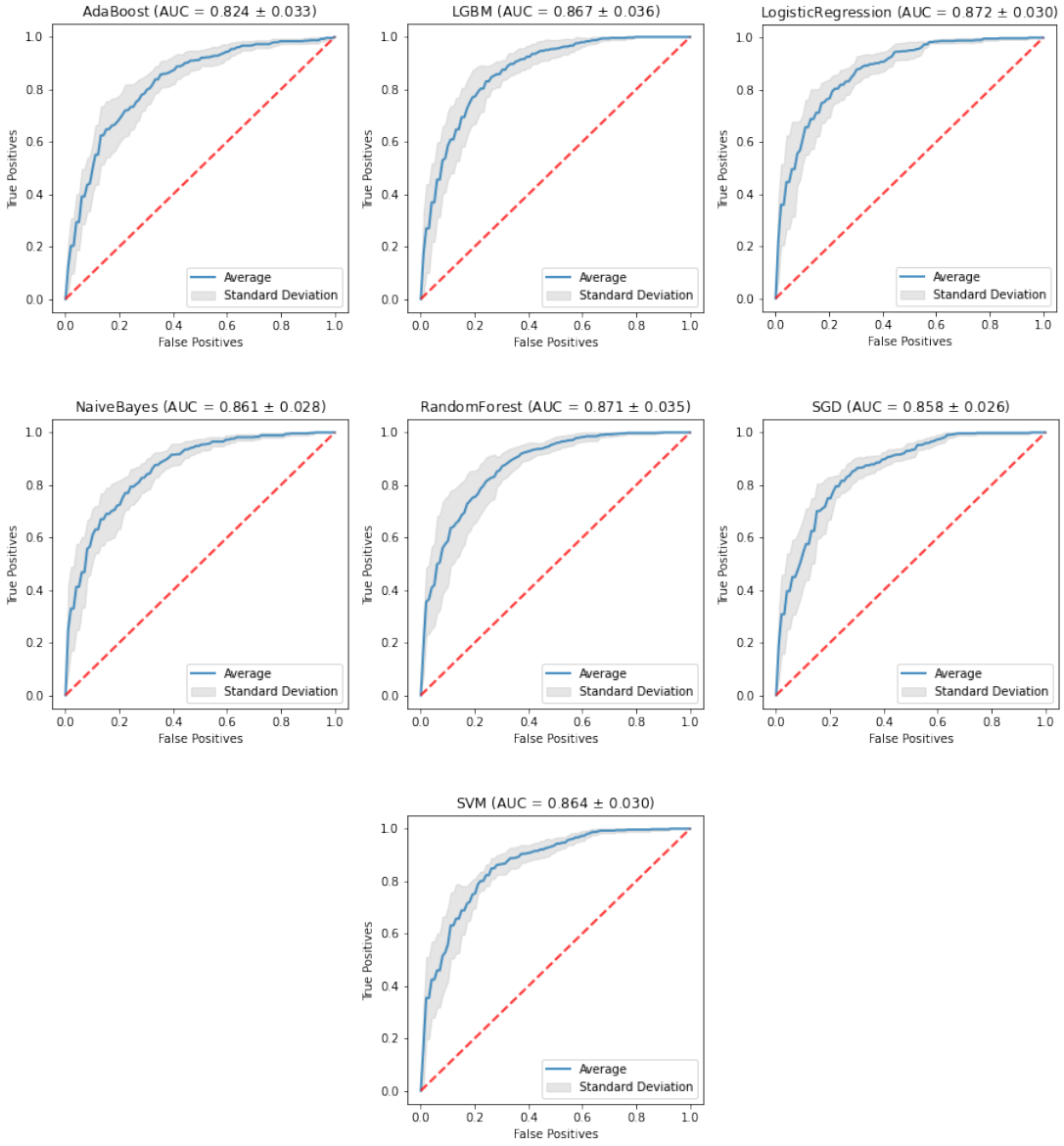

Figure 3. ROC curves of classifiers with standard deviation.

References

[1] V. G. Boscolo. Sistema de gerenciamento de Help-Desk, 2009.

[2] G. O. T. Cavalari and H. A. X. Costa. Modelagem e desenvolvimento de um sistema Help-Desk para a prefeitura municipal de Lavras. *Revista Eletrônica de Sistemas de Informação*, vol. 4, n. 2, 2005.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Recuperação de Informação: Conceitos e Tecnologia das Máquinas de Busca*. Bookman Editora, 2013.

[4] A. K. Singh and M. Shashi. Vectorization of text documents for identifying unifiable news articles. *Int. J. Adv. Comput. Sci. Appl*, vol. 10, n. 7, 2019.

[5] A. Rajaraman and J. D. Ullman. *Data Mining*, pp. 1–17. Cambridge University Press, 2011.

[6] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown. Text classification algorithms: A survey. *Information*, vol. 10, n. 4, pp. 150, 2019.

[7] R. Swinburne. Bayes' theorem. *Revue Philosophique de la France Et de l*, vol. 194, n. 2, 2004.

[8] J. Chen, H. Huang, S. Tian, and Y. Qu. Feature selection for text classification with Naïve Bayes. *Expert Systems with Applications*, vol. 36, n. 3, Part 1, pp. 5432–5435, 2009.

[9] E. Frank and R. R. Bouckaert. Naive Bayes for text classification with unbalanced classes. In J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, eds, *Knowledge Discovery in Databases: PKDD 2006*, pp. 503–510, Berlin,

Heidelberg. Springer Berlin Heidelberg, 2006.
[10] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors). *The Annals of Statistics*, vol. 28, n. 2, pp. 337 – 407, 2000.
[11] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning - from Theory to Algoritms*. Cambridge University Press, 2014.
[12] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody, and S. D. Brown. An introduction to decision tree modeling. *Journal of Chemometrics*, vol. 18, n. 6, pp. 275–285, 2004.
[13] T. K. Ho. Random Decision Forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pp. 278–282 vol.1, 1995.
[14] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *CoRR*, vol. abs/1309.4168, 2013.
[15] S. Sra, S. Nowozin, and S. Wright. *Optimization for Machine Learning*. Neural information processing series. MIT Press, 2012.
[16] J. J. Levy and A. J. O'Malley. Don't dismiss logistic regression: the case for sensible extraction of interactions in the era of machine learning. *BMC Medical Research Methodology*, vol. 20, n. 1, pp. 171, 2020.
[17] T. Rymarczyk, E. Kozłowski, G. Kłosowski, and K. Niderla. Logistic regression for machine learning in process tomography. *Sensors*, vol. 19, n. 15, pp. 3400, 2019.
[18] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, vol. 20, n. 3, pp. 273–297, 1995.
[19] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
[20] R. Yacouby and D. Axman. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pp. 79–91, 2020.
[21] H. Dalianis. Evaluation metrics and evaluation. In *Clinical text mining*, pp. 45–53. Springer, 2018.
[22] X. Zhang, X. Li, Y. Feng, and Z. Liu. The use of roc and auc in the validation of objective image fusion evaluation metrics. *Signal processing*, vol. 115, pp. 38–48, 2015.
[23] S. Bates, T. Hastie, and R. Tibshirani. Cross-validation: what does it estimate and how well does it do it?, 2021.
[24] P. Refaeilzadeh, L. Tang, and H. Liu. *Cross-Validation*, pp. 1–7. Springer New York, New York, NY, 2016.
[25] R. Atenstaedt. Word cloud analysis of the BJGP. *British Journal of General Practice*, vol. 62, n. 596, pp. 148–148, 2012.
[26] L. Hickman, S. Thapa, L. Tay, M. Cao, and P. Srinivasan. Text preprocessing for text mining in organizational research: Review and recommendations. *Organizational Research Methods*, vol. 25, n. 1, pp. 114–146, 2022.
[27] M. Okkalioglu and B. D. Okkalioglu. Afe-mert: imbalanced text classification with abstract feature extraction. *Applied Intelligence*, vol. 52, pp. 1–17, 2022.
[28] N. V. Chawla and J. Sylvester. Exploiting diversity in ensembles: Improving the performance on unbalanced datasets. In M. Haindl, J. Kittler, and F. Roli, eds, *Multiple Classifier Systems*, pp. 397–406, Berlin, Heidelberg. Springer Berlin Heidelberg, 2007.
[29] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi. Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166, 2015.
[30] G. G. Sundarkumar and V. Ravi. A novel hybrid undersampling method for mining unbalanced datasets in banking and insurance. *Engineering Applications of Artificial Intelligence*, vol. 37, pp. 368–377, 2015.