Received 26 February 2024, accepted 4 May 2024, date of publication 13 May 2024, date of current version 30 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3400693

RESEARCH ARTICLE

A Flat-Hierarchical Approach Based on Machine Learning Model for e-Commerce Product Classification

HAROLD COTACALLAPA[1], NEMIAS SABOYA[1], (Member, IEEE), PAULO CANAS RODRIGUES[2], RODRIGO SALAS[3,4], (Senior Member, IEEE), AND JAVIER LINKOLK LÓPEZ-GONZALES[5], (Member, IEEE)

[1]Facultad de Ingeniería y Arquitectura, Universidad Peruana Unión, Lima 15464, Peru
[2]Department of Statistics, Federal University of Bahia, Salvador 40110-909, Brazil
[3]Escuela de Ingeniería C. Biomédica, Universidad de Valparaíso, Valparaiso 2362905, Chile
[4]Millennium Institute for Intelligent Healthcare Engineering (iHealth), Santiago 7820436, Chile
[5]Escuela de Posgrado, Universidad Peruana Unión, Lima 15464, Peru

Corresponding author: Nemias Saboya (saboya@upeu.edu.pe)

This work was supported in part by the Agencia Nacional de Investigación y Desarrollo de Chile (ANID)-Millennium Science Initiative Program under Grant ICN2021_004; in part by the ANID Fondo Nacional de Desarrollo Científico y Tecnol0́gico (FONDECYT) Research under Grant 1221938; in part by CNPq Grant "bolsa de produtividade PQ-2" under Grant 309359/2022-8; and in part by the Federal University of Bahia and CAPES-PRINT-UFBA, under the topic "Modelos Matemáticos, Estatísticos e Computacionais Aplicados ás Ciências da Natureza.

ABSTRACT Within the e-commerce sphere, optimizing the product classification process assumes pivotal importance, owing to its direct influence on operational efficiency and profitability. In this context, employing machine learning algorithms stands out as a premier solution for effectively automating this process. The design of these models commonly adopts either a flat or local (hierarchical) approach. However, each of them exhibits significant limitations. The regional approach introduces taxonomic inconsistencies in predictions, whereas the flat approach becomes inefficient when dealing with extensive datasets featuring high granularity. Therefore, our research introduces a solution for hierarchical product classification based on a Machine Learning model that integrates flat and local (hierarchical) classification approaches using a 4-level electronic product dataset obtained from a renowned e-commerce platform in Latin America. In pursuit of this goal, a comparative analysis of seven machine learning algorithms, including Multinomial Naive Bayes, Linear Support Vector Classifier, Multinomial Logistic Regression, Random Forest, XGBoost, FastText, and Voting Ensemble, was conducted. This hybrid approach model performs better than models using a single approach. It surpassed the top-performing flat approach model by 0.15% and outperformed the leading local approach (Local Classifier per Level) model by 4.88%, as measured by the weighted F1-score. Additionally, this paper contributes to the academic community by presenting a significant Spanish-language dataset comprising over one million products and discussing the preprocessing techniques tailored for the dataset. It also addresses the study's inherent limitations and potential avenues for future exploration in this field.

INDEX TERMS Machine learning, e-commerce, hierarchical product classification, local classifier per level, ensemble.

I. INTRODUCTION

The rise of e-commerce platforms in recent years, accelerated by the challenges posed by the COVID-19 pandemic, has driven the digital transformation of underdeveloped countries. This trend is particularly pronounced in Latin American countries, where post-pandemic e-commerce sales continue to outpace the global average (10.4%).[1] Brazil,

The associate editor coordinating the review of this manuscript and approving it for publication was Seifedine Kadry.

[1]Latin America Trends to Watch for 2023. Source: Insider Intelligence and eMarketer.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

Argentina, and Mexico notably lead this growth at 17.0%, 14.0%, and 13.5%, respectively.

Emphasized by Gupta et al. [1] and Das et al. [2], a crucial determinant for the success of an e-commerce platform is its product classification system. This system involves assigning a category path or taxonomy to products within a hierarchical structure organized from general to specific categories (e.g., 'Technology > Computing > Laptops and Accessories > Laptops'), as outlined by Umaashankar et al. [3]. Such an organization enables customers to swiftly and accurately retrieve products [4].

However, the challenges of achieving efficient hierarchical product classification automation lie in the vast volume of e-commerce data [5], the ambiguity in product descriptions, data imbalance [6], multilingualism [7], [8], and scalability [9]. Recent advancements in machine learning, along with the continuous efforts of numerous authors, provide tools to address these challenges. These tools span from the introduction of new datasets [3], [10] to the development of transformer-based models [8], [11] and the exploration of multimodal approaches [12], [13].

According to the reviewed literature, hierarchical classification models are typically categorized into flat, local, and global approaches, with the first two being the most prevalent [11], [14]. However, a notable drawback of the local approach is taxonomic inconsistency in predictions [15]. Conversely, the effectiveness of a flat approach diminishes notably in scenarios involving high granularity and imbalanced data. Despite these latent challenges, it is worth noting that the weaknesses of one approach are often counterbalanced by the strengths of the other [14], [16].

In this context, various studies have indeed compared and scrutinized the performance of flat and local classification approaches [15], [16], [17], [18]. However, despite sustained efforts to determine the best classification approach, conclusive findings remain elusive. To the best of our knowledge, the outcomes of combining both approaches and leveraging their strengths to enhance hierarchical classification have not yet been explored.

Motivated by this gap in the literature, this study aims to compare the performance of machine learning algorithms for hierarchical product classification using a flat, local, and hybrid approach. The hypothesis posits that the hybrid approach, integrating the strengths of both flat and local approaches, may offer a more robust solution to enhance the efficacy of hierarchical product classification within the e-commerce domain.

The study seeks to validate this hypothesis through a comprehensive evaluation of seven selected classification algorithms: Multinomial Naive Bayes (MNB), Multinomial Logistic Regression (MLR), Linear Support Vector Classifier (LSVC), XGBoost (XGB), Random Forest (RF), Hard Voting Ensemble, and FastText (FT), which are among the most commonly used traditional machine learning algorithms in product classification problems. Additionally, a detailed examination is conducted to assess the impact of preprocessing and feature engineering techniques on the performance of these models.

On the other hand, despite the longstanding nature of the issue, there is a scarcity of studies utilizing datasets in languages other than English. Liu et al. [10] attribute this scarcity to the non-availability of many datasets in the public domain. To address this limitation, our research contributes significantly by introducing a substantial Spanish dataset, featuring over one million products from Mercado Libre Peru,[2] a prominent Latin American marketplace, accessed through its public API. The dataset primarily includes product titles and taxonomies, with the hierarchical structure and category names reflecting the reality and context of Latin America. This new dataset is openly accessible on the Zenodo[3] platform.

In summary, this research introduces a novel solution for hierarchical product classification in e-commerce by integrating flat and local (hierarchical) approaches based on a Machine Learning model. The hybrid approach model demonstrated better performance compared to standalone models. Additionally, the study presents a significant Spanish dataset, serving as a benchmark for addressing multilingual challenges. This study provides insights into preprocessing techniques, discusses inherent limitations, and offers directions for future exploration in e-commerce product classification.

The paper is structured as follows: Section I introduces the problem and research objectives. Section II reviews related work. Section III discusses classification approaches and algorithms. Section IV details the research resources and methodology. Section V interprets results. Section VI discusses specific aspects of the results. Finally, Section VII presents conclusions.

II. RELATED WORK

The variety of attributes and the vast volume of data that can exist in e-commerce product datasets have motivated researchers to address this problem from various perspectives. One of the early efforts was the establishment of international standards aimed at harmonizing the hierarchical structure of different e-commerce platforms and facilitating their management [18], [19], [20].

In the realm of Machine Learning (ML) applied to product classification, studies can be categorized based on: (i) the nature of the study object: text [11], [18], [21], [22], image [23], [24], [25], or a combination of both [13], [26], [27]; (ii) the algorithmic approach employed: machine learning [20], [28], [29] or deep learning [12], [17], [18], [30]; (iii) the dataset size: small [26], [29] or large [2], [10], [16];

[2]*Mercado libre Peru* online platform.

[3]https://doi.org/10.5281/zenodo.8415496

and (iv) the classification methodology: flat, local, or big-bang [14], [15], [31].

Considering only textual product attributes, as in this research, it is common to work with, but not limited to, the product title, description, price, brand, and breadcrumbs. Nevertheless, most tend to work with the product title and/or description [4], [8], [11], [18], [22]. The following paragraphs describe several pieces of research that utilize textual product attributes as their study object.

GoldenBullet [32] was one of the first systems to apply information retrieval techniques and machine learning algorithms to address the problem of product classification. This system achieved an accuracy of 78% using the Naive Bayes algorithm with a flat approach and a dataset of 41,000 products. The authors also developed models with a local approach, but those did not outperform the flat approach [32]. Chavaltada et al. [33] conducted a performance comparison of various traditional machine learning algorithms employing a flat classification approach across three distinct datasets. The largest dataset comprised 28,355 product names. The outcomes indicated that the Naive Bayes emerged as the best-performing model. Recently, Oancea [34] conducted a performance comparison of 13 traditional classification models using 2,853 product titles. Among these models were Random Forest, XGBoost, KNN, and Artificial Neural Networks. However, Logistic Regression and Support Vector Machine algorithms outperformed the others, achieving a weighted F1-score metric of 96.3% and 95.7%, respectively.

Regarding large-scale datasets, Ha et al. [5] developed a flat categorization model called DeepCN based on multiple recurrent neural networks and using a dataset of 94 million products. Similarly, Xia et al. [35] employed attention mechanisms and convolutional neural networks (CNNs) to streamline training and eliminate the need for feature engineering in dimensionality reduction. Their study focused on the top-level categories of the Rakuten Ichiba catalog, so the resulting model adopted a flat approach. Moreover, in the SIGIR 2018 eCom Rakuten Data Challenge, the top-performing model emerged as a bidirectional ensemble of six LSTMs, utilizing a flat approach and achieving an outstanding weighted F1-score of 85.13%. The dataset consisted of over one million products [30]. Furthermore, Das et al. [2] illustrated that incorporating price and product navigational breadcrumbs can notably enhance classifier performance. The authors conducted experiments on two extensive datasets, one sourced from Rakuten and the other from Amazon. They recommended the application of gradient-boosted trees (GBTs) and CNNs for product taxonomy categorization on top-level category subtrees.

As observed, there is a tendency to use traditional ML algorithms to classify e-commerce products when dealing with a small dataset. However, Deep Learning (DL) algorithms are commonly employed when working with extensive datasets. The utilization of DL algorithms in larger datasets proves advantageous due to their capability to capture intricate patterns and relationships within the data. This capability becomes a critical factor in their preference over traditional ML algorithms in scenarios involving vast datasets [11], [36].

Focusing more on the type of approach used to build classification models, both the flat and the local approaches are the most common. Among the most recent studies that use the flat approach in their classification models, we can mention Oancea [34] and Hafez et al. [29], who employed traditional ML algorithms. Highlighting that Hafez et al. was the only study that used a Spanish dataset according to our literature review, they worked with 20,888 products organized into three levels and 159 final categories, also called leaf nodes. They explored models such as BM25, KNN, FKNN, XGBoost, and Multilayer Perceptron (MLP). Among these, FKNN demonstrated better performance, achieving an 84.59% F1-score. Similarly, Akritidis et al. [21] worked with 313,706 product titles organized into three levels and 191 leaf nodes. The authors proposed a novel SPC algorithm (Supervised Products Classifier) based on combining words, n-grams, and a token importance score. This algorithm outperformed Logistic Regression and Random Forest models in their study. Conversely, Lehmann et al. [7], Chen et al. [37], Skinner [30], and Suzuki et al. [38] utilize Deep Learning algorithms, considering only textual product attributes for their models.

Among the studies adopting the local classification approach, Ozyegen et al. [11] and Brinkmann and Bizer [18] stand out, both using pre-trained models based on a transformers architecture. The former employs a local classifier per level (LCL), while the latter utilizes a local classifier per node (LCN). Additionally, Allweyer et al. [20] demonstrated improved performance by combining the Support Vector Machine (SVM) algorithm with the TF-IDF method using the LCL approach. Conversely, the results of Vandic et al. [31] favor the combination of Naive Bayes with the Information Gain (IG) method using the LCN approach. These two papers compare the performance of various traditional ML algorithms, including KNN, SVM, Naive Bayes, Random Forest, and Single Layer Perceptron.

Some authors have also attempted to compare which classification approach is better. Krishnan and Amarthaluri [17] find that the flat approach is better than the local approach according to their experiments using CNN and LSTM. In contrast, Gao et al. [16] claim the opposite. They propose a model with an LCL (Local Classifier by Level) approach based on Neural Networks that outperforms both traditional flat classifiers (SVM, FastText, TextCNN) and hierarchical classifiers (HSVM, HiNet). Brinkmann and Bizer [18] also achieved similar results using neural networks in their architecture.

On the other hand, according to Bojanowski et al. [39], each language has its specific characteristics that need to be studied individually. However, despite this problem being old, few studies use datasets in languages other than English. Among them, Korean [5], [19], German [7], [20], Japanese [35], [40], Turkish [11], and Spanish [29] stand out. This behavior might be because few relevant datasets are publicly available [10].

In conclusion, our examination of related research provides a comprehensive overview of e-commerce product classification, including standardization efforts, machine learning approaches, and challenges related to datasets and languages. The comparison between flat and local classification approaches highlights the intricacies of hierarchical product classification. Our study proposes a hybrid approach that combines both methods' strengths and includes a substantial Spanish dataset to address multilingual challenges. This sets the stage for our subsequent sections.

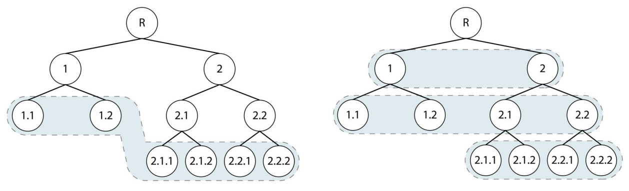
III. THEORETICAL BACKGROUND

A. HIERARCHICAL CLASSIFICATION APPROACHES

Regardless of the application domain, Silla and Freitas [14] approach hierarchical classification as a specific type of structured classification problem, where the output of the classification algorithm is a class taxonomy. Thus, based on how the model explores the hierarchical structure, they categorize classification models into flat, local, and global (big-bang) classifiers. This classification is also employed by other authors, e.g., [6], [18].

The **flat classification approach** involves the direct prediction of classes at the lowest level, known as leaf nodes, and disregards the parent-child relationship among classes. However, this approach indirectly addresses the hierarchical classification problem. This is because, assuming that the classes in the hierarchy maintain an "IS-A" relationship, assigning a lower-level class to an instance implicitly assigns all the ancestor classes to it [14]. Figure 1a illustrates this approach.

In contrast to the flat classification, the **local classification approach** considers the hierarchy by adopting a local information perspective. This approach can be divided into three distinct groups, depending on how the classes are organized in the model: Local Classifier per Parent Node (LCPN), Local Classifier per Level (LCL), and Local Classifier per Node (LCN). In LCPN, a multiclass classifier is trained for each parent node, typically using the same algorithm for each classifier. The training dataset construction often considers the "siblings" and "exclusive siblings" criteria. A notable advantage is that it uses fewer classifiers than LCN [14]. Meanwhile, the LCL approach involves training a multiclass classifier for each level of the hierarchical structure, and depending on the problem type (single-label or multi-label), it may predict one or multiple classes at the respective hierarchical level [14]. When using this approach, it is necessary to complement it with a method for dealing with taxonomy inconsistency [15]. LCN entails training a binary classifier for each node in the hierarchy, which is helpful for multi-label problems. However, as pointed out by Borges et al. [15], a notable disadvantage is the number of classifiers it can entail when working with large-scale datasets. Figure 1b illustrates the approach used in this research.



(a) Flat Classifier (b) Local Classifier Per Level

FIGURE 1. Hierarchical Classification Approaches: The dotted lines symbolize an ML model. In the flat approach, a single model classifies directly into leaf nodes. In contrast, the LCL approach employs a distinct or the same model for each hierarchy level.

The **global classification (big-bang) approach** is based on training a single classifier for all the classes within the hierarchical structure, considering the class hierarchy as a whole. This model is usually smaller than the total number of all local classifier types but lacks the modularity for training [14].

B. MACHINE LEARNING ALGORITHMS

1) MULTINOMIAL NAIVE BAYES

The Multinomial Naive Bayes (NB) classifier is a variation of the NB probabilistic algorithm designed for multinomial distributed data. Multinomial NB is used when there are multiple classes, and it aims to model the probability of an event belonging to each of these classes [41]. This model relies on Bayes' theorem and assumes conditional independence between features given the class, which simplifies probability calculations and enhances computational efficiency [33].

2) MULTINOMIAL LOGISTIC REGRESSION

The Multinomial Logistic Regression is based on the probabilistic algorithm Logistic Regression (LR) [42], which is a linear classifier aimed at establishing a relationship between features and the dependent variable (class) to predict the probability of an example belonging to a specific class. The multinomial variation [43] of this algorithm typically employs a formula that includes the softmax function or creates a set of multiple binary classifiers in a one-vs-rest (OVR) scheme [36].

3) SUPPORT VECTOR MACHINE (SVM)

SVM is a non-probabilistic supervised learning algorithm typically applied to binary classification problems [44]. However, when used for multi-class classification, it internally divides the task into multiple binary problems and solves them using multiple SVMs [45]. For linear classification, it employs support vectors from each class to construct hyperplanes between them. In cases of non-linear classification, it applies a kernel function, which maps the input data into a higher-dimensional feature space [33]. In this research,

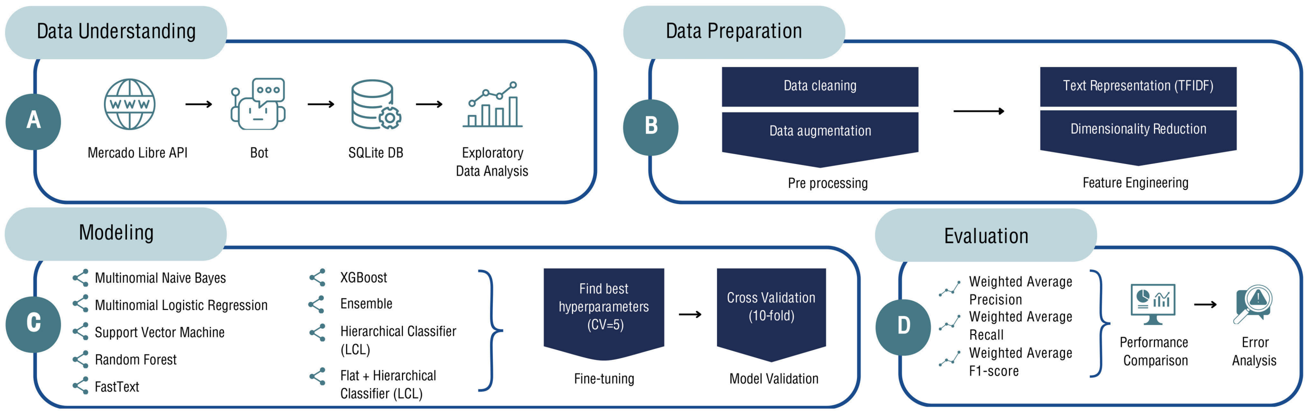


FIGURE 2. **Comprehensive guide to our research methodology based on CRISP-DM. These four primary stages of a Machine Learning project are inherently recursive, emphasizing the iterative nature of the process.**

the LinearSVC classifier [46] is used, which employs a linear kernel and is more efficient for large datasets.[4] Furthermore, it has demonstrated good performance in multi-class classification problems [11], [22], [45], [47].

4) RANDOM FOREST (RF)

Introduced by Ho [48] and further developed by Breiman [49], Random Forest (RF) is an ensemble learning algorithm that utilizes the concept of 'bagging' for sample selection. It combines multiple individual decision trees to achieve more accurate and robust predictions [36]. Each decision tree is trained on a subset of the training data, and at each tree node, a random subset of features is selected. For prediction, RF takes the majority vote from all individual trees. However, training a large number of trees can be computationally expensive, require longer training times, and consume significant memory [50].

5) EXTREME GRADIENT BOOSTING (XGBOOST)

XGBoost is also an ensemble learning algorithm that relies on the Boosting technique to train weak classifiers and combine them into a stronger model [29], [51]. It constructs sequential decision trees and adds them iteratively. Each tree adjusts its weight, giving greater importance to misclassified instances and less importance to correctly classified ones. In this way, the next classifier focuses on the "hard" instances classified by the previous model [36]. Each weak classifier learns using an objective function composed of the loss function and regularization function in each iteration [52].

6) FASTTEXT

The algorithm FastText was created by Facebook for text classification and word representation learning [39], [53]. It is based on the skip-gram architecture and, unlike previous techniques, considers the morphology of words, thereby achieving a better vector representation for out-of-vocabulary (OOV) words [36]. Facebook has provided pre-trained models for 294 languages, which were trained on Wikipedia using FastText with 300 dimensions and the skip-gram model with its default parameters [50].

[4]A Practical Guide to Support Vector Classification.

7) ENSEMBLE

Ensemble algorithms combine the predictions of various base algorithms to improve the generalization or robustness of an individual model. Two families of ensemble methods can be distinguished: Average and Boosting, with the main difference being how they construct the base algorithms. The former builds multiple independent base algorithms and then averages their predictions, while the latter constructs base algorithms sequentially, aiming to reduce the error in each iteration [54]. They also address overfitting issues and achieve good results with limited training data [55].

This article used the method of majority voting (hard voting), which involves classifying an instance according to the class that received the most votes [56]. Other studies have also considered this method in their experiments [12], [23], [27], [57], [58].

IV. MATERIALS AND METHODOLOGY

This research was conducted following the procedures outlined in Figure 2, which are rooted in the widely recognized CRISP-DM framework [59]. The CRISP-DM framework, standing for Cross-Industry Standard Process for Data Mining, provides a systematic and structured approach to guide the stages of a data mining project. In line with the CRISP-DM framework, our methodology comprises distinct phases.

In the initial stage, known as Data Understanding, data was collected from the e-commerce platform, followed by an exploratory data analysis. Subsequently, during the Data Preparation phase, accurate cleaning and transformation procedures were executed. The transformed data was then

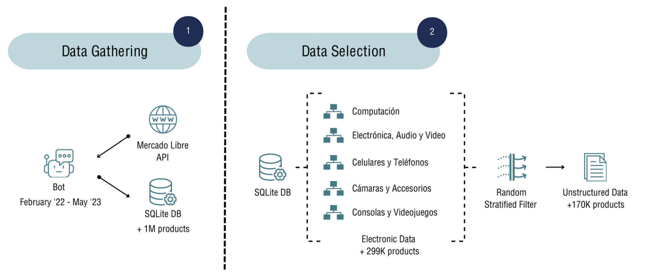


FIGURE 3. Data Retrieval and Selection Process. Step 1: Automated process for collecting and storing Data. Step 2: Selection of top-level electronics-related categories and stratified random sampling.

employed for model training and hyperparameter tuning, constituting the Modeling phase. Finally, we evaluated each algorithm's performance using metrics tailored for multiclass classification, encompassing the Evaluation phase.

This methodical and structured approach, inspired by CRISP-DM, forms a robust foundation for this study's experimental design and analysis, ensuring transparency and replicability in the research process. The subsequent sections will expound upon these steps and elucidate how the various experiments were applied.

All experiments and data analysis procedures were executed on a computing system with the following specifications: Linux Operating System version 6.2.0-31-generic x86_64, equipped with 64 CPU cores, 128 GB of RAM, and a dedicated NVIDIA GeForce RTX 3080 graphics card.

A. DATA UNDERSTANDING

Figure 3 depicts the data collection and sample selection process used for this research. This data collection occurred between February 2022 and May 2023, where a Python bot was used to access Mercado Libre's API. It collected the product code, category, title, price, currency, and product link, storing this information in a local SQLite database. In total, 1,198,398 unique products were collected and distributed across 31 categories at the top level.

Given the prevalence of technological items in the original dataset, this study is focused exclusively on categories associated with such items. These categories, such as Computing, Electronics, Audio and Video, Cell Phones and Telephones, Cameras and Accessories, and Video Game Consoles, account for 303,508 products in the dataset. Additionally, levels 5 and 6 of the hierarchical structure were discarded because only 7.85% of the products belong to a class at one or both of these levels. Finally, a stratified random sample of 170,332 instances was extracted to reduce model complexity.

Several preprocessing steps were applied to the dataset to ensure data quality and reduce noise. Firstly, all titles were converted to lowercase. Then, the following actions were employed: 1) Removal of classes with only one instance. 2) Elimination of the categories labeled as 'Others' [60].

TABLE 1. Overview of the dataset variables.

Variable	Description	Total	Example
title	Product title (categorical)	145,219	Dragon Touch Instant Print Kids Camera Instantf...b08hyn6zl6
text	Clean title (categorical)	145,219	dragon touch instant print kids camera instantf
taxonomy	Category path (categorical)	543	1039_430361_1040

TABLE 2. Data distribution for each hierarchical level.

	Nivel 1	Nivel 2	Nivel 3	Nivel 4
Number of products	145,219	145,219	137,554	71,096
Number of classes	5	57	298	299
Number of products in leaf categories	0	7,665	66,458	71,096

3) Removal of false titles. 4) Elimination of Amazon Standard Identification Number codes (e.g., 'b09mcz6ndg') and Peruvian phone numbers found at the end of some titles. 5) Deletion of duplicate titles and null cells. After this cleaning process, the dataset was left with 145,219 products, which served as the basis for all experiments in this research. This refined dataset will be referred to as D_s hereafter.

The variable descriptions for D_s can be found in Table 1. The 'title' variable represents the original product title, while the 'text' variable corresponds to the title processed using the previously mentioned methods. The 'taxonomy' variable denotes the category path to which the product belongs, with each category having a code separated by an underscore. The first code represents the most general category, while the last corresponds to the most specific category.

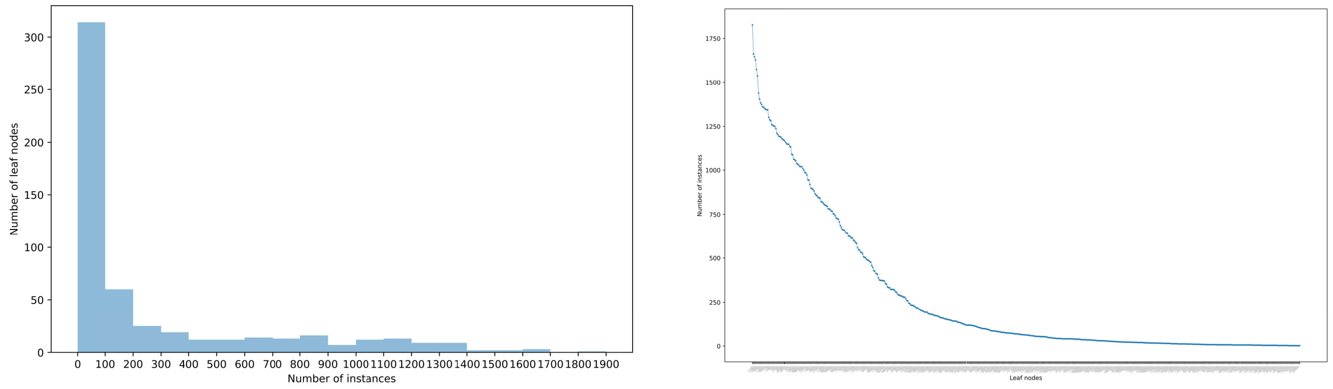
According to the exploratory data analysis, the dataset D_s exhibits a tree structure (Υ), is a Single Label Path (Ψ), and follows a Full Depth Labeling (Φ). In other words, the classes have a single root node (Technology), each node (class) has a unique parent node, and all nodes have labels. Consequently, every prediction must go down to a final node. Silla and Freitas proposed this nomenclature [14] for hierarchical classification problems.

In addition, Table 2 reveals that as the hierarchical level increases, the number of examples decreases, and the number of classes increases significantly. This implies that fewer data points are available for the model to learn to discriminate between the final classes, a phenomenon often referred to as the granularity or fine-grained challenge [10].

Other characteristics of this dataset (D_s) can also be observed in Table 3. For instance, at level 3, there are 298 classes, with a minimum of 2 products, a maximum of 6,522 products, and a median of 97.5, indicating a right-skewed distribution. Furthermore, each category contains an average of 461.59 products, with a standard deviation of

TABLE 3. Descriptive statistics of the distribution of the examples by category and by level.

	Count	Min	Q25	Median	Q75	Max	Mean	Std
Nivel 1	5	7,719	18,746	26,613	26,729	65,412	29,043.8	21,764.10
Nivel 2	57	6	430	1,245	3,212	15,895	2,547.70	3,486.57
Nivel 3	298	2	18.5	97.5	661.5	6,522	461.59	811.52
Nivel 4	299	2	15	56	237.5	1,827	237.78	369.91
Leaf categories	543	2	15.5	63	333	1,827	267.44	397.23



(a) Histogram of the number of instances.

(b) Frequency of the instances per leaf node.

FIGURE 4. Shape of the Data Distribution. (a) Histogram depicting a right-skewed distribution with a binwidth of 100 instances. (b) Frequency plot organized by the total instances per leaf node.

811.52, nearly double the average, indicating high variability in the data. This feature of imbalanced data is prevalent not only at level 3 but also throughout the entire dataset, notably affecting the distribution of leaf nodes, as illustrated in Figure 4b.

Likewise, the descriptive statistics for product titles are shown in Table 4. These statistics indicate that a title can have between 2 and 143 characters, with an average title length of 40.90. On average, each title contains eight words, with each word having an average size of 5.31 characters. In general, there is low dispersion in these indicators. However, outliers exist; for instance, some titles consist of 60 characters in a single token, resulting from the hyphenation of all the words in the title (e.g., canon-r6-mirrorless-camera-rf-24-105mm-adapter-bag-flash-tri).

The dataset D_s was split into training and test sets using an 80-20 proportion for modeling purposes. Table 5 illustrates the data distribution for each subset.

B. DATA PREPARATION

1) PRE-PROCESSING

Eight cleaning and feature engineering techniques were encoded during the data preprocessing, and multiple experiments were conducted to determine the best combination for each algorithm. The methods used were: removal of numeric characters, replacement of non-alphanumeric characters (,.%#), replacement of patterns in the text (measurements, abbreviations, codes), removal of stop words and single-character words, stemming, lemmatization, custom n-grams function, bi-grams, and tri-grams. The application of these techniques is described in the pseudocode of Algorithm 1.

Algorithm 1 Data Cleaning

```
Input: $titles$: titles to be processed
Output: $titles_p$: titles processed
 1: procedure Preprocess($titles$)
 2:     $n \leftarrow$ length($titles$)
 3:     $titles_p \leftarrow$ []
 4:     for $i \leftarrow 1$ to $n$ do
 5:         $title_p \leftarrow$ remove_numeric_token($titles[i]$)
 6:         $title_p \leftarrow$ replace_symbols($title_p$)
 7:         $title_p \leftarrow$ replace_patterns($title_p$)
 8:         $title_p \leftarrow$ remove_stopw_smallw($title_p$)
 9:         $title_p \leftarrow$ stemming($title_p$)
10:         $title_p \leftarrow$ lemmatization($title_p$)
11:         $title_p \leftarrow$ custom_ngrams($title_p$)
12:         $title_p \leftarrow$ bigrams($title_p$)
13:         $title_p \leftarrow$ trigrams($title_p$)
14:         $titles_p \leftarrow titles_p + [title_p]$
15:     end for
16:     $titles_p \leftarrow$ remove_empty_titles($titles_p$)
17:     return $titles_p$
18: end procedure
```

To address the issue of unbalanced data, illustrated in Figure 4a as the long-tailed distribution, this study employed a category-level data augmentation technique inspired by the

TABLE 4. Descriptive statistics of the product title.

	Min	Q25	Mediana	Q75	Max	Mean	Std
# chars per title	2	35	44	49	143	41.00	10.19
# tokens per title	1	6	8	10	29	8.00	2.42
Average length per token	1.5	4.5	5.13	6	60	5.32	1.35
# letters per title	1	29	37	43	137	35.56	9.74
# symbols per title	0	2	4	8	42	5.44	4.81

TABLE 5. Data distribution of training and testing datasets.

	Nivel 1	Nivel 2	Nivel 3	Nivel 4
Training data	115950	115950	109822	56754
Testing data	29269	29269	27732	14342

method used by Suzuki et al. [38]. This method involves combining all the words from a final category into an array and randomly selecting words to form a new title in that category based on the average number of words in a title. To apply this method, a threshold θ was defined, representing the minimum number of products that all final categories must have. Consequently, the number of new titles generated is the difference between the threshold θ and the number of existing titles in the final category. Three different values were considered for θ: 20, 60, and 100.

2) FEATURE ENGINEERING

- Text Representation

Some techniques provided by the scikit-learn library were used for text representation in a vector space: CountVectorizer and TfidfVectorizer. The first one calculates the term frequency (TF) in a document, and the second technique combines the first one with the Inverse Document Frequency (TF-IDF) to reduce the impact of common words [50]. Based on these two methods, three experiments were defined and used in the data preparation stage: 1) CountVectorizer, 2) TF-IDF, and 3) Sublinear TF-IDF.

- Dimensionality Reduction

The Term Frequency (TF) method and its combination with Inverse Document Frequency (IDF) are based on the corpus vocabulary, meaning the vector representation is subject to a fixed number of unique words. Consequently, when working with a large volume of data, the vocabulary becomes extensive, resulting in a high number of dimensions for the sparse matrix. This requires more processing power and longer model training times.

For this reason, feature reduction techniques are employed, with the most commonly used ones adapted for sparse matrices being: Truncated Singular Value Decomposition (TSVD), known for its speed and scalability; Latent Semantic Analysis (LSA), also based on Singular Value Decomposition (SVD); and Latent Dirichlet Allocation (LDA), a probabilistic model. This study implemented the TSVD method for feature reduction, considering 25, 100, 500, and 1000 dimensions.

C. MODELING: FLAT-HIERARCHICAL APPROACH

The modeling stage consists of two main steps, as visualized in the model selection process in Figure 5. The first step involves searching for the best hyperparameters in the vectorization and the ML algorithm. This is accomplished by using the GridSearchCV method from Scikit-learn, which employs stratified 5-fold cross-validation as the evaluation strategy. Cross-validation is a statistical technique used to assess the performance of a predictive model and reduce the risk of overfitting. All the results reported are the average performance of the n-fold used as test sets.

The second step is the validation, to ensure robust and reliable assessments, the evaluation of model performance was conducted using a 10-fold cross-validation technique and the selected metrics for model effectiveness included accuracy along with the weighted versions of F1-score, recall and precision. All the data is randomly shuffled for each fold, then a 20% stratified and random sample is extracted for the test set, with the remainder used for the training set. The average performance metrics obtained from the fold used as test sets for each model are presented in Table 7.

To build the hierarchical model that uses the local classifier per level (LCL) approach, the top three best ensemble models from the flat approach were selected and tested at each level of the hierarchical structure, as illustrated in Figure 6. For the training and validation of these algorithms at each level, 5-fold cross-validation sets were chosen from the previously constructed 10-fold set. Depending on the hierarchy level, each algorithm was trained using all products, but those that did not have a class at that level received a default label of '0'. The left-hand side of Figure 6 shows the hierarchical model construction process. The results of each model at each level are shown in Table 8.

After selecting the best model for each level, the hierarchical model (LCL) was constructed, and its performance was validated using the same 10-fold used for the flat approach models. Additionally, two experiments were conducted regarding the use of the training dataset. In the first experiment (hier-approach-1), only products belonging to a category at the studied level were considered in the training data. In the second experiment (hier-approach-2), a label equal to '0' was added for products that did not belong to a category at any level. In this way, all products had a depth level of 4. For example, if a product's taxonomy were '1500_850_26', with experiment hier-approach-2, its new taxonomy would be '1500_850_26_0'. Finally, the

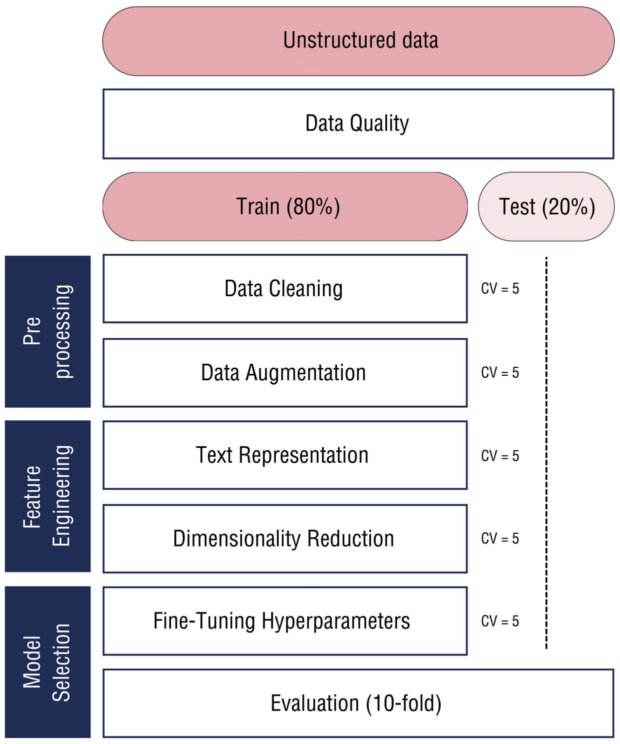


FIGURE 5. Overview of the flat model selection process.

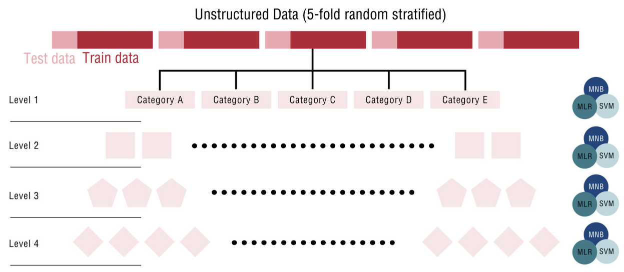


FIGURE 6. Hierarchical model selection process, based on an LCL approach.

hierarchical algorithm's (LCL) final prediction combines the individual prediction from each algorithm at every level, as illustrated in Figure 7.

To avoid predictive inconsistency in the hierarchical model, leverage its modular classification, and enhance the generalization of the flat model, a hybrid model was constructed by combining both approaches, as depicted in Figure 7. Both approaches are merged in the predictive phase.

Merging both models follows the logic of Algorithm 2, which examines each prediction from the hierarchical model, and if it doesn't exist in the list of taxonomies, it is replaced by the prediction from the flat model. This function generally avoids predictive inconsistency and retains the predictions from the existing hierarchical classifiers.

D. EVALUATION

The metrics used to assess the efficacy of the models are the weighted average (WA) versions of precision, recall,

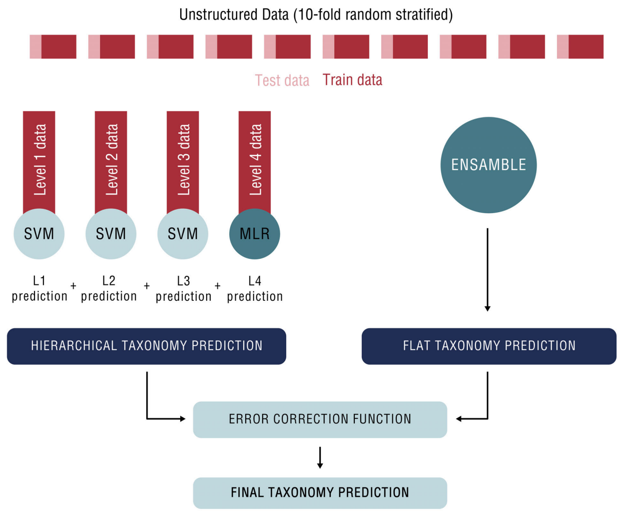


FIGURE 7. Flat-Hierarchical Approach. It utilizes an error correction function to merge predictions from the flat and local approaches.

Algorithm 2 Best Prediction Selection

```
Input: $row$: DataFrame Row
Input: $taxons$: List of all taxonomies
Output: $best\_prediction$: Best prediction for the row
1: procedure CreateBestPrediction($row$, $taxons$)
2:     if $row[taxo\_joined]$ not in $taxo\_names$ then
3:         $best\_prediction \leftarrow row[taxo\_ensemb]$
4:     else
5:         $best\_prediction \leftarrow row[taxo\_joined]$
6:     end if
7:     return $best\_prediction$
8: end procedure
```

and F1-score for the complete prediction of taxonomy, meaning that partial prediction of a product's taxonomy does not count as a correct prediction. The weighted average F1-score is also considered the decisive metric for model classification. These metrics are commonly employed in hierarchical classification problems [10], [61], [62] as they better reflect the classification quality in the presence of a highly imbalanced dataset [34]. These metrics are denoted as follows:

$$\text{WA-Precision} = \frac{1}{N}\sum_{i=1}^{K} n_i \cdot \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i} \tag{1}$$

$$\text{WA-Recall} = \frac{1}{N}\sum_{i=1}^{K} n_i \cdot \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i} \tag{2}$$

$$\text{WA-F1} = \frac{1}{N}\sum_{i=1}^{K} n_i \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \tag{3}$$

where it is assumed that there are K classes, $c_i | i = 1, 2, \ldots, K$, both in the training dataset and the testing dataset, due to the stratified partitioning. The number of true instances for

TABLE 6. Data preparation results.

Algorithm	Data Cleaning	Data Augmentation	Text Representation	Dimensionality Reduction
MNB	replace_symbols, remove_numeric_token, remove_stopw_smallw, custom_ngrams	100	CountVectorizer	NOT IMPROVED
MLR	replace_symbols, remove_stopw_smallw, stemming, custom_ngrams	100	CountVectorizer	NOT IMPROVED
LSVC	replace_symbols, remove_stopw_smallw, custom_ngrams	100	TF-IDF sublinear	NOT IMPROVED
RF	replace_symbols, replace_patterns, remove_numeric_token, remove_stopw_smallw, stemming	NOT IMPROVED	CountVectorizer	NOT IMPROVED
XGB	replace_symbols, remove_numeric_token, replace_patterns, lemmatization	NOT IMPROVED	CountVectorizer	1000
FT*	replace_symbols, remove_stopw_smallw	NOT IMPROVED	DOES NOT APPLY	DOES NOT APPLY
Ensemble (hard)*	replace_symbols, remove_stopw_smallw, custom_ngrams	100	TF-IDF sublinear	DOES NOT APPLY

* *No cross-validation applied*

each class is denoted as n_i (support), and the total number of instances is $N = \sum_{i=1}^{K} n_i$.

V. RESULTS

A. FLAT CLASSIFICATION

The flat classification approach was applied to the seven algorithms defined in this study, which were constructed following the process illustrated in Figure 5, and the results of each of the experiments conducted in the Pre-processing and Feature Engineering stage are shown in Table 6.

This table shows that the FastText and Voting Ensemble (hard) algorithms differ from the rest in the Feature Engineering stage, as both received the label 'DOES NOT APPLY.' In the case of FastText, this is because it used its own method of text representation and hyperparameter tuning. As for the Ensemble, none of the three algorithms that compose it (MNB, MLR, LSVC) improved their individual performance by reducing their dimensionality, as seen in the first rows of the same table. Therefore, that technique was not applied to this algorithm either. It is worth mentioning that in both the Preprocessing and Feature Engineering stages, the 5-fold cross-validation method was applied using only the training dataset. The test dataset was only used in the final model evaluation.

Furthermore, based on the results from Table 6, we can assert that the Data Cleaning methods that most benefit the ML models are *replace_symbols*, *remove_stopw_smallw*, and *custom_ngrams*. Additionally, the Random Forest and XGBoost algorithms perform better when adding the stemming or lemmatization technique while removing numeric tokens. The stemming technique also improves the MLR algorithm, and removing numeric tokens enhances the MNB algorithm. Similarly, the custom n-grams method outperforms all algorithms' bi-gram or tri-gram techniques.

Regarding data augmentation, Table 6 shows that the algorithms achieved their best performance when θ was set to 100, except for XGB, RF, and FT, where no value of θ improved their performance. On the other hand, the text representation method that demonstrated better results in most of the algorithms was CountVectorizer, except for the LSVC and Voting Ensemble (hard) algorithms, in which the sublinear TF-IDF method outperforms the other vectorization techniques. Finally, the Truncated-SVD method, used for dimensionality reduction, only improved the performance of XGB when reduced to 1000 dimensions.

Following the process described in Figure 5, the last stage corresponds to model selection, which involves fine-tuning hyperparameters and evaluating the algorithm's performance. We have applied a 5-fold cross-validation using grid search to select each model's best set of hyperparameters, thus avoiding overfitting. The range of values used for each hyperparameter is detailed in Appendix , including the final values and those used in the final model for each algorithm. It is important to note that the hyperparameter selection differs for the FT and Ensemble (hard) algorithms. The FT algorithm was performed using its self-training method for three hours. For the Ensemble (hard), the same hyperparameters used in the independent experiments of each algorithm were adopted.

In summary, Table 7 displays the results of the evaluation metrics for the seven algorithms with a flat classification approach. Based on these results, we can observe that the algorithm with the best performance is the Ensemble (hard), followed by LSVC and MLR. These are the only algorithms that exceed 80% in the weighted F1-score metric.

B. HIERARCHICAL CLASSIFICATION

As evidenced in Table 8, the LSVC model exhibits the best performance for levels 1, 2, and 3, while at level 4, the MLR model outperforms the LSVC model by 0.62%. Based on these results, the hierarchical (LCL) architecture comprised three LSVC models and one MLR model, as illustrated in Figure 7.

As observed in the same table, the efficacy of the models decreases as the hierarchy level increases. However, there is a sharp decline at level 4, primarily because the classes are more specific at this level. Consequently, it becomes more challenging to identify patterns that differentiate between them. This effect is amplified by the increase in the number

TABLE 7. Model performance using the flat approach (10-folds).

Algorithm	accuracy	waF1	waPrecision	waRecall
MNB	0.8097 ± 0.002	0.8045 ± 0.002	0.8072 ± 0.002	0.8097 ± 0.002
MLR	0.8151 ± 0.002	0.8132 ± 0.002	0.8154 ± 0.002	0.8151 ± 0.002
LSVC	0.8238 ± 0.002	0.8183 ± 0.002	0.8194 ± 0.002	0.8238 ± 0.002
RF	0.7895 ± 0.002	0.7818 ± 0.002	0.7846 ± 0.002	0.7895 ± 0.002
XGB	0.7717 ± 0.003	0.7643 ± 0.003	0.7625 ± 0.003	0.7717 ± 0.003
FT	0.7854 ± 0.003	0.7811 ± 0.003	0.7839 ± 0.002	0.7854 ± 0.003
Ensemble (hard)	**0.8254 ± 0.002**	**0.8197 ± 0.002**	**0.8210 ± 0.002**	**0.8254 ± 0.002**

TABLE 8. Model performance per hierarchy level (5-folds).

Level	Algorithm	accuracy	waF1	waPrecision	waRecall
	MNB	0,9282 ± 0.001	0,9284 ± 0.001	0,9288 ± 0.001	0,9282 ± 0.001
L1	MLR	0,9344 ± 0.000	0,9345 ± 0.000	0,9347 ± 0.000	0,9344 ± 0.000
	LSVC	0,9412 ± 0.001	**0,9411 ± 0.001**	0,9411 ± 0.001	0,9412 ± 0.001
	MNB	0,8877 ± 0.001	0,8877 ± 0.001	0,8900 ± 0.001	0,8877 ± 0.001
L2	MLR	0,8910 ± 0.002	0,8912 ± 0.002	0,8924 ± 0.002	0,8910 ± 0.002
	LSVC	0,9009 ± 0.002	**0,8999 ± 0.002**	0,8999 ± 0.002	0,9009 ± 0.002
	MNB	0,7947 ± 0.002	0,7754 ± 0.002	0,7663 ± 0.002	0,7947 ± 0.002
L3	MLR	0,8005 ± 0.001	0,7822 ± 0.001	0,7711 ± 0.001	0,8005 ± 0.001
	LSVC	0,8103 ± 0.001	**0,7883 ± 0.001**	0,7745 ± 0.001	0,8103 ± 0.001
	MNB	0,4093 ± 0.001	0,2978 ± 0.001	0,2446 ± 0.001	0,4093 ± 0.001
L4	**MLR**	0,4157 ± 0.001	**0,3102 ± 0.001**	0,2627 ± 0.001	0,4157 ± 0.001
	LSVC	0,4190 ± 0.001	0,3040 ± 0.001	0,2500 ± 0.001	0,4190 ± 0.001

of classes (2) and the limited representation per class. This behavior is typical in hierarchical models [11], [20].

The specific results of the hierarchical model (LCL) are presented in Tables 9 and 10. The difference between these two tables is their use of the training data. Table 9 uses the hier-approach-1, yielding a weighted F1-score of 39.26%, while Table 10 uses the hier-approach-2, yielding a weighted F1-score that reaches 77.23%. This comparison shows that the hier-approach-2, which uses all products from the dataset in each model and assigns the label '0' to products that do not belong to a category at the predicted hierarchy level, is the more effective approach.

C. FLAT-HIERARCHICAL CLASSIFICATION

Tables 9 and 10 also show that when using Algorithm 2 to combine the predictions of the flat model and the hierarchical model (LCL), the results diverge depending on the use of training data. Specifically, when the hier-approach-1 is used, the *flat-hierarchical* model achieves a weighted F1-score of 81.90%, which is slightly lower than the flat model's score of 81.96%. Conversely, when hier-approach-2 is employed, the proposed *flat-hierarchical* model attains a weighted F1-score of 82.11%, surpassing the *flat* model by 0.15% and the *hierarchical* model by 4.88%.

D. ERROR ANALYSIS

This section conducts a detailed analysis of the errors stemming from our hierarchical classification model. The aim is to comprehend the areas where the model may encounter failures and provide valuable insights for future enhancements.

Figure 8 illustrates the confusion matrix of the proposed model in this research, considering the 543 final classes or leaf nodes of the dataset D_s. From this figure, we observe that in most classes where our model is ineffective, it is due to a close similarity with another class. For example, only 4% of instances from the class taxonomy 1648_430687_445198_445199, referring to *Computación* > *Laptops y Accesorios* > *Repuestos para Laptops* > *Memorias RAM para Laptops*, were predicted correctly. Instead, 93% of them were assigned to the leaf node 1648_444889_1694, corresponding to *Computación* > *Componentes de PC* > *Memorias RAM*. Therefore, this close similarity between both leaf nodes is one of the reasons why the model fails to be more effective.

On the other hand, based on Figure 9, there is clear evidence that when a class has more instances, the model can better predict the taxonomy of new products. Although some classes with limited representation achieved acceptable performance, 14% of the leaf nodes (78) attained an F1-score of 0, with the number of instances in the test dataset ranging from 1 to 11. This reinforces the importance of the quantity of representation per class.

VI. DISCUSSION

According to the literature review, the results of this research align with previous studies in this area. Firstly, it was demonstrated that a flat model outperforms a hierarchical model (LCL), which is consistent with findings by Ding et al. [32], who used traditional ML algorithms and an LCL-type hierarchical classifier, and Krishnan and Amarthaluri [17], who used Deep Learning algorithms and an LCN-type hierarchical classifier. However, other authors have shown

TABLE 9. **Flat-hierarchical model performance using hier-approach-1 (10-folds).**

Model	accuracy	waF1	waPrecision	waRecall
lsvc-level1	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001
lsvc-level2	0,9012 $\pm$ 0.002	0,9002 $\pm$ 0.002	0,9002 $\pm$ 0.002	0,9012 $\pm$ 0.002
lsvc-level3	0,8112 $\pm$ 0.001	0,7889 $\pm$ 0.002	0,7751 $\pm$ 0.001	0,8112 $\pm$ 0.001
mlr-level4	0,4118 $\pm$ 0.001	0,3090 $\pm$ 0.001	0,2623 $\pm$ 0.001	0,4118 $\pm$ 0.001
hierarchical	0,3810 $\pm$ 0.001	0,3926 $\pm$ 0.001	0,4100 $\pm$ 0.001	0,3810 $\pm$ 0.001
flat	**0,8252** $\pm$ **0.002**	**0,8196** $\pm$ **0.002**	0,8209 $\pm$ 0.002	**0,8252** $\pm$ **0.002**
flat-hierarchical	0,8239 $\pm$ 0.002	0,8190 $\pm$ 0.002	**0,8210** $\pm$ **0.002**	0,8239 $\pm$ 0.002

TABLE 10. **Flat-hierarchical model performance using hier-approach-2 (10-folds).**

Model	accuracy	waF1	waPrecision	waRecall
lsvc-level1	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001	0,9418 $\pm$ 0.001
lsvc-level2	0,9012 $\pm$ 0.002	0,9002 $\pm$ 0.002	0,9002 $\pm$ 0.002	0,9012 $\pm$ 0.002
lsvc-level3	0,8521 $\pm$ 0.001	0,8482 $\pm$ 0.002	0,8486 $\pm$ 0.001	0,8521 $\pm$ 0.001
mlr-level4	0,7951 $\pm$ 0.002	0,8080 $\pm$ 0.002	0,8452 $\pm$ 0.002	0,7951 $\pm$ 0.002
hierarchical	0,7283 $\pm$ 0.002	0,7723 $\pm$ 0.002	**0,8606** $\pm$ **0.001**	0,7283 $\pm$ 0.002
flat	0,8252 $\pm$ 0.002	0,8196 $\pm$ 0.002	0,8209 $\pm$ 0.002	0,8252 $\pm$ 0.002
flat-hierarchical	**0,8263** $\pm$ **0.002**	**0,8211** $\pm$ **0.002**	0,8225 $\pm$ 0.002	**0,8263** $\pm$ **0.002**

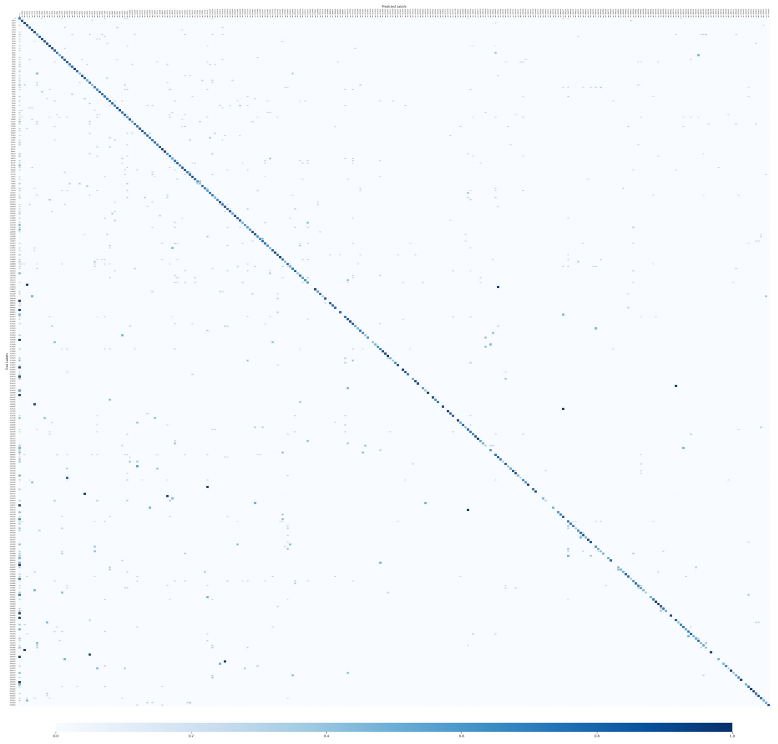


FIGURE 8. **Confusion matrix for the flat-hierarchical ML model. The results belong to one of the ten folds used for the final validation model.**

that a hierarchical approach based on a neural network architecture can yield better performance [16], [18], a factor not considered in this study. Additionally, Ozyegen et al. [11] suggest using pre-trained models and transformers to enhance the effectiveness of the hierarchical model (LCL).

Secondly, when comparing the performance of traditional ML algorithms, LinearSVC stands out among all of them. These results are consistent with those obtained by Allweyer et al. [20] and Goumy and Mejri [63]. The first study worked with a dataset similar in size and depth to the hierarchical structure and developed a hierarchical model (LCL). On the other hand, the second study worked with a much larger dataset and developed both a flat and a hierarchical model that combined LCL and LCPN approaches. Both studies concluded that the LinearSVC algorithm demonstrated the best performance.

Thirdly, to the best of our knowledge, no research papers have investigated the combination of flat and hierarchical approaches in a machine learning model, which adds greater significance to this research. However, this study has some

TABLE 11. Fine-tuning hyperparameters for Machine Learning models.

Algorithm	Search space (cv=5)	Best hyperparameters	Model

MNB	min_df: [1, 2, 3, 4, 6, 10],
	alpha: [0.001, 0.01, 0.05, 0.1, 1.0, 5.0, 10.0]
		min_df'=1
		alpha=0.01
			CountVectorizer()
			MultinomialNB(alpha=0.01)

MLR	min_df: [1,2,3,4,8],
	max_features: [1000, 10000, 50000, 100000, 150000, None]
	C: [0.5, 1.0, 2.0, 3.0, 5.0, 10.0],
	solver: ['saga','sag'],
	class_weight: ['balanced',None]
		min_df': 1
		max_features: None
		C: 5.0,
		solver: 'sag'
		class_weight': 'balanced'
			CountVectorizer()
			LogisticRegression(
			C=5.0, solver='sag'
			class_weight='balanced',
			max_iter=1000,
			multi_class='multinomial',
			n_jobs=-1, random_state=42
			)

LSVC	min_df: [1,2,3,4,6,10],
	max_features: [1000, 10000, 50000, 100000, None]
	C: [1.0, 2.0, 3.0, 5.0, 10.0, 15.0],
	loss: ['hinge', 'squared_hinge'],
	class_weight: ['balanced', None],
	dual: [True, False],
	max_iter: [100,1000,10000,100000]
		max_features: None,
		min_df: 1,
		C: 2.0, loss:'hinge',
		max_iter:100,
		class_weight: None,
		dual: True
			CountVectorizer()
			TfidfTransformer(sublinear_tf=True)
			LinearSVC(
			C=2.0,
			loss='hinge', max_iter=100,
			random_state=42, verbose=3
			)

RF	min_df: [1,3,4,8,10],
	max_features: [1000, 10000, 100000, None]
	criterion: ['gini','entropy','log_loss'],
	max_depth: [50,100,150,300],
	n_estimators: [70,100,200]
		max_features: None,
		min_df: 8,
		criterion: 'gini',
		max_depth: 300,
		n_estimators: 70
			CountVectorizer(min_df=8)
			RandomForestClassifier(
			max_depth=300,
			n_estimators=70, n_jobs=-1,
			random_state=42
			)

XGB	min_df: [1,3,4,8,10],
	max_features: [2000, 5000, 10000, 100000, None]
	max_depth: [1,2,3,4,5,7,10,12]
	objective: ['multi:softprob','multi:softmax'],
	min_child_weight: [1,2,3,5,7,10],
	learning_rate:[0.05,0.07,0.1,0.2,0.3,0.4,0.5]
	n_estimators: [70,100,200]
		max_features: 10000,
		min_df: 3,
		max_depth: 7,
		objective: 'multi:softmax',
		min_child_weight: 10
		learning_rate:0.2
		n_estimators:500
			CountVectorizer(max_features=10000,
			min_df=3)
			TruncatedSVD(n_components=1000)
			MinMaxScaler(),
			XGBClassifier(learning_rate=0.2,
			max_depth=7, min_child_weight=10,
			n_estimators=500, num_class=475,
			objective='multi:softmax', n_jobs=-1)
			)

FT	NOT APPLY	dim=200, epochs=100,
		lr=0.05
			dim=200, epochs=100, lr=0.05

Ensemble (hard)	NOT APPLY	NOT APPLY	VotingClassifier(estimators=[('svm', svm_model), ('nb', nb_model), ('lr', lr_model)], voting='hard')

limitations, such as not experimenting with pre-trained models for text representation (e.g., BERT, RoBERTa, ELMO) [50], classification models based on deep learning, or large language models (LLM). Additionally, this study did not use the original dataset, which consists of more than one million products and could be helpful for more complex models.

Finally, the dataset provided by this study is the first in the scientific community with its characteristics: Spanish-language, publicly accessible, large-scale, and with a 6-level hierarchical structure. Among other publicly accessible datasets used for hierarchical product classification, we can mention the large-scale dataset provided by the SIGIR 2018 eCom Rakuten Data Challenge, which consists of one million products in English [61]. Similarly, the MWPD Challenge provided a hierarchical dataset of 13K products [62], and Brinkmann and Bizer [18] used the ICECAT/WDC-222 dataset, which contains over 750K products. However, it is worth noting that the datasets mentioned, including those presented in this research, are single-modal, which is unsuitable for multimodal models (text and image).

VII. CONCLUSION

After thoroughly analyzing the data and evidence, several significant conclusions can be drawn. Firstly, it is confirmed that the best non-ensemble ML model is LinearSVC combined with the sublinear TF-IDF method, and the most effective data cleaning methods for this dataset are: *replace_symbols*, *remove_stopw_smallw* y *custom_ngrams*, with *custom_ngrams* notably providing better features than the common uni-gram and bi-gram.

Secondly, it is established that the flat-focused model, Voting Ensemble (hard), outperforms the hierarchical-focused model (LCL) by 5.39% according to the weighted F1-score metric. It was also concluded that the best strategy for using training data in the hierarchical model (LCL) is the hier-approach-2, which outperforms hier-approach-1 by 33.79%.

Finally, the ML model suggested in this study, which combines flat and hierarchical approaches for product taxonomy

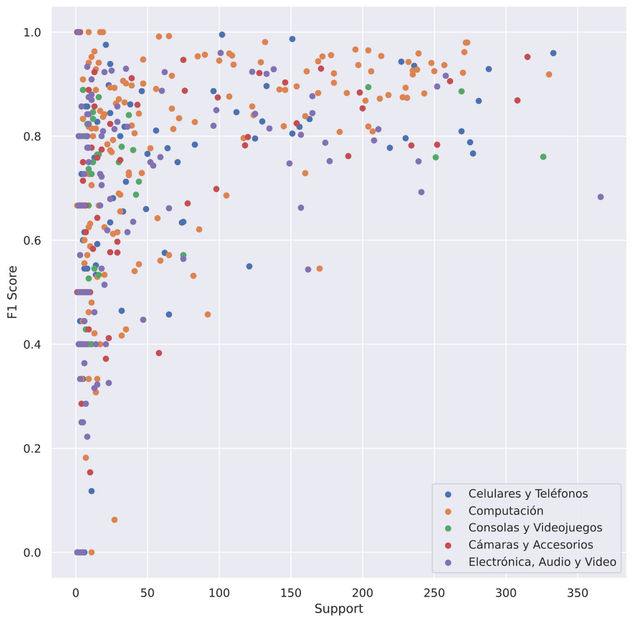


FIGURE 9. Correlation between F1-score and Support values for each leaf node, grouped by the top-level category.

prediction, outperforms the flat_ensemble model by 0.15% and the hierarchical model by 4.88%. This demonstrates its potential as a solution for improving a hierarchical classification system.

While this study has shed light on certain aspects, unexplored areas still deserve more detailed investigation. Future research could use transformers and LLM (Large Language Models) in classification models to determine if combining both approaches can improve individual performance. Furthermore, it is also necessary to investigate the performance of other variations of local classifiers, such as LCPN and LCPN. The proposal can be further improved by characterizing the product from e-commerce platforms using network analysis, similar to the work of [64]. Another potential avenue for research could be multilingual models or those using Transfer Learning methods because collecting data in Portuguese from the same e-commerce platform is possible. Ultimately, considering the various paths for future research, the proposed dataset is believed to motivate and facilitate multilingual research and research into more complex models requiring large-scale datasets.

APPENDIX. FINE-TUNING HYPERPARAMETERS

See Table 11.

REFERENCES

[1] V. Gupta, H. Karnick, A. Bansal, and P. Jhala, "Product classification in e-commerce using distributional semantics," in *Proc. 26th Int. Conf. Comput. Linguistics, Tech. Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, 2016, pp. 536–546.

[2] P. Das, Y. Xia, A. Levine, G. Di Fabbrizio, and A. Datta, "Large-scale taxonomy categorization for noisy product listings," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 3885–3894.

[3] V. Umaashankar, S. G. Shanmugam, and A. Prakash, "Atlas: A dataset and benchmark for e-commerce clothing product categorization," 2019, *arXiv:1908.08984*.

[4] Z. Kozareva, "Everyone likes shopping! Multi-class product categorization for e-commerce," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Human Lang. Technol.* Denver, CO, USA: Association for Computational Linguistics, May 2015, pp. 1329–1333.

[5] J.-W. Ha, H. Pyo, and J. Kim, "Large-scale item categorization in e-commerce using multiple recurrent neural networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 107–115.

[6] R. M. Pereira, Y. M. G. Costa, and C. N. Silla, "Handling imbalance in hierarchical classification problems using local classifiers approaches," *Data Mining Knowl. Discovery*, vol. 35, no. 4, pp. 1564–1621, Jul. 2021.

[7] E. Lehmann, A. Simonyi, L. Henkel, and J. Franke, "Bilingual transfer learning for online product classification," in *Proc. Workshop Natural Language Process. E-Commerce*. Barcelona, Spain: Association for Computational Linguistics, Dec. 2020, pp. 21–31.

[8] W. Zhang, Y. Lu, B. Dubrov, Z. Xu, S. Shang, and E. Maldonado, "Deep hierarchical product classification based on pre-trained multilingual knowledge," *IEEE Bull. Tech. Committee Data Eng.*, vol. 44, no. 2, pp. 26–37, Jun. 2021.

[9] I. Hasson, S. Novgorodov, G. Fuchs, and Y. Acriche, "Category recognition in e-commerce using sequence-to-sequence hierarchical classification," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 902–905.

[10] F. Liu, D. Chen, X. Du, R. Gao, and F. Xu, "MEP-3M: A large-scale multi-modal e-commerce product dataset," *Pattern Recognit.*, vol. 140, Aug. 2023, Art. no. 109519.

[11] O. Ozyegen, H. Jahanshahi, M. Cevik, B. Bulut, D. Yigit, F. F. Gonen, and A. Başar, "Classifying multi-level product categories using dynamic masking and transformer models," *J. Data, Inf. Manage.*, vol. 4, no. 1, pp. 71–85, Mar. 2022.

[12] T. M. Tashu, S. Fattouh, P. Kiss, and T. Horváth, "Multimodal e-commerce product classification using hierarchical fusion," in *Proc. IEEE 2nd Conf. Inf. Technol. Data Sci. (CITDS)*, May 2022, pp. 279–284.

[13] Q. Chen, Z. Shi, Z. Zuo, J. Fu, and Y. Sun, "Two-stream hybrid attention network for multimodal classification," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2021, pp. 359–363.

[14] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining Knowl. Discovery*, vol. 22, nos. 1–2, pp. 31–72, Jan. 2011.

[15] H. B. Borges, C. N. Silla, and J. C. Nievola, "An evaluation of global-model hierarchical classification algorithms for hierarchical classification problems with single path of labels," *Comput. Math. Appl.*, vol. 66, no. 10, pp. 1991–2002, Dec. 2013.

[16] D. Gao, W. Yang, H. Zhou, Y. Wei, Y. Hu, and H. Wang, "Deep hierarchical classification for category prediction in e-commerce system," 2020, *arXiv:2005.06692*.

[17] A. Krishnan and A. Amarthaluri, "Large scale product categorization using structured and unstructured attributes," 2019, *arXiv:1903.04254*.

[18] A. Brinkmann and C. Bizer, "Improving hierarchical product classification using domain-specific language modelling," *IEEE Data Eng. Bull.*, vol. 44, no. 2, pp. 14–25, Jun. 2021.

[19] M. Harth, C. Schorr, and R. Krieger, "A hierarchical multi-level product classification workbench for retail," in *Proc. LWDA*, vol. 2738, 2020, pp. 59–69.

[20] O. Allweyer, C. Schorr, R. Krieger, and A. Mohr, "Classification of products in retail using partially abbreviated product names only," in *Proc. 9th Int. Conf. Data Sci., Technol. Appl.* SciTePress, Jul. 2020, pp. 67–77.

[21] L. Akritidis, A. Fevgas, and P. Bozanis, "Effective products categorization with importance scores and morphological analysis of the titles," in *Proc. IEEE 30th Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2018, pp. 213–220.

[22] J. Dai, T. Wang, and S. Wang, "A deep forest method for classifying e-commerce products by using title information," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2020, pp. 1–5.

[23] S. A. Oyewole and O. O. Olugbara, "Product image classification using eigen colour feature with ensemble machine learning," *Egyptian Informat. J.*, vol. 19, no. 2, pp. 83–100, Jul. 2018.

[24] Y. Tang, F. Borisyuk, S. Malreddy, Y. Li, Y. Liu, and S. Kirshner, "MSURU: Large scale e-commerce image classification with weakly supervised search data," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Jul. 2019, p. 2518.

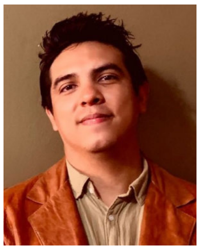
[25] Y. Seo and K.-S. Shin, "Hierarchical convolutional neural networks for fashion image classification," *Expert Syst. Appl.*, vol. 116, pp. 328–339, Feb. 2019.
[26] E. Verma, S. Chakraborty, and V. Motupalli, "Deep multi-level boosted fusion learning framework for multi-modal product classification," in *Proc. SIGIR eCom*, 2018, pp. 1–5.
[27] Y. Bi, S. Wang, and Z. Fan, "A multimodal late fusion model for e-commerce product classification," 2020, *arXiv:2008.06179*.
[28] C. Chavaltada, K. Pasupa, and D. R. Hardoon, "Combining multiple features for product categorisation by multiple kernel learning," in *Recent Advances in Information and Communication Technology*. Chiang Mai, Thailand: Springer, 2019, pp. 3–12.
[29] M. M. Hafez, A. Fernández Vilas, R. P. D. Redondo, and H. O. Pazó, "Classification of retail products: From probabilistic ranking to neural networks," *Appl. Sci.*, vol. 11, no. 9, p. 4117, Apr. 2021.
[30] M. Skinner, "Product categorization with LSTMs and balanced pooling views," in *Proc. SIGIR eCom*, 2018, p. 15.
[31] D. Vandic, F. Frasincar, and U. Kaymak, "A framework for product description classification in e-commerce," *J. Web Eng.*, vol. 17, pp. 1–27, Mar. 2018.
[32] Y. Ding, M. Korotkiy, B. Omelayenko, V. Kartseva, V. Zykov, M. Klein, E. Schulten, and D. Fensel, "GoldenBullet: Automated classification of product data in e-commerce," in *Proc. 5th Int. Conf. Bus. Inf. Syst.*, vol. 5, 2002, p. 9.
[33] C. Chavaltada, K. Pasupa, and D. R. Hardoon, "A comparative study of machine learning techniques for automatic product categorisation," in *Advances in Neural Networks—ISNN 2017* (Lecture Notes in Computer Science, Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Cham, Switzerland: Springer, 2017, pp. 10–17.
[34] B. Oancea, "Automatic product classification using supervised machine learning algorithms in price statistics," *Mathematics*, vol. 11, no. 7, p. 1588, Mar. 2023.
[35] Y. Xia, A. Levine, P. Das, G. Di Fabbrizio, K. Shinzato, and A. Datta, "Large-scale categorization of Japanese product titles using neural attention models," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, vol. 2. Valencia, Spain: Association for Computational Linguistics, 2017, pp. 663–668.
[36] A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli, "A survey on text classification algorithms: From text to predictions," *Information*, vol. 13, no. 2, p. 83, Feb. 2022.
[37] H. Chen, J. Zhao, and D. Yin, "Fine-grained product categorization in e-commerce," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 2349–2352.
[38] S. Suzuki, Y. Iseki, H. Shiino, H. Zhang, A. Iwamoto, and F. Takahashi, "Convolutional neural network and bidirectional LSTM based taxonomy classification using external dataset at SIGIR eCom data challenge," in *Proc. SIGIR eCom*, vol. 2319, 2018, p. 5.
[39] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.
[40] L. Tan, M. Y. Li, and S. Kok, "E-commerce product categorization via machine translation," *ACM Trans. Manage. Inf. Syst.*, vol. 11, no. 3, pp. 1–14, Sep. 2020.
[41] S. Xu, Y. Li, and Z. Wang, "Bayesian multinomial Naïve Bayes classifier to text classification," in *Advanced Multimedia and Ubiquitous Engineering* (Lecture Notes in Electrical Engineering), vol. 448. Singapore: Springer, 2017, pp. 347–352.
[42] A. Genkin, D. D. Lewis, and D. Madigan, "Large-scale Bayesian logistic regression for text categorization," *Technometrics*, vol. 49, no. 3, pp. 291–304, Aug. 2007.
[43] B. Krishnapuram, L. Carin, M. A. T. Figueiredo, and A. J. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 957–968, Jun. 2005.
[44] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999.
[45] C. N. Kamath, S. S. Bukhari, and A. Dengel, "Comparative study between traditional machine learning and deep learning approaches for text classification," in *Proc. ACM Symp. Document Eng.*, Aug. 2018, pp. 1–11.
[46] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.
[47] C. Heistracher, F. Mignet, and S. Schlarb, "Machine learning techniques for the classification of product descriptions from darknet marketplaces," in *Proc. ICAI*, 2020, pp. 128–137.
[48] T. K. Ho, "Random decision forests," in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, vol. 1, Aug. 1995, pp. 278–282.
[49] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 5–32, Oct. 2001.
[50] U. Naseem, I. Razzak, S. K. Khan, and M. Prasad, "A comprehensive survey on word representation models: From classical to state-of-the-art word representation language models," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 20, no. 5, pp. 1–35, Sep. 2021.
[51] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2016, pp. 785–794.
[52] Abdullah-All-Tanvir, I. Ali Khandokar, A. K. M. M. Islam, S. Islam, and S. Shatabda, "A gradient boosting classifier for purchase intention prediction of online shoppers," *Heliyon*, vol. 9, no. 4, Apr. 2023, Art. no. e15163.
[53] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, vol. 2, Jul. 2017, pp. 427–431.
[54] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Dec. 2011.
[55] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining Knowl. Discovery*, vol. 8, no. 4, p. 18, Jul. 2018.
[56] C. A. Gonçalves, A. S. Vieira, C. T. Gonçalves, R. Camacho, E. L. Iglesias, and L. B. Diz, "A novel multi-view ensemble learning architecture to improve the structured text classification," *Information*, vol. 13, no. 6, p. 283, Jun. 2022.
[57] L. Yang, E. Shijia, S. Xu, and Y. Xiang, "Bert with dynamic masked softmax and pseudo labeling for hierarchical product classification," in *Proc. MWPD@ISWC*, 2020, p. 8.
[58] G. P. D. Oliveira, A. Fonseca, and P. C. Rodrigues, "Diabetes diagnosis based on hard and soft voting classifiers combining statistical learning models," *Brazilian J. Biometrics*, vol. 40, no. 4, pp. 415–427, Dec. 2022.
[59] R. Wirth and J. Hipp, "CRISP-DM: Towards a standard process model for data mining," in *Proc. 4th Int. Conf. Practical Appl. Knowl. Discovery Data Mining*, vol. 1, Manchester, U.K., 2000, pp. 29–39.
[60] A. Cevahir and K. Murakami, "Large-scale multi-class and hierarchical product categorization for an e-commerce giant," in *Proc. 26th Int. Conf. Comput. Linguistics*. Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 525–535.
[61] Y.-C. Lin, P. Das, and A. Datta, "Overview of the SIGIR 2018 eCom Rakuten data challenge," in *Proc. SIGIR eCom*, vol. 2319, 2018, p. 8.
[62] Z. Zhang, C. Bizer, R. Peeters, and A. Primpeli, "MWPD2020: Semantic web challenge on mining the web of HTML-embedded product data," in *Proc. MWPD@ISWC*, vol. 2720, 2020, p. 17.
[63] S. Goumy and M.-A. Mejri, "Ecommerce product title classification," in *Proc. SIGIR eCom*, vol. 2319, 2018, p. 4.
[64] F. H. Leiva, R. Torres, O. Nicolis, and R. Salas F., "Characterization of the Chilean public procurement ecosystem using social network analysis," *IEEE Access*, vol. 8, pp. 138846–138858, 2020.

HAROLD COTACALLAPA received the B.S. degree in systems engineering from Universidad Peruana Unión, Peru. He has experience working as a Software Developer and a Data Analyst. His main research interests include machine learning applications in neuroscience, climate change, and finances.

NEMIAS SABOYA (Member, IEEE) received the B.S. degree in systems engineering from Universidad Peruana Unión (UPeU), Peru, and the M.S. degree in computer and systems engineering, with a specialization in information technology management, USMP, Peru. He is currently pursuing the Ph.D. degree in systems engineering with UPeU. His main research interests include data governance, statistical machine learning, IT, and data science.

PAULO CANAS RODRIGUES received the Ph.D. degree in statistics from the Nova University of Lisbon, Portugal, in 2012, and the Aggregation (Habilitation) degree in mathematics, with a specialization in statistics and stochastic processes, Lisbon University, Portugal, in 2019. He is currently a Professor of statistics with the Federal University of Bahia and the Head and Principal Investigator of the Statistical Learning Laboratory (SaLLy). His research interests include statistical learning, time series forecasting, and data science in general.

RODRIGO SALAS (Senior Member, IEEE) received the B.S. and M.Sc. degrees in informatics engineering and the Dr. Eng. degree in informatics from Federico Santa María Technical University (UTFSM), Chile, in 2001, 2002 and 2010, respectively. From 2002 to 2004, he was a Research Assistant with the Department of Informatics, UTFSM. Since 2004, he has been with Universidad de Valparaíso, where he is currently a Full Professor with the Biomedical Engineering School and teaches in data mining, probability and statistics, and machine learning. He is also a Main Researcher with the Millennium Institute for Intelligent Healthcare Engineering (i-HEALTH) and he is the Director of the Center of Interdisciplinary Biomedical and Engineering Research for Health (MEDING). His research interests include artificial intelligence, data science, computational statistics, decision support systems, intelligent systems and their applications to finance, air pollution, and healthcare and medicine.



JAVIER LINKOLK LÓPEZ-GONZALES (Member, IEEE) received the B.S. degree in statistical and informatics engineering from Universidad Peruana Unión (UPeU), Peru, the M.Sc. degree in metrology from Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil, and the Ph.D. degree in statistics from Universidad de Valparaíso (UV), Chile. He is currently a Senior Professor with the Universidad Peruana Unión. In addition, he is qualified as RENACYT Researcher. His main research interests include pattern recognition in machine learning, air pollution with deep learning techniques, and time series with singular spectrum analysis.