applied sciences

MDPI

Article

Classification of Retail Products: From Probabilistic Ranking to Neural Networks

Manar Mohamed Hafez [1,*], Ana Fernández Vilas [2], Rebeca P. Díaz Redondo [2] and Héctor Olivera Pazó [2]

[1] College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport (AASTMT)-Smart Village, Giza P.O. Box 12676, Egypt
[2] AtlanTTic, Universiade de Vigo [Information & Computing Lab], 36310 Vigo, Spain; avilas@det.uvigo.es (A.F.V.); rebeca@det.uvigo.es (R.P.D.R.); iclab@det.uvigo.es (H.O.P.)
* Correspondence: m.mohamed@aast.edu

Abstract: Food retailing is now on an accelerated path to a success penetration into the digital market by new ways of value creation at all stages of the consumer decision process. One of the most important imperatives in this path is the availability of quality data to feed all the process in digital transformation. However, the quality of data are not so obvious if we consider the variety of products and suppliers in the grocery market. Within this context of digital transformation of grocery industry, *Midiadia* is a Spanish data provider company that works on converting data from the retailers' products into knowledge with attributes and insights from the product labels that is maintaining quality data in a dynamic market with a high dispersion of products. Currently, they manually categorize products (groceries) according to the information extracted directly (text processing) from the product labelling and packaging. This paper introduces a solution to automatically categorize the constantly changing product catalogue into a 3-level food taxonomy. Our proposal studies three different approaches: a score-based ranking method, traditional machine learning algorithms, and deep neural networks. Thus, we provide four different classifiers that support a more efficient and less error-prone maintenance of groceries catalogues, the main asset of the company. Finally, we have compared the performance of these three alternatives, concluding that traditional machine learning algorithms perform better, but closely followed by the score-based approach.

Keywords: grocery industry; digital transformation; classification; deep learning; machine learning; probabilistic ranking

Citation: Hafez, M.M.; Fernández Vilas, A.; Redondo, R.P.D.; Pazó, H.O. Classification of Retail Products: From Probabilistic Ranking to Neural Networks. *Appl. Sci.* **2021**, *11*, 4117. https://doi.org/10.3390/app11094117

Academic Editor: Mauro Castelli

Received: 4 March 2021
Accepted: 7 April 2021
Published: 30 April 2021

1. Introduction

According to [1], digital transformation facilitates new ways of value creation at all stages of the consumer decision process: pre-purchase (need recognition, information search, consideration or evaluation of alternatives), the purchase (choice, ordering, payment), and the post-purchase (consumption, use, engagement, service requests). This value creation is especially relevant given the very competitive nature of the retail market to gain a larger market share. Although digital transformation [2] is being addressed from different fields: multi-channel solutions, user modeling, internet of things, etc. all rely to some extent on the availability of information on operations, supply chains, and consumer and shopper behaviors. This information is the raw material for data analysis as a central driver towards digital transformation. Food retailing is now on an accelerated path to a success penetration into the digital market and manufacturers and retailers must prepare to achieve six digital imperatives [3,4]: (1) Integrate the digital offerings with their brick-and- mortar operations; (2) Forecasts to increase operational efficiency; (3) Optimizing omnichannel marketing and promotions; (4) Fixing inaccurate master data and (5) a single comprehensive view of customer insights; and (6) Integrating digital and in-store shelf capabilities.

This paper focuses on the most supportive digital imperatives, that is, the availability of quality data (accurate, complete and well-maintained data) in order to feed all the visible

processes in digital transformation. The quality of data are not so obvious if we consider the variety of products and suppliers in the grocery market. As a remarkable example, Walmart is the largest grocery retailer with more than 10,000 retailer units all over the world and around 3000 suppliers and 20 million food products. In addition, the Walmart Marketplace gives access to third party retailers who would like to offer their products to more than 90 million unique visitors who shop on Walmart.com every month. With this scale of locations, online users, providers, and third-party retailers, maintaining the quality of the information is an issue to address every day. Moreover, Walmart provides suppliers up-to-date sales data. Having the magnitude of Walmart or not, the challenge into a successful transition into the grocery digital market is to get the right data, process it at a high speed, and obtain some valued of it.

As well as in other sectors, like financial or consumers in general, the grocery industry is harnessing digital means to innovate through data-drive business models. This will allow us to optimize processes and operations as replenishment and pricing and, at the same time, offer superior convenience in online groceries [5]. Of course, it should also be applied in brick and mortar by redesigning their strategies towards omnichannel retailing. The proliferation of these data drive models brings the emergence of data vendors also in the grocery industry. Precisely, this is where *Midiadia* (https://www.midiadia.com/, access on 5 March 2018) comes in. This Spanish company works on converting data from the retailers' products into knowledge with attributes and insights from the product labels. The so-called *MidiadiaTECH* provides the retailers with deep knowledge about their stock and the customers with a new customized shopping experience. While *Midiadia* extracts the information from the labels of the products, the categorization of the products is manually carried out. Since there are a lot of manufacturers and a lot of different sizes of shops, there is a considerable dispersion of products. Moreover, the list of products changes constantly, so it must be managed properly, taking into account the complex European legislation [6] as well as the national [7] and regional [8] legislation dealing specifically with groceries. Therefore, the main contribution of our approach is providing retail companies with an automatic categorization solution for new products. This classification is uniquely based on the description of each product. The methodology can be horizontally applied to different domains assuming there is a previously known taxonomy or hierarchy of categories. Thus, it would be applied to recruitment processes, travel decision support systems, investment, etc.

In this context, our objective is to provide a solution to automatically categorize the constantly changing products in the market. Being more specific, *Midiadia* has defined a food taxonomy and our purpose is to automatically assign which is the right variety/category for a new product. This task must be done exclusively using the information provided by the product labeling and packaging. This would help to maintain the *Midiadia* data in a more efficient and less error-prone way. With this aim, we have defined different classifiers based on three different approaches: a score-based ranking method (based on BM25), machine learning algorithms (such as K-Nearest Neighbors (KNN), fuzzy K-Nearest Neighbors (FKNN)), eXtreme Gradient Boosting (XGBoost), and deep neural networks (particularly Multilayer Perceptrons (MLP)). After comparing the results, we can conclude that, if we offer only one value for the Variety, then FKNN is the best approach, but if we offer two or three options (leaving the decision between them to the experts in the company), then, the best approach is the score-based ranking method, closely followed by FKNN.

The paper is organized as follows: Section 2 describes previous work related to classification in retailing from a customer's point of view; Section 3 described the dataset provided by *Midiadia*; Section 4 provides an overview of the problem to be solved. Then, the four classifiers are introduced: the score-based classifier (Section 5), the nearest neighbors' classifier (Section 6), an extreme gradient boosting classifier (Section 7), and the neural network classier (Section 8). Finally, Section 9 shows evaluation and results and Section 10 concludes our work.

2. Related Work

Automatic retail classification is an essential task for industry, since it could reduce the huge amount of human labor needed in the product chain, from distribution to inventory management. Within the traditional market field (stores and supermarkets), a considerable number of approaches in the specialized literature focus on recognizing retail products on shelves, which means that the automatic classification approaches are usually based on the product appearance. This will be very useful for retailers, who can audit the placement of products, and, for costumers, who may obtain additional information about products by taking a simple picture of the shelf. These approaches usually rely on computer vision techniques [9–12] and face challenging issues such as the similar appearance in terms of shape, color, texture, and the size of different products. In order to overcome these problems, other research combines the information obtained from image analysis with other information based on statistical methods in order to provide a fine-grained retail product recognition and classification [13]. Advances in this area can help with logistic problems like shelf space planning [14].

Apart from these specific problems for image analysis, there is a well-known problem with retail products: the overwhelming amount of different products and categories in any supermarket. A typical supermarket could have more than a thousand different products [15], and this number increases when we talk about merging catalogues from different supermarkets. Because of the popularization of e-commerce, this large number of different products are usually managed through multilevel category systems. Two relevant problems arise within this field: first, obtaining relevant information (characteristics and description) of retail products and, second, the automatic retail classification based on this data. Some approaches have focused on the first topic and try to automatically extract relevant metadata from the analysis of product images [16,17]. However, retailers usually acquire this data from suppliers and/or third parties to complete the product information on their websites. Precisely, the research work introduced in this paper is being used by one of these third parties, a Spanish SME (*Midiadia*) that offers relevant data to retailers for their e-commerce platforms. Both retailers and third parties need to have automatic classification systems based on the information of the retail products to have consistent multilevel category structured data.

In [18], this problem is tackled by applying natural language processing techniques to the text titles. One of the problems this research faces are the typical absence of grammar rules in product titles or names, so they apply short text classification methods, such as character embedding, to overcome this issue. In [19], a United States patent, a machine learning classifier helps to automatically select one or more categories for a product. The information is extracted from the metadata fields of the product description, and it is used to enrich a recommendation engine. In [20], another United States patent, a new method to automatically categorize a product in an electronic marketplace is provided. It is based on parsing the category information (provided by the retailer) to obtain a first category identifier. This information is used to search the most adequate category identifier within the available taxonomy. This methodology takes into account information of expired category identifiers. The authors in [21] perform an interesting analysis of text classification and offer a solution to automated classification on e-commerce by using distributional semantics. The idea here is to create a taxonomy by first applying a modification of the Bag-of-Words model [22]. This approach creates feature vectors from the description of the products and combines three types of predictors: the *Path-Wise Prediction Classifier*, in which each path in the taxonomy tree is considered as a class; the *Node-Wise Prediction Classifier*, in which each node in the taxonomy tree is considered as a class; and the *Train Depth-Wise Node Classifier*, which combines different classifiers on each level of the taxonomy tree, where each node represents a different class inside these sub-classifiers. There is also an interesting and different analysis [23] that studies how shape and packaging impact on brand status categorization, concluding that slender packages are more likely to be

categorized as high-end products (high brand status) than those in short, wide packages (low brand status).

Deep learning technologies are also used in this field (retail) to enhance the information analysis processes. In fact, [24] offers a survey of the most recent works in the field of computer vision applied to automatic product recognition combined with deep learning techniques. Other interesting analysis that supplements the previous one is the work in [25]. Here, the authors focus on fine-grained image analysis (FGIA) combined with deep learning to automatically assign subordinate categories in specific application fields. Although species of birds or models of cars are explicitly mentioned, the bases might also be applied to retail commerce as well. In the paper, the FGIA techniques are organized into three main categories: recognition, retrieval, and generation. When dealing with natural language, instead of images, there are also research works that apply deep learning to deal with automatic analysis of this kind of data. For instance, in [26], the authors face the automatic analysis of natural language to infer polarity and sentiment information from reviews in social media. Their approach is based on vector machine, logistic regression, and advanced deep neural network models. It explores dependency-based rules to extract multilingual concepts from a mixture of sentences in two languages (Persian and English) to detect subjectivity and sentiment in sentences.

Other approaches take into account the customers for gathering relevant information about products. For instance, the authors of [27] study the influence that product categorization and types of online stores have on customers, mainly using hypothesis testing. The research follows three steps: the first one is a test to see if the online shopping preferences of the customers vary according to the type of product, without taking into account the kind of online retail store; the second step is a test on the interaction effects of both the online store and the types of products; and the last step is a hypothesis testing, which results in the identification of the attributes considered relevant by the customers when they purchase retailing products. Along this line, the proposal in [28] analyzes the prediction of customer behavior in the purchase process with machine learning techniques used in real-world digital signage viewership data. The data were obtained from a camera that recorded a clothing store and additional features that were added manually. In this case, the customers can be considered in buying groups—for example, a family that is going to purchase a product. The goal was to classify the customers in six categories: (i) *Initiator*, the person from the buying group who recognizes the need and finds the product that the group needs, (ii) *Influencer*, the person whose opinion has a relevant effect on the purchase decision, (iii) *User*, the final customer who will use the product or service, (iv) *Decider*, the one who makes the final decision about whether or not to purchase a product, (v) *Purchaser*, the person who pays for the purchased product and also determines the terms of the purchase, and (vi), *Passive influencer*, a member of the buying group but is not involved in the purchase. In [29], the approach is exclusively customer-behavior based, providing a new clustering mechanism able to organize retail products into different categories.

In the specialized literature, there are different approaches for automatic categorization of retail products. Computer vision solutions are mainly oriented to traditional supermarkets and have to face important accuracy problems because of the huge amount of different products and the high similarity in the packaging of products within the same categories. In the E-commerce field, there are different needs, but automatic classification based on product characteristics is a relevant one. The number of different products is overwhelming, especially if we are working on combined catalogues, i.e., catalogues from different supermarkets and retailers, as it is our case. Analyzing the product title or name is not enough and taking into account customer behavior is not always available. Thus, to the best of our knowledge, there are not other studies that have faced the automatic classification problem of combined catalogues, without data from customers and only based on the product description (name, trademark, and ingredients). Our proposal fills in

an interesting gap for third parties specialized on enriching the retailers catalogues with smart data.

3. Dataset

The dataset used in this work is provided by *Midiadia*, which consists of a collection of 31,193 products. The list of attributes considered in this study are shown in Table 1. The *European Article Number* (EAN) is a standard describing a barcode symbology and numbering system used in global trade to identify a specific retail product type, in a specific packaging configuration, from a specific manufacturer. *Category*, *Subcategory*, and *Variety* represent the 3-levels (hierarchy) the company considers to organize the catalog into a taxonomy. *Brand* is an identifying name for a product manufactured by a particular company. Finally, the other terms are coherent to the EU regulation, as follows. Just to clarify the meaning of these terms, a sample extract (Although product attributes are in Spanish, in the following, the translated version of fields and values are used to improve readability) of the dataset is shown in Table 2:

- 'Customary name' means a name which is accepted as the name of the food by consumers in the Member State in which that food is sold, without that name needing further explanation.
- 'Legal name' means the name of food prescribed in the Union provisions applicable to it, or, in the absence of such Union provisions, the name provided for in the laws, regulations, and administrative provisions applicable in the Member State in which the food is sold to the final consumer or to mass caterers.
- An 'Ingredient' means any substance or product, including flavorings, food additives, and food enzymes, and any constituent of a compound ingredient, used in the manufacture or preparation of food and still present in the finished product, even if in an altered form; residues shall not be considered as 'ingredients'. Moreover, the EU regulation on the provision of food information to consumers.
- 'List of ingredients' (suitable heading which includes the word 'ingredients') shall include all the ingredients of the food, in descending order of weight, as recorded at the time of their use in the manufacture of the food.

Table 1. Product attributes in the dataset.

Field	Levels	Description
EAN		Unique Product Number
Category	16	1st Level Category
Subcategory	62	2nd Level Category
Variety	159	3rd Level Category
Brand	3015	Product Brand
Name	11,139	Product Customary Name
Legal Name	8442	Official product denomination regarding the European Union provisions
Ingredients		List of Ingredients in the product

The dataset also includes other attributes, such as packaging size, healthy claims, calories, etc. However, these attributes are aimed to recommend alternative or replacement products in a specific variety, but the problem we are trying to solve is a different one: when a new product is incorporated into the company food catalog, all the information in the product labeling and packaging is automatically extracted, but the decision of which Variety is the one that corresponds to the new product is currently manually done. Thus, we need to automatically decide to which Variety a new product belongs, so we have focused only on attributes directly related to the kind of food, discarding the others. According to this selection, rows with blank fields in all the selected columns of Table 1 are removed. This entails that we have worked with a dataset with 20,888 products, hereinafter the MDD-DS (*Midiadia* DataSet).

Table 1 also includes the number of different values that exist in the variable considered MDD-DS. Thus, as a categorical variable, the dataset shows 159 levels for *Variety*. The fact of knowing in which upper level of categorization is each variety can be helpful for classification purposes. *Brand* can bring useful information apart from the *Name* and *Legal Name* since they are defining the kind of product. Regarding *Ingredients*, it is a list of ingredients that require further processing in order to support the classification. Figure 1 shows the distribution of products per variety as well as the statistical parameters of this distribution. The average number of products per variety is around 200, being the 1st quartile 116.5, so that 75% of the varieties have a number of assigned products above 213 products.

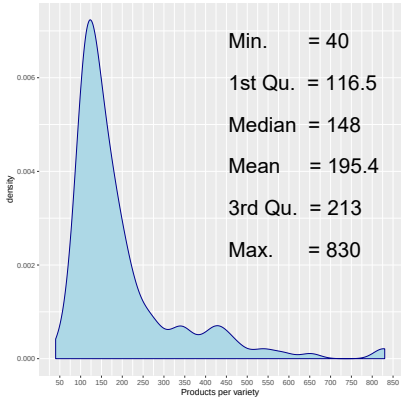

Figure 1. Number of products per variety: Density plot.

Table 2. Extract of the MDD-DS.

Produt N.	**1**	**2**	**...**	**31,193**
EAN	62	1292	...	84654
Category	Fresh	Beverages	...	Snacks and nuts
Subcategory	Greens and Vegetables	Strong Alcoholic Beverages	...	Nuts
Variety	Greens and Vegetables	Ron	...	Seeds
Mark	Generics	Pujol	...	Facundo
Name	Raw leek	Golden Ron	...	Gian Seeds
Ingredients	Leek	Golden Ron 40% vol. alc.	...	Sunflower seeds and salt (4%)
LegalName	Leek	Ron	...	Roasted and salty giant sunflower seeds

4. Proposed Model

In order to automatize the time-consuming classification process, this paper proposed a multi-label classification methodology to map inputs (new products) to a maximum of three labels from all the varieties considered (The number of varieties is a restriction of the dataset). According to the scheme in Figure 2, the deployment of our solution (bottom part of the figure) takes *Name*, *Legal Name*, and *Ingredients* of a new product (NP) as input observational variables. Then, a multi-label classifier predicts the varieties' score of NP that is the classification score for NP to be a member of each variety (class) $C_1 \dots C_n$ in the taxonomy. The output is defined as the three varieties C_1, C_2, C_3 with higher scores. From these three varieties, the sets $Top\_1 = C_1$; $Top\_2 = \{C_1, C_2\}$ and $Top\_3 = \{C_1, C_2, C_3\}$ are defined. This paper describes the methodology and experiments to obtain the classifiers by using the MDD-DS (described in Section 3). The modeling

methodology consists of three main steps: (1) preprocessing the data and potentially reducing their dimensionality; (2) learning the classifier model: four different classifier models, such as score-based model, nearest-neighbors models (KNN: K-Nearest Neighbors and FKNN: Fuzzy K-Nearest Neighbors), XGBoost model and deep learning (MLP or Multi-Layer Perceptron Network); (3) testing the classifier models in order to compare them and estimate the quality measures of our approach. In step (1), and in order to apply machine learning and deep learning approaches, we include a new sub-step for dimensionality reduction based on PCA (Principal Component Analysis).

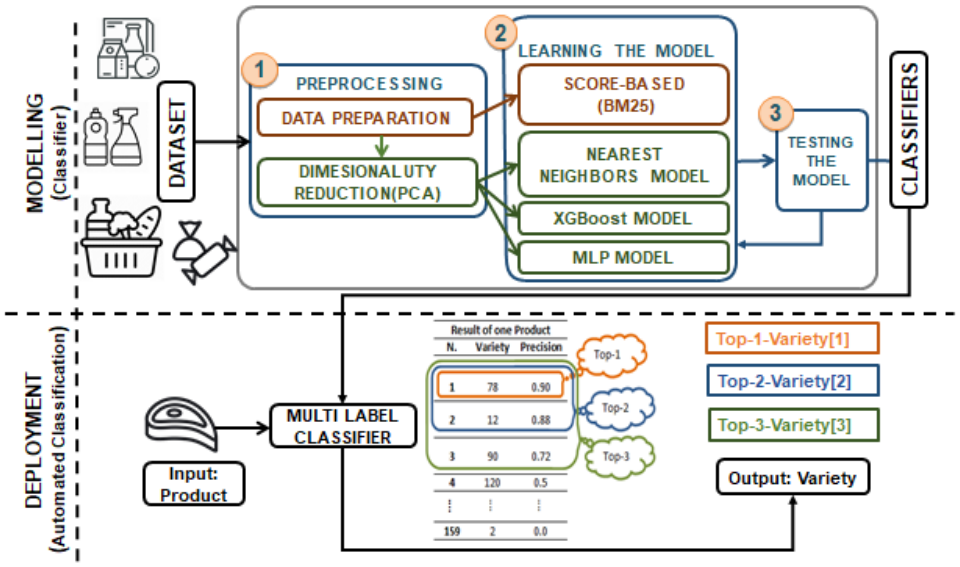

Figure 2. Description of the proposed model definition and evaluation.

The main idea of this paper is to select the best classification algorithm for retail products among a wide range of possibilities that goes from probabilistic ranking approaches to deep neural network ones. With this aim, we have selected different techniques within each approach: a score-based model in probabilistic ranking based on the BM25 algorithm; K-Nearest Neighbors (KNN), Fuzzy K-Nearest Neighbors (FKNN, a fuzzy version of the KNN classification algorithm), and eXtreme Gradient Boosting (XGBoost) in machine learning; and Multi-Layer Perceptron in deep neural networks. This selection was made taking into account the advantages of each algorithm. Consequently, BM25 was selected because of its efficiency, since its performance is well known in different ad-hoc retrieval tasks, especially those designed by TREC [30] (see Section 5 for further details). Within the machine learning field, we have selected KNN and FKNN because both of them need to perform predictions in order to learn new knowledge. Therefore, new products can be added because they will not affect the accuracy of the algorithms [31]. In FNNN, we have proposed to use the fuzzy *memberships* values of the samples to adjust the contribution values, since this algorithm delays the decision to assign a sample to a certain class across *memberships* [32,33] (see Section 6 for further details) , and XGBoost for predictive modeling as a sturdy and efficient machine learning method for prediction. XGBoost is a boosting algorithm that belongs to supervised learning, which is an ensemble algorithm based on gradient boosted trees. It integrates the predictions of "weak" classifiers to achieve a "strong" classifier (tree model) through a serial training process. It is additionally a comparatively new technique; however, wonderful results are achieved in several classification tasks (see Section 7 for further details). Finally, within the deep learning area, we have selected the MLP algorithm. This is a simple but efficient algorithm that represents a broad family and that was used because of its ability to adaptive learn to perform tasks based on the data provided for the initial training or experience. In addition, it is based on a

defining a decision function that it is directly obtained through training [34] (see Section 8 for further details).

Preprocessing

As previously mentioned, MDD-DS consists of 20,888 products, which is the result of cleaning the original dataset (31,193 products) before applying the classification algorithm. This first cleaning procedure consisted of removing all the products with no name, no ingredients, and no legal name. After that, MDD-DS is pre-processed by extracting all the meaningful words for the attributes 'Name', 'Legal Name' and 'Ingredients'. For each product p, the three attributes are merged into a single text which aims to describe the product $des(p)$. This description ($des(p)$) is obtained after a cleansing process described [35] step by step in Algorithm 1: (i) parenthesis are transformed into blank spaces, (ii) numbers, stop-words, punctuation, and extra spaces are removed; (iii) all letters are converted to lowercase, and (iv) repeated strings are removed.

Algorithm 1 Preprocessing pseudocode

```
procedure PREPROCESS(MDD-DS)
    cleaning(MDD_DS)
    product_words[] ← new_list(m)
    all_products_words ← new_vector (0)
    for i ← 1 : m do
        p ← MDD-DS[i, ]
        des(p) ← concatenate(
        p[Name], p[Legal Name], p[Ingredients])
        des(p) ← transform_brackets_into_space(des(p))
        des(p) ← transform_into_lowercase(des(p))
        des(p) ← remove_numbers(des(p))
        des(p) ← remove_stopwords(des(p))
        des(p) ← remove_punctuation(des(p))
        des(p) ← remove_extra_spaces(des(p))
        des(p) ← split_by_spaces(des(p))
        des(p) ← remove_duplicates(des(p))
        des(p) ← remove_empty_words(des(p))
        product_words[i] ← des(p)
        all_products_words
        ← all_products_words ∪ des(p)
    end for
end procedure
```

Then, words are split, and a vector of words $product\_words(p)$ is generated, an example is shown in Table 3. Additionally, we also create the *all_products_words* (for all the products in MDD-DS). This is another vector of words that contains as many elements as different words in the products' vectors (from the fields Name, Legal Name, and Ingredients). After preprocessing, *all_products_words* contains a total of 11,359 unique words, an example is shown in Table 4.

Table 3. Examples of *product_words* for every p.

Product Id	Product-Vector
1	['oil', 'olive', 'virgin', ...]
...	...
218	['pepper', 'sweet', 'glass', ...]
...	...
20,888	['biochips', 'cheese', 'spelt', 'oil', ...]

Table 4. Extract from *all_products_words*.

['oil', 'olive', 'virgin', 'extra', 'arbequina', 'extraction', 'cold', 'filter', 'quality', 'high', 'origin', 'gluten', 'fondant', 'picual', 'spanish', 'refined', 'e420ii', 'e1520', 'liquid egg', 'gluten', 'e501i', 'e281', 'powder', 'bicarbonate', 'fructose', 'e218', , 'icing', 'e492', 'fondant', 'aluminum', 'e155', . . .]

With the information provided by the *all_products_words*, we obtain the product matrix $X[m, n]$, which is mathematically defined as follows. Let $\vec{w}$ be the n-dimensional vector obtained from *all_products_words* such that $\vec{w} = (w_1, \ldots, w_n)$ and $\forall\ k \in [1, \ldots, n],\ w_k$ is a string $\in$ *all_products_words* and $N = \dim(all\_products\_words)$ is the total number of different meaningful words in the dataset (after cleaning).

- The product matrix $X[m, n]$ is an MxN matrix, where each column represents a word $w_i \in \vec{w}$, and each row represents a product p of the dataset so that M is the total number of products
- Let $\vec{p}$ be the *product_words*[] list such that each element contains a vector $\vec{p}_i$, hence $\vec{p} = (\vec{p}_1, \ldots, \vec{p}_m)$. Likewise, $\vec{p}_i = (p_i[1], \ldots, p_i[l])$, where $l = \dim(product\_words[i])$, length of the vector in the i element of the list $\vec{p}$. Note that $\forall\ k \in [1, \ldots, l],\ p_i[k]$ is a string.

  Therefore,

$$X[i, j] = \begin{cases} 1, \text{if } \vec{w}[j] \in \vec{p}[i] \\ 0, \text{if } \vec{w}[j] \notin \vec{p}[i] \end{cases} \quad (1)$$

where

$$\vec{w}[j] \in \vec{p}[i] \iff \exists\ k\ /\ \vec{p}_i[k] = \vec{w}[j] \quad (2)$$

This means that $X[i, j]$ is 1 if the product i has the word j in its *product_words[i]*; otherwise, the value will be 0. For evaluation purposes, we include the variety of the product in an additional column, as shown in Table 5.

Table 5. Example of a *product matrix*.

***p* Id**	**'oil'**	**'olive'**	**'virgin'**	$\cdots$	**Variety**
1	1	1	1	$\cdots$	'Virgin olive oil and extra virgin olive oil'
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
20,888	1	0	0	$\ldots$	'Salads'

After this data preparation, the next step is applying dimensionality reduction in order to face the learning models based on traditional machine learning and deep neural networks. As broadly known, dimensionality reduction improves computational efficiency and accuracy of data analysis. Dimensionality reduction is usually performed using two approaches: (i) Feature selection and (ii) Feature extraction. On the one hand, feature selection [36] removes variables that are not so relevant for the problem that must be faced (classification, clustering, prediction, etc.). There are some interesting techniques within this field, such as Analysis of Variance (ANOVA) [37], Minimum Redundancy Maximum Relevance (mRMR) [38], and Max-Relevance-Max-Distance (MRMD) [39]. On another hand, feature extraction [40] tries to identify and to find a smaller set of features from the available data that are relevant for the problem. This approach entails that it is possible to reformat, combine and/or transform primary features (variables) into new ones until a new data set only with relevant data are obtained. We have opted to this approach (feature extraction) because our previous analysis showed correlation among the available features, but it was not easy to perform a direct selection of relevant variables (because of the overwhelming number of primary features).

There are some interesting techniques within this field, such as Principal Component Analysis (PCA) [41], Linear Discriminant Analysis (LDA) [42], and Autoencoders [43]. PCA and LDA are both linear transformation techniques used for dimensionality reduction and visualization, but while PCA is an unsupervised technique, LDA is a supervised one. PCA searches for attributes with the most variation and ignores class labels. In contrast, LDA maximizes the separation of known categories. An autoencoder is a neural network, with as many output units as input units and at least one hidden layer that is used to learn a representation (encoding) for a data set by training the network to avoid or ignore signal noise. This provides a nonlinear transformation that reduces the dimensionality of the data. The authors of [44] evaluate the performance of this technique by comparing an autoencoder with other models that can perform a similar reduction task, such as PCA, LDA, etc. The results showed that PCA and LDA are relatively more stable than autoencoder (low stability implies that different results might be obtained when replicating the experiment). To sum up, we opted for a stable reduction technique and, more precisely, an unsupervised one that is PCA. Both stability and the absence of a training step before applying reduction eases the deployment of automatic classification in a real retailing scenario, where the product catalogue is intrinsically dynamic.

PCA uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of linearly uncorrelated variables. In addition, PCA collects all the required features for MDD-DS, which removes correlated features [45], improves algorithm performance [41], reduces over-fitting [46], and improves visualization [46]. To achieve the research goal, our study compares the results obtained with different values for PCA (400, 500, 600, 700, and 800) to select the best option (in Sections 6–8).

5. Score-Based Model

Our first approach is based on the bag-of-words BM25 model [47] (BM for Best Matching), widely used in information retrieval to rank documents based on a query of words. The BM25 equation considers the frequency of occurrence of the words in the documents, smoothing out the weighting in favor of how distinctive they are:

$$\sum_{w \in Q \cap D} \frac{(k+1)c(w,D)}{c(w,D) + k(1 - b + b\frac{|D|}{avdl})} \cdot \log \frac{N - df(w) + 0.5}{df(w) + 0.5}, \quad (3)$$

where w is the word analyzed, $c(w, D)$ is the number of times the word w appears in the document D, Q is the query, N is the number of documents, D is the document analyzed, $df(w)$ is the number of documents that contain the word w, k is a constant with the value 1.2 (typical value obtained from TREC experimentation [30]), $|D|$ is the total number of words in the document D, b is a constant with the value 0.75 (typical value obtained from TREC experimentation [30]), and $avdl$ is the average number of words per document.

In order to apply this probabilistic model to the classification of retail products into varieties, after obtaining the *product matrix*, we generate a *variety matrix*. This matrix will have as many columns as different words exist in the dataset—i.e., the same number of columns as the *product matrix*. The number of rows will be the same as the number of different varieties obtained from the dataset.

- Let Y be the *variety matrix* (y x n).
- Let $\vec{z}$ be the y-dimensional vector of integers where each one represents a variety. Thus, $\vec{z} = (z_1, \ldots, z_y)$.
- Let $\vec{v}$ be the m-dimensional vector that contains an integer that represents each product variety. Thus, $\vec{v} = (v_1, \ldots, v_y)$.

Therefore:

$$Y[i,j] = \sum_{\vec{v}[k]=\vec{z}[i]} X[k,j] \quad (4)$$

In Equation (4), the value of $Y[i, j]$ will be the number of times the word j appears in the *variety* i (i.e., the set of all the products belonging to it and represented by the index k), as shown in Table 6.

Table 6. Example of a *variety matrix*.

	'oil'	**'olive'**	···	**'aromatic'**
Almonds / hazelnuts	0	0	...	0
⋮	⋮	⋮	···	⋮
Virgin and Extra virgin Olive Oil	233	224	...	0
⋮	⋮	⋮	···	⋮
Juices (100%)	0	0	···	0

To obtain the variety V of a product, we apply the modification of the BM25 model depicted in (Equation (5)). This equation gives a realistic smoothing when the term frequency is high and rewards low-frequency words even more than the one in (Equation (3)). We consider two modifications with regard to the original BM25 equation. First, one unit is added to both the nominator and the denominator. The reason for the addition in the denominator is that the term of the query may be missing in the analyzed document within the corresponding iteration of the addition, so $vf(w)$ can be equal to zero. Adding a unit instead of 0.5 makes more sense in our scenario because it is more realistic to add integers (i.e., entire words) than a rational number; the lower bound is also equal to the unit. At the same time, the unit added to the numerator balances out the fraction. The second modification is that the factor $c(w, V)$ is a constant. This is due to the fact that the word duplicates contained in the query are removed and the terms are unique. Thus, $c(w, V)$ will always be equal to 1:

$$\sum_{w \in P \cap V} \frac{(k+1)c(w,V)}{c(w,V) + k(1 - b + b\frac{|v|}{avvl})} \log\left(\frac{N+1}{vf(w)+1}\right) \quad (5)$$

Once the modified BM25 equation is repeated for all the varieties, the variety was chosen as a result for a product is the one ranked the highest.

6. Nearest-Neighbors Model

The nearest neighbor (NN) [48] rule is a nonparametric method for classification based on instances. Taking into account the problem in Figure 2, a product in the MDD-DS p follows the definition $\{p_1, p_2, \ldots, p_n, v\}$, where n is the number of the aforementioned product attributes, and v is its assigned product variety.

The KNN approach [31] classifies an unlabelled product on the majority of similar sample products among the k-nearest neighbors in MDD-DS that are the closest to the unlabelled product. The distances between the unlabelled product and each of the training product samples are determined by a specific distance measure. Therefore, the variety of new product $\vec{p}$ is assigned to the most common variety among its closest k training samples according to the attributes considered in the product vectors. More formally, let $\vec{unlabel} - p$ an unlabelled product, the decision rule predicts a variety $\hat{v}$ for the $\vec{unlabel} - p$ according to the variety v of the majority of its k nearest neighbors; in the case of a tie, $\hat{v}$ is given by the closest nearest neighbor that belongs to one of the tied varieties.

One of the problems for KNN is to give equivalent importance to each of the samples to determine the class membership, neglecting the typicality among them. Alternatively, some challenges arise for classification in a high dimensionality dataset where it is complex to discriminate between classes. This problem is due to the fact that the number of training samples and the increase in dimensionality grow overwhelmingly. The aforementioned

drawbacks have been addressed in the literature by Fuzzy Sets Theory to allow imprecise knowledge to be represented and fuzzy measures to be introduced.

Consequently, FKNN is a fuzzy version of the KNN classification algorithm [32]. Fuzzy classifiers delay the decision to assign a sample to a certain class by using *memberships*. In terms of our problem in Figure 2, FKNN assigns to an $\vec{unlabel\_p}$ a variety's fuzzy membership (not just true or false) by taking into account the *distance* from $\vec{unlabel\_p}$ to its k nearest neighbors and those neighbors' memberships for this variety. More formally:

- Variety membership is defined for each product $\vec{p}$ in MDD-DS with a value in $[0, 1]$ for each variety v. Although the details can be found in [32], the main idea is assigning to each product $\vec{p}$ a membership value for a variety v not just according to the assigned variety (true value) but also according to the assigned variety for its k nearest neighbors.
- Besides given an unlabeled product $\vec{unlabel\_p}$, each neighboring product $\vec{p} = \{p_1, p_2, \ldots, p_n, v\}$ votes for every variety $\{v_1, v_2, \ldots, v_m\}$ by using its variety memberships (not just for its assigned one v). Again, the details can be found in [32], but votes are weighted according to the inverse of the distance to $\vec{unlabel\_p}$ and added. The estimated variety $\hat{v}$ for $\vec{unlabel\_p}$ is the variety with the greatest combined vote.

The main benefit of using the FKNN model may not be decreasing the error rate, but, more importantly, the model provides a degree of certainty that can be used with a "refuse to decide" option. Therefore, objects with overlapping classes can be detected and processed separately.

Since KNN and FKNN are computationally expensive algorithms, we apply Principal Component Analysis (PCA) to MDD-DS to reduce its dimensionality. In our experiment, the original set of product attributes are transformed into a smaller set (the so-called principal components) so that the reduced dataset MDD-DS-reduced still contains most of the information in MDD-DS. More specifically, we evaluate the performance of KNN and FKNN applying PCA and selecting 400, 500, 600, 700, and 800 principal components. As a result, the *product matrix* is replaced by a PCA-reduced matrix. Given the variance retention for the following number of eigenvectors $(400, 0.796452)$; $(500, 0.825814)$; $(600, 0.84735)$; $(700, 0.865117)$; $(800, 0.879153)$, the experiments in this paper have been carried out over a reduced dataset of 600 principal components, referred to as MDD-DS-PCA-600.

Optimal k and Distance for KNN and FKNN

The key to success in KNN and FKNN are finding the optimal value of k and the distance function. Usually, the k parameter is chosen empirically depending on each problem, different numbers of nearest neighbors are tested, and the parameter with the best performance (accuracy) is chosen to define the classifier. Choosing the optimal k is almost impossible for a variety of problems since the performance of a KNN classifier varies significantly when changing k and the change of the *distance* metric used [49].

Therefore, the decision about the *distance* measure also plays a vital role in determining the final result of the classification. Although Euclidean distance is the most widely used distance metric in NN-based classifiers, we have applied a variety of *distance* metrics in MDD-DS. We call the "best distance metric" the metric which allows KNN, FKNN to classify test samples with the highest accuracy, recall, precision, and F-score. The definition of the distances considered in this paper (Spearman, Cosine, Correlation, Euclidean, Cityblock, Chebychev, Hamming, Jaccard, and Seuclidean) are described in Table A1, included as an Appendix A.

To determine the optimal value of k and *distance* for Nearest-Neighbors Models KNN and FKNN, we deploy the pseudocode in Algorithm 2 to try different combinations in two loops (odd k values initialized from $[1$ *to* $30]$ and the different *distances* used) over the reduced dimension MDD-DS for 600 principal components. This value was selected taking into account that, with 600 principal components, we obtain a value around 85% of retained variance, as a criteria to choose the appropriate number of principle components.

In addition, we have perceived that the results do not significantly improve if the number of principal components increase. Since we pursue obtaining the optimal pair of *k* and *distance* not only for classification per se (*Top*_1) but also for *Top*_2 and *Top*_3, the pseudocode obtains the accuracy for all the combinations, as it is shown in Figure 3.

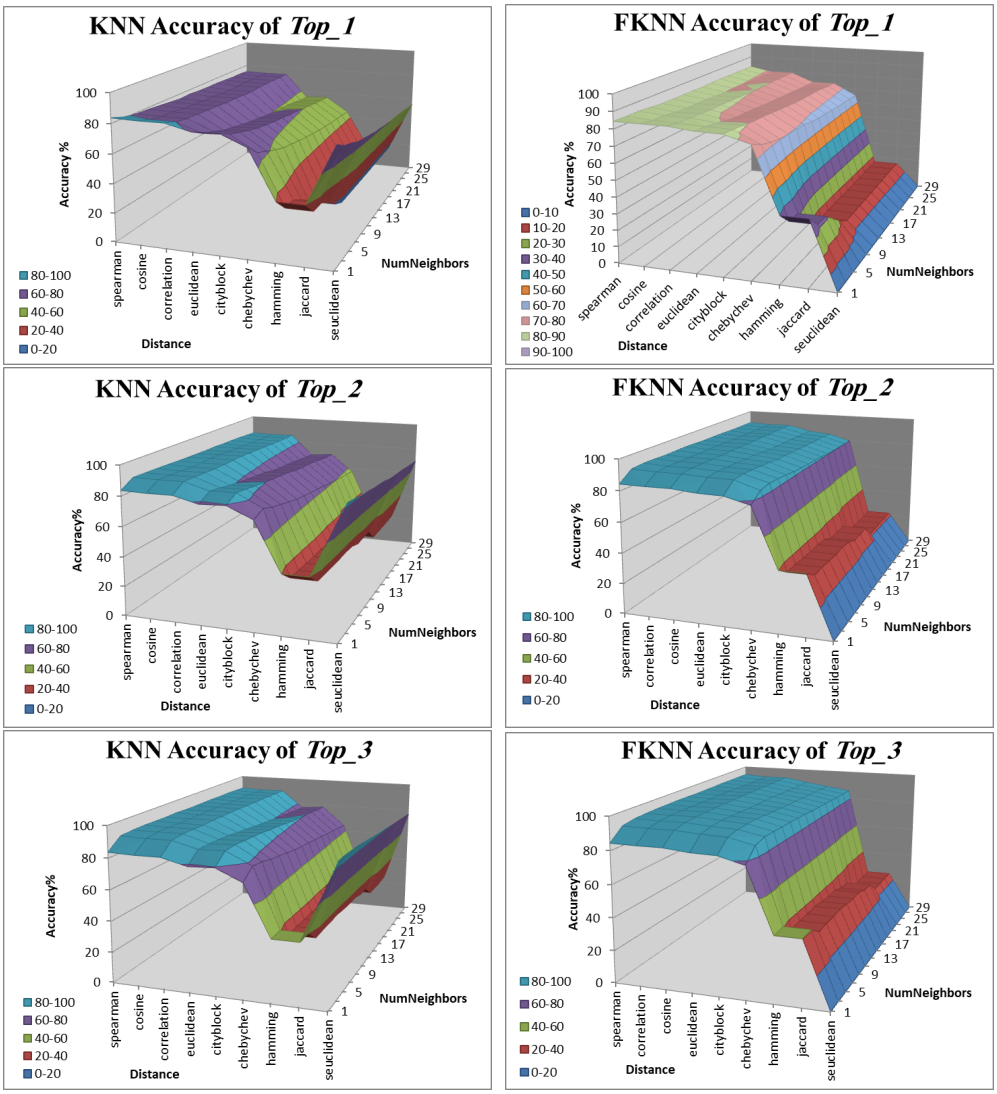

Figure 3. Accuracy of KNN and FKNN for the three approaches: *Top*_1, *Top*_2 and *Top*_3.

The optimal type of the *distance* is Spearman of three approaches of KNN and FKNN. Table 7 shows that the optimal value of *k* and accuracy result for each approach according to Figure 3.

Table 7. Optimal parameters of *KNN, FKNN* for each approach.

	KNN		FKNN	
	k	*Accuracy*	*k*	*Accuracy*
*Top*_1	1	0.8367	1	0.8446
*Top*_2	3	0.8874	13	0.9153
*Top*_3	7	0.9141	23	0.9413

Algorithm 2 Tuning parameter for KNN and FKNN

```
procedure TUNING_PARAMETERS(Train_PCA600, Test_PCA600)
    Dist[] = ['Spearman', 'Cosine', 'Correlation', 'Euclidean', 'Cityblock', 'Chebychev', 'Hamming', 'Jaccard', 'Seuclidean']
    Label C1                                                    ▷ First, Variety
    Label C2                                                    ▷ Second Variety
    Label C3                                                    ▷ Third Variety
    i ← 1
    for K ← 1 : 30 by 2 do
        for J ← 1 : 9 do
            M ← build_model(Train_PCA600, k, Dist[J])
            [C1, C2, C3] ← Predict(M, Test_PCA600)
        end for
        i ← i + 1
    end for
    Top_1 ← C1
    Top_2 ← C1 & C2
    Top_3 ← C1 & C2 & C3
    return Top_1, Top_2, Top_3
end procedure
```

7. XGBoost Model

eXtreme Gradient Boosting (a.k.a XGBoost) [50,51] is an extremely scalable end-to-end tree boosting system, a machine learning technique for classification and regression issues. XGBoost is an additionally boosting algorithm belonging to supervised learning; it uses an ensemble of X classification and regression trees (CARTs); each inside node represents values for attributes test, and a leaf node with scores represents a choice ($X_E^i | i \in 1 \ldots X$).

In terms of our problem in Figure 2, and given $\vec{p} = \{p_1, p_2, \ldots, p_i, \hat{v}\}$, XGBoost can learn using Equation (6), which is $(p_1, p_2, \ldots, p_i) \rightarrow \hat{v}$. Given a set of features $\{p_1, p_2, \ldots, p_i\}$ for $\vec{unlabel\_p}$, XGBoost can predict based on the total of the prediction scores for every tree:

$$\hat{v}_i = \sum_{x=1}^{X} s_x(p_i), s_k \in S \tag{6}$$

where p_i represents the ith training sample of MDD-DS and v_i is the variety corresponding label, s_x represents the score for xth tree, and S represents the set of all X scores for all CARTs. XGBoost adopts a similar gradient boosting as the Gradient Boosting Machine (GBM). Regularization is applied to enhance the ultimate result:

$$LF(\theta) = \Sigma_i l(\hat{v}_i, v_i) + \Sigma_x \Omega(s_x) \tag{7}$$

where $LF(\theta)$ means the total objective function, l represents the differentiable loss function that measures the difference between the prediction of a variety $\hat{v}_i$ and the target label of a variety v_i. The Ω avoids over-fitting, penalizing the complexity of the model:

$$\Omega(s) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{8}$$

where γ and λ are constants that control the degree of regularization, T represents the leaves number in the tree, and w means the weight of each leaf. XGBoost also applies a second-order approximation to extend the loss function and removes the constant term and applies two additional techniques to reduce overfitting further; the details can be found in [50,51].

8. MLP Model

Multilayer perceptron's (MLP) [52] is a deep feed-forward neural network particularly designed for multi-class supervised learning problems. It has one or more hidden layers between the input and the output; each layer fully connected to the next. An MLP network is trained on a set of input–output pairs so that the network learns to model the dependencies between those inputs and outputs. Training involves adjusting the parameters, or weights and biases, of the model to minimize error. Back-propagation is used to make those weight and bias adjustments in relation to error, and error itself can be measured in various ways. Figure 4 shows MLP structure with scalar output.

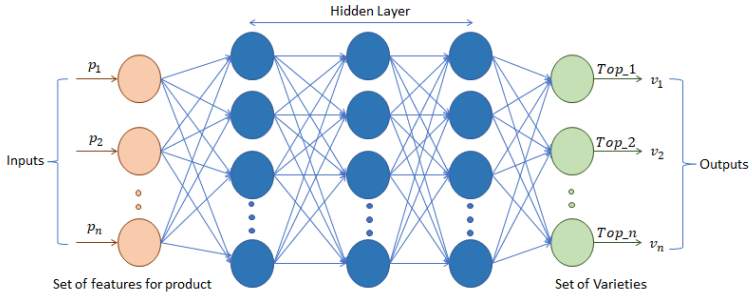

Figure 4. MLP structure.

In terms of our problem in Figure 2, and given $\vec{p} = \{p_1, p_2, \ldots, p_n, v\}$, MLP learns a function $f(\{p_1, p_2, \ldots, p_n\}) \rightarrow v$. Given a set of features $\{p_1, p_2, \ldots, p_n\}$ for $\vec{unlabel\_p}$, MLP can learn a nonlinear function for classification in the varieties space v. For the purpose of comparison, we built-up an MLP network from 600 principal components as a reduced input set. With a Leaning Rate of 0.001, which is its default value (Learning rate is a hyper-parameter that controls how much we are adjusting the weights of our network with respect the loss gradient.), the first experiments showed a significative increase of performance from 1 to 3 hidden layers and, above three hidden layers (3,4,5), the increase in performance slows down while computation time increases significantly.

In practice, as the number of hidden layers determine the ability to generalize, we selected three hidden layers; upper values, apart from increasing computation time, tend to overfit, which leads to reduce classification performance for out-of-sample instances. To obtain the optimal values of the number of nodes per layer N and the times an algorithm visits the data set, training epoch T, we applied the pseudocode in Algorithm 3 over the reduced dimension MDD-DS for 600 principal components (as mentioned, the reason in Section 6). According to the curve in Figure 5, accuracy increases when the number of training epochs significantly increase, and remarkably does not increase when applying 600, 700, and 800 and the same for the number of nodes. Therefore, the two loops can stop at 800; number of nodes $[300, 400, 500, 600, 700, 800]$ and training epochs $[100, 200, 300, 400, 500, 600, 700, 800]$. The result of the evaluation shows that the optimal number of nodes is 800 and the number of training epochs is 600 for the three approaches (Top_1, Top_2 and Top_3).

9. Evaluation and Results

For evaluation purposes, we have implemented the approaches using different tools and programming languages: (i) R (Rstudio) to pre-process the data set and then (ii) Matlab and Python to implement machine learning algorithms and deep neural networks. We have analyzed the usual parameters of each algorithm: (i) k and the distance for KNN and FKNN because these two parameters are the ones that condition the algorithm Section 6; and (ii) the number of hidden layers, the number of nodes and the number of epochs for MLP are the three most relevant parameters that condition the algorithm of neural networks Section 8.

Algorithm 3 Tuning parameter for MLP to the reduce MDD-DS with 600 principal components and with a total of 159 classes (varieties)

procedure TUNING_PARAMETERS(Train_600, Test_600)
    $Learning\_RateLR = 0.001$; $Hidden\_layerHL = 3$
    $Number\_of\_Node\ N$; $Training\_epochs\ T$;
    Label $C1$ ▷ 1st Variety
    Label $C2$ ▷ 2nd Variety
    Label $C3$ ▷ 3rd Variety
    for $N \leftarrow 300 : 800$ by 100 **do**
        for $T \leftarrow 100 : 800$ by 100 **do**
            $Model\ M \leftarrow$ build_model_MLP(Train_600, Test_600, HL, LR)
            $[C1, C2, C3] \leftarrow$ Predict(M, Test_600)
            $T \leftarrow T + 1$
        end for
        $N \leftarrow N + 1$
    end for
    $Top\_1 \leftarrow C1$
    $Top\_2 \leftarrow C1 \ \& \ C2$
    $Top\_3 \leftarrow C1 \ \& \ C2 \ \& \ C3$
    return Top_1, Top_2, Top_3
end procedure

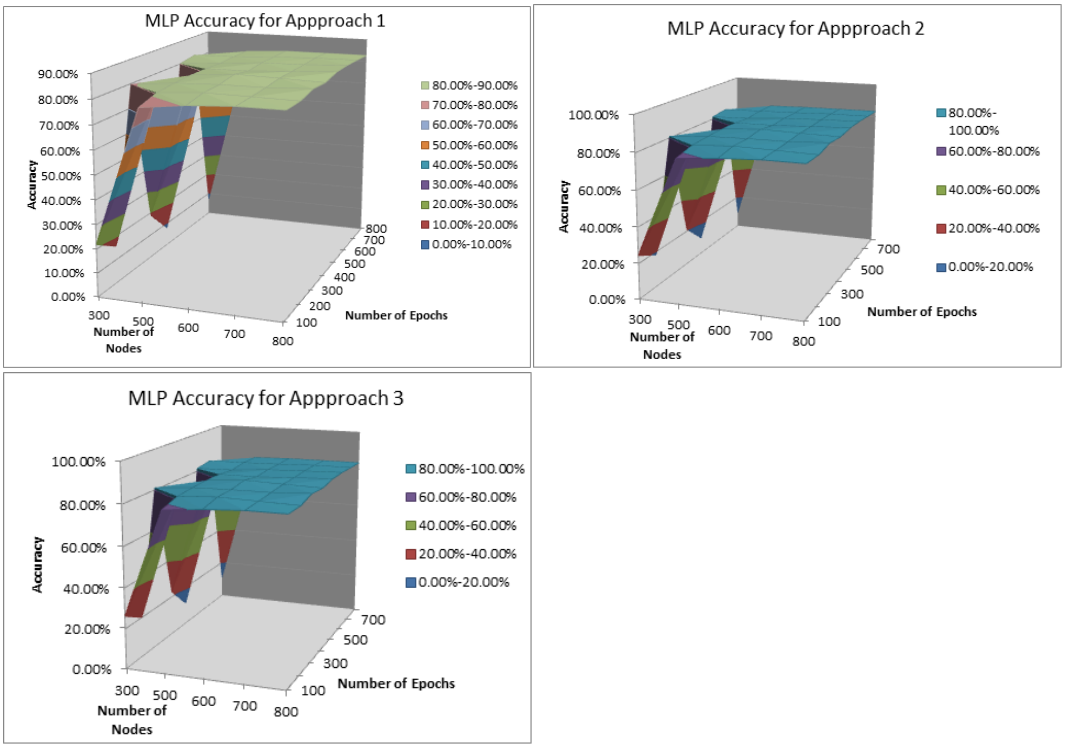

Figure 5. Accuracy of MLP for the three approaches: Top_1, Top_2, and Top_3.

In order to evaluate the different approaches, we randomly split the data from the *product matrix* obtained from the dataset into a training set ($\simeq$ 90% of the data) and a testing set ($\simeq$ 10% of the data). The evaluation performed on some of the measure parameters, such as accuracy, precision, recall, and $F1$-score [53] as shown in Equation (9). These parameters are obtained from (i) True Positive (TP) (when the real value is positive and prediction is positive); (ii) True Negative (TN) (when the real value is negative and the prediction is negative); (iii) False Positive (FP) (when the real values is negative but the prediction is positive); and (iv) False Negative (FN) (when the real value is positive but the prediction is negative). Therefore, accuracy is described as the ratio of correctly classified instances, which is the total number of correct predictions divided by the total number of

predictions made for a dataset; meanwhile, precision and recall are used to quantify how well the proposed algorithm matches the ground truth. Precision quantifies the number of positive class predictions that actually belong to the positive class. In addition, Recall quantifies the number of positive class predictions made out of all positive examples in the dataset. Finally, the F1-score provides a way to combine both precision and recall into a single measure that captures both properties:

$$\begin{aligned} Accuracy &= \frac{TP + TN}{TP + TN + FP + FN} \\ Precision &= \frac{TP}{TP + FP} \\ Recall &= \frac{TP}{TP + FN} \\ F1 - Score &= 2 * \frac{Precision * Recall}{Precision + Recall} \end{aligned} \tag{9}$$

Table 8 and Figure 6 summarize the performance of our score-based approach, based on the BM25 model.

Table 8. Score-based Methods: Quality measures.

	Top_1	*Top_2*	*Top_3*
Accuracy	0.833	0.920	0.948
Precision	0.829	0.914	0.937
Recall	0.824	0.905	0.932
F1-score	0.818	0.905	0.931

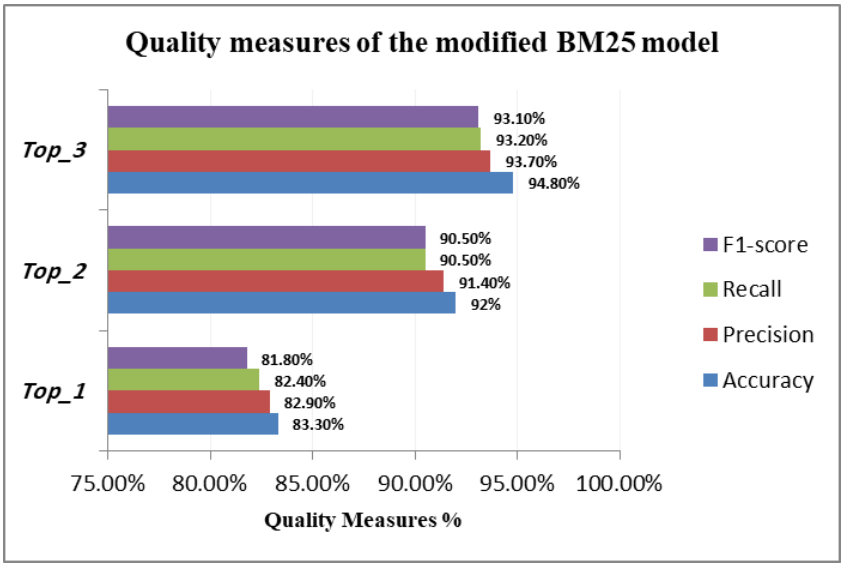

Figure 6. Score-based Methods: Quality measures.

We evaluate the performance of our nearest neighbors models over a reduced dimension dataset with 400, 500, 600, 700, and 800 principal components. KNN results are shown in Table 9. The results represent the best values for *Top*_1: 700 principal components with Accuracy (0.8427). The best fit with *Top*_2 is 600 principal components with Accuracy (0.8874). However, 500 principal components are also suitable for the F1-score (0.8241). In addition to the *Top*_3, the exact result is 400 principal components with Accuracy (0.9201). On the other hand, 700 principal components are also suitable for the F1-score (0.8263). Regarding FKNN, Table 10 illustrates that the best fit with *Top*_1 for FKNN is 800 principal components with accuracy (0.8459). In addition, the best fit with *Top*_2 is 600 principal

components with accuracy (0.9153), also suitable for the F1-score (0.8302). In addition to the *Top*_3, the best result is 600 principal components with accuracy (0.9413) and F1-score (0.8302). We would like to draw attention to the values of *Top*_1 results in Tables 9 and 10. Both of them are the same, which indicates that recall equals precision and accuracy. This means that the model is somehow "balanced", that is, its ability to correctly classify positive samples (*TP*+*FN*) is the same as its ability to correctly classify negative samples (*TN*+*FP*). Therefore, the F1-score is also the same [53].

Table 9. KNN: Quality measures by number of PCA components.

PCA	400	500	600	700	800
			***Top*_1**		
Accuracy	0.8339	0.8348	0.8367	0.8427	0.8332
Precision	0.8339	0.8348	0.8367	0.8427	0.8332
Recall	0.8339	0.8348	0.8367	0.8427	0.8332
F1-score	0.8339	0.8348	0.8367	0.8427	0.8332
			***Top*_2**		
Accuracy	0.8862	0.8871	0.8874	0.8871	0.8865
Precision	0.8405	0.8441	0.8403	0.8408	0.8346
Recall	0.8019	0.805	0.8072	0.8072	0.8063
F1-score	0.8207	0.8241	0.8234	0.8237	0.8202
			***Top*_3**		
Accuracy	0.9201	0.9125	0.9141	0.915	0.9137
Precision	0.8302	0.8313	0.8282	0.8048	0.8235
Recall	0.78	0.7793	0.7873	0.7844	0.7841
F1-score	0.8043	0.8045	0.8072	0.8263	0.8033

Table 10. FKNN: Quality measures by PCA components.

PCA	400	500	600	700	800
			***Top*_1**		
Accuracy	0.8396	0.8424	0.8446	0.8443	0.8459
Precision	0.8396	0.8424	0.8446	0.8443	0.8459
Recall	0.8396	0.8424	0.8446	0.8443	0.8459
F1-score	0.8396	0.8424	0.8446	0.8443	0.8459
			***Top*_2**		
Accuracy	0.8957	0.9141	0.9153	0.9061	0.9096
Precision	0.7908	0.8289	0.8317	0.8165	0.8216
Recall	0.7885	0.8266	0.8288	0.8126	0.8177
F1-score	0.7897	0.8278	0.8302	0.8146	0.8196
			***Top*_3**		
Accuracy	0.9315	0.94	0.9413	0.9362	0.9362
Precision	0.7949	0.8291	0.8319	0.8129	0.8167
Recall	0.7803	0.8231	0.8285	0.8088	0.8123
F1-score	0.7875	0.8261	0.8302	0.8109	0.8145

Table 11 presents the performance of our XGBoost classifier, which was also obtained for the principal components (400, 500, 600, 700, and 800), which declares selecting 400 as the best value for the principal component for *Top*_1, *Top*_2 and *Top*_3; accuracy 0.7071 for *Top*_1; 0.8126 for *Top*_2 and 0.8568 for *Top*_3.

Table 11. XGBoost: Quality measures by PCA components.

PCA	400	500	600	700	800
			Top_1		
Accuracy	0.7071	0.6909	0.6928	0.6885	0.6923
Precision	0.5908	0.5651	0.5664	0.5395	0.5648
Recall	0.5722	0.5467	0.5471	0.5176	0.5411
F1-score	0.5814	0.5557	0.5577	0.5283	0.5526
			Top_2		
Accuracy	0.8126	0.8095	0.7949	0.7994	0.7946
Precision	0.6850	0.6756	0.6445	0.6179	0.6544
Recall	0.6619	0.6556	0.6348	0.6058	0.6424
F1-score	0.6733	0.6655	0.6396	0.6117	0.6483
			Top_3		
Accuracy	0.8568	0.8534	0.8404	0.8489	0.8389
Precision	0.7243	0.7059	0.6878	0.6391	0.6647
Recall	0.7114	0.6987	0.6723	0.6272	0.6597
F1-score	0.7178	0.7023	0.6799	0.6331	0.6622

The performance of our MLP classifier was obtained also for 400, 500, 600, 700, and 800 principal components, as it is shown in Table 12, which suggests selecting 800 as the best value for the principal component for *Top*_1, *Top*_2, and *Top*_3; accuracy 0.8388 for *Top*_1; 0.8436 for *Top*_2 and 0.8436 for *Top*_3.

Table 12. MLP: Quality measures by PCA components.

PCA	400	500	600	700	800
			Top_1		
Accuracy	0.8327	0.8218	0.8366	0.8346	0.8388
Precision	0.8332	0.8220	0.8362	0.8343	0.8391
Recall	0.8324	0.8218	0.8360	0.8340	0.8389
F1-score	0.8328	0.8219	0.8361	0.8342	0.8390
			Top_2		
Accuracy	0.8407	0.8324	0.8417	0.8404	0.8436
Precision	0.9688	0.9717	0.9814	0.9803	0.9815
Recall	0.8324	0.8218	0.8360	0.8340	0.8389
F1-score	0.8954	0.8905	0.9029	0.9013	0.9046
			Top_3		
Accuracy	0.8411	0.8327	0.8420	0.8407	0.8436
Precision	0.9908	0.9907	0.9988	0.9950	0.9965
Recall	0.8324	0.8218	0.8360	0.8340	0.8389
F1-score	0.9047	0.8983	0.9102	0.9074	0.9109

Additionally, we have assessed the results for the accuracy and the F1-score for the last four algorithms with respect to *Top*_1, *Top*_2 and *Top*_3 on a reduced dimension dataset with 400, 500, 600, 700, and 800 principal components, as shown in Figure 7. The FKNN obtains the best accuracy on *Top*_1, *Top*_2, and *Top*_3. The highest accuracy, in *Top*_1, with the appropriate PCA, is PCA-800 is (84.59%), and, in *Top*_2 and *Top*_3, with the suitable PCA, it is PCA-600 is (91.53%, 94.13%), respectively. The best algorithm to obtain the highest result in the F1-score is FKNN in *Top*_1 with the appropriate principal component which is PCA-800 at the rate of 84.59%, while, in *Top*_2 and *Top*_3, the best algorithm is MLP with PCA-800 at the rate of 90.46%, 91.09%, respectively.

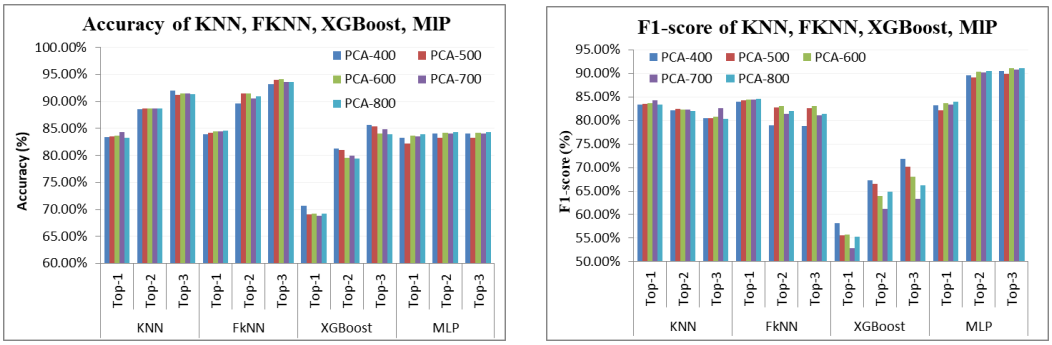

Figure 7. Accuracy & F1-score of KNN, FKNN, XGBoost, MLP.

To sum up, Figure 8 shows the comparison in terms of accuracy among all the models (for their optimal values) regarding *Top*_1, *Top*_2 and *Top*_3. The best in *Top*_1 is FKNN followed by KNN. In addition, the appropriate one in *Top*_2, *Top*_3 is BM25 followed by FKNN.

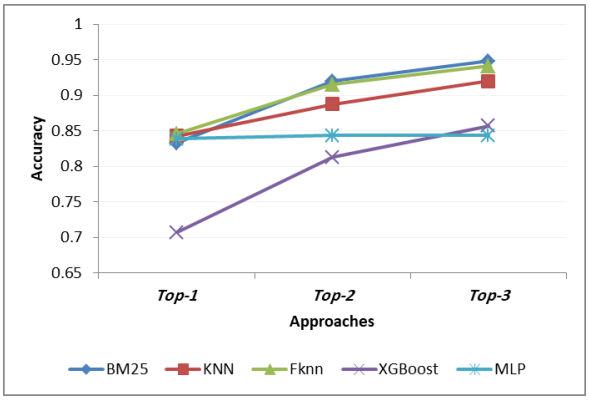

Figure 8. Comparative study between all algorithms.

10. Conclusions

In this paper, we propose a solution for automatic classification of groceries in a digital transformation scenario, where the availability of quality data are key for retailers and costumers. In addition, the food market is a very dynamic context where new products emerge daily, so product catalogs are composed of a huge amount of data that need constant updates. Within this context, the Spanish company *Midiadia* provides retailers with quality data about their product catalogs. *Midiadia* processes the information from the products packaging and labeling to offer relevant knowledge to retailers that eventually support commercial processes (offers, customization, etc.). Our main objective is to collaborate with *Midiadia* to improve the efficiency of one of their internal processes: classification of new products into the ontology the company uses. In particular, we focused on providing an automatic classification mechanism that replaces the manual task of deciding to which *Variety* (one of the taxonomy levels that is composed of 159 values) a new product belongs.

With this aim, we worked with three different alternatives in order to decide the most appropriate: score-based algorithms, machine learning approaches, and deep neural networks. After intensive work to preprocess the dataset (MDD-DS, composed by 20,888 products), we applied the following algorithms: first, we define a score-based algorithm based on the probabilistic model BM25, but adapting the formulation to our specific problem of groceries. Then, we applied the KNN, FKNN, and XGBoost algorithms, after reducing the data dimensionality with PCA. Finally, we also applied an MLP algorithm using the same PCA mechanism for reducing data dimensionality.

The main objective is providing a totally automatic classification tool that directly offers one result: the most appropriate *Variety* for a new product (*Top*_1 option). However, and considering the company benefits, we also studied other two alternatives: (i) *Top*_2

that obtains the two most appropriate *Varieties* for a new product and (ii) Top_3 that obtains the three most appropriate *Varieties* for a new product. Both of them would also help the classification process by turning the problem of classifying a new product within 159 categories into a problem of classifying a new product within two or three categories. According to our results, the best in Top_1 is FKNN, closely followed by KNN. However, in Top_2 and Top_3, the score-based algorithm performs better, closely followed by FKNN.

We are currently working on the creation of a dictionary of synonyms for ingredients. This need arises because the current regulations allow manufacturers to include different terms for the same element (vitamin C or ascorbic acid, for instance). Additionally, we are working on extending our analysis to other approaches, such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Deep Bidirectional LSTMs (BILSTM), Bidirectional Encoder Representations from Transformers (BERT), and Support Vector Machine (SVM). Regarding the course of dimensionality, applying DNN with abstract features is a natural next step provided that the issues related with memory consumption during training can be mitigated. After that, we expect to perform a comparison experiment with deep neural networks using additional hidden layers. In addition, the research work already done for the classification problem is the basis of a recommender algorithm to provide product alternatives to the customer when the desired product is not available. With this aim, we are currently working on a multidimensional recommender that takes into account the list of ingredients and other relevant data, such as calories, allergens, healthy aspects, etc.

Author Contributions: Conceptualization, M.M.H., A.F.V., R.P.D.R. and H.O.P.; methodology, M.M.H., A.F.V., R.P.D.R. and H.O.P.; software, M.M.H. and H.O.P.; validation, M.M.H. and H.O.P.; formal analysis, M.M.H., A.F.V., R.P.D.R. and H.O.P.; investigation, M.M.H., A.F.V., R.P.D.R. and H.O.P.; resources, A.F.V. and R.P.D.R.; data curation, M.M.H.; writing—original draft preparation, M.M.H., A.F.V. and R.P.D.R.; writing—review and editing, M.M.H., A.F.V. and R.P.D.R.; supervision, A.F.V. and R.P.D.R.; funding acquisition, A.F.V. and R.P.D.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work has received financial support from the European Regional Development Fund (ERDF) and the Galician Regional Government, under the agreement for funding the Atlantic Research Center for Information and Communication Technologies (atlanTTIC), and the Spanish Ministry of Economy and Competitiveness, under the National Science Program (TEC2017-84197-C4-2-R).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors would like to thank the European Regional Development Fund (ERDF) and the Galician Regional 7 Government, under the agreement for funding the Atlantic Research Center for Information and Communication Technologies (atlanTTIC), and the Spanish Ministry of Economy and Competitiveness, under the National Science Program (TEC2017-84197-C4-2-R).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Extra Tables

Table A1. Definition of distance measures [31,54].

Cityblock, also known as Manhattan distance, It examines the absolute differences between the opposite values in vectors as shown in Equation (A1).

$$CB[a,b] = \sum_{i=1}^{n} |a_i - b_i| \quad \text{(A1)}$$

Cosine is a measure of similarity between two nonzero vectors of an internal product space that measures the cosine of the angle between them as shown in Equation (A2).

$$cos(a,b) = \frac{a \cdot b}{||a|| \cdot ||b||} \quad \text{(A2)}$$

Correlation is a measure of dependency between two paired random vectors of arbitrary dimension, not necessarily equal as shown in Equation (A3).

$$CorD[a,b] = \frac{\sum_{i=1}^{n}(a_i - \overline{a})(b_i - \overline{b})}{\sqrt{\sum_{i=1}^{n}(a_i - \overline{a})^2 (b_i - \overline{b})^2}} \quad \text{(A3)}$$

Euclidean is the ordinary straight-line distance between two points in Euclidean space as shown in Equation (A4).

$$ED[a,b] = \sqrt{\sum_{i=1}^{n} |a_i - b_i|^2} \quad \text{(A4)}$$

Seuclidean, Standardized Euclidean distance. Defined as the Euclidean distance calculated on standardized data as shown in Equation (A5).

$$SED[a,b] = \sqrt{\sum_{i=1}^{n} \frac{1}{k_i^2}(a_i - b_i)^2} \quad \text{(A5)}$$

Jaccard distance measures the difference between sets of samples, it is a complement to the Jaccard similarity coefficient and is obtained by subtracting the Jaccard coefficient from one as shown in Equation (A6):

$$Jac[a,b] = \frac{\sum_i \min(a_i, b_i)}{\sum_i \max(a_i, b_i)} \quad \text{(A6)}$$

Hamming, the percentage of coordinates that differ as shown in Equation (A7).

$$HD(a,b) = \sum_{i=1}^{n} 1_{a_i \neq b_i} \quad \text{(A7)}$$

Chebychev, Maximum coordinate difference as shown in Equation (A8).

$$CD[a,b] = \max_i |a_i - b_i| \quad \text{(A8)}$$

Spearman's ρ is a nonparametric measure of the statistical dependence of rank correlation between the rankings of two variables. Evaluate how well the relationship between two variables can be described using a monotonic function as shown in Equation (A9).

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \quad \text{(A9)}$$

where d = the pairwise distances of the ranks of the variables a_i *and* b_i. n = the number of samples.

References

1. Reinartz, W.; Wiegand, N.; Imschloss, M. The impact of digital transformation on the retailing value chain. *Int. J. Res. Mark.* **2019**, *36*, 350–366. [CrossRef]
2. Wessel, L.; Baiyere, A.; Ologeanu-Taddei, R.; Cha, J.; Jensen, T. Unpacking the difference between digital transformation and IT-enabled organizational transformation. *J. Assoc. Inf. Syst.* **2020**, doi: 10.17705/1jais.00655. [CrossRef]
3. The Digitally Engaged Food Shopper: Developing Your Omnichannel Collaboration Model. 2018. Available online: https://www.fmi.org/forms/store/ProductFormPublic/the-digitally-engaged-food-shopper-developing-your-omnichannel-collaboration-model (accessed on 20 May 2019)
4. Bahn, R.A.; Abebe, G.K. A Descriptive Analysis of Food Retailing in Lebanon: Evidence from a Cross-Sectional Survey of Food Retailers. In *Food Supply Chains in Cities*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 289–346.
5. Hafez, M.M.; Redondo, R.P.D.; Vilas, A.F. A Comparative Performance Study of Naïve and Ensemble Algorithms for E-commerce. In Proceedings of the 2018 14th International Computer Engineering Conference (ICENCO), Cairo, Egypt, 29–30 December 2018; pp. 26–31.
6. European Commission. General Food Law. 2002. Available online: https://ec.europa.eu/food/safety/general_food_law_en (accessed on 4 June 2019).
7. BOE. Real Decreto Legislativo 1/2007, de 16 de Noviembre, por el que se Aprueba el Texto Refundido de la Ley General Para la Defensa de los Consumidores y Usuarios y Otras Leyes Complementarias. Available online: https://www.boe.es/eli/es/rdlg/2007/11/16/1/con (accessed on 15 June 2019).

8. BOE. Ley 2/2012, de 28 de Marzo, Gallega de Protección General de las Personas Consumidoras y Usuarias. Available online: https://www.boe.es/eli/es-ga/l/2012/03/28/2 (accessed on 23 June 2019).
9. Baz, I.; Yoruk, E.; Cetin, M. Context-aware hybrid classification system for fine-grained retail product recognition. In Proceedings of the 2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP), Bordeaux, France, 11–12 July 2016; pp. 1–5.
10. Fuchs, K.; Grundmann, T.; Fleisch, E. Towards identification of packaged products via computer vision: Convolutional neural networks for object detection and image classification in retail environments. In Proceedings of the 9th International Conference on the Internet of Things, Bilbao, Spain, 22–25 October 2019; pp. 1–8.
11. Hafez, M.M.; Shehab, M.E.; El Fakharany, E.; Hegazy, A.E.F.A.G. Effective selection of machine learning algorithms for big data analytics using apache spark. In Proceedings of the International Conference on Advanced Intelligent Systems and Informatics, Cairo, Egypt, 24 October 2016; pp. 692–704.
12. Peng, J.; Xiao, C.; Wei, X.; Li, Y. RP2K: A Large-Scale Retail Product Dataset forFine-Grained Image Classification. *arXiv* **2020**, arXiv:2006.12634.
13. Baz, İ. Statistical Methods for Fine-Grained Retail Product Recognition. Ph.D. Thesis, Sabanci University, Tuzla/Istanbul, Turkey, 19 July 2019.
14. Bianchi-Aguiar, T.; Hübner, A.; Carravilla, M.A.; Oliveira, J.F. Retail shelf space planning problems: A comprehensive review and classification framework. *Eur. J. Oper. Res.* **2020**, *289*, 1–16. [CrossRef]
15. Goldman, E.; Herzig, R.; Eisenschtat, A.; Goldberger, J.; Hassner, T. Precise detection in densely packed scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 5227–5236.
16. Gundimeda, V.; Murali, R.S.; Joseph, R.; Babu, N.N. An automated computer vision system for extraction of retail food product metadata. In *First International Conference on Artificial Intelligence and Cognitive Computing*; Springer: Singapore, 2019; pp. 199–216.
17. Wang, X.; Sun, Z.; Zhang, W.; Zhou, Y.; Jiang, Y.G. Matching user photos to online products with robust deep features. In Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, New York, NY, USA, 6–9 June 2016; pp. 7–14.
18. Zhong, C.; Jiang, L.; Liang, Y.; Sun, H.; Ma, C. Temporal Multiple-convolutional Network for Commodity Classification of Online Retail Platform Data. In Proceedings of the 2020 12th International Conference on Machine Learning and Computing, Shenzhen, China, 19–21 June 2020; pp. 236–241.
19. Pobbathi, N.R.; Dong, A.; Chang, Y. Automated Categorization of Products in a Merchant Catalog. U.S. Patent 10,528,907, 7 January 2020.
20. Seth, S.; Johnson, B.S.; Kennedy, R.; Kothari, N. Method and System to Categorize Items Automatically. U.S. Patent 10,706,076, 7 July 2020.
21. Gupta, V.; Karnick, H.; Bansal, A.; Jhala, P. Product Classification in E-Commerce using Distributional Semantics. *arXiv* **2016**, arXiv:1606.06083.
22. Baeza-Yates, R.; Ribeiro-Neto, B. *Modern Information Retrieval: The Concepts and Technology behind Search*, 2nd ed.; Addison-Wesley Publishing Company: Boston, MA, USA, 2011.
23. Chen, H.; Pang, J.; Koo, M.; Patrick, V.M. Shape Matters: Package Shape Informs Brand Status Categorization and Brand Choice. *J. Retail.* **2020**, *96*, 266–281. [CrossRef]
24. Wei, Y.; Tran, S.; Xu, S.; Kang, B.; Springer, M. Deep learning for retail product recognition: Challenges and techniques. *Comput. Intell. Neurosci.* **2020**, *2020*, 8875910. [CrossRef] [PubMed]
25. Wei, X.S.; Wu, J.; Cui, Q. Deep learning for fine-grained image analysis: A survey. *arXiv* **2019**, arXiv:1907.03069.
26. Dashtipour, K.; Gogate, M.; Li, J.; Jiang, F.; Kong, B.; Hussain, A. A hybrid Persian sentiment analysis framework: Integrating dependency grammar based rules and deep neural networks. *Neurocomputing* **2020**, *380*, 1–10. [CrossRef]
27. Korgaonkar, P.; Silverblatt, R.; Girard, T. Online retailing, product classifications, and consumer preferences. *Internet Res.* **2006**, *16*, 267–288. [CrossRef]
28. Ravnik, R.; Solina, F.; Zabkar, V. Modelling in-store consumer behavior using machine learning and digital signage audience measurement data. In *International Workshop on Video Analytics for Audience Measurement in Retail and Digital Signage*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 691–695.
29. Holỳ, V.; Sokol, O.; Černỳ, M. Clustering retail products based on customer behavior. *Appl. Soft Comput.* **2017**, *60*, 752–762. [CrossRef]
30. Robertson, S.; Walker, S.; Beaulieu, M. Experimentation as a way of life: Okapi at TREC. *Inf. Process. Manag.* **2000**, *36*, 95–108. [CrossRef]
31. Abu Alfeilat, H.A.; Hassanat, A.B.; Lasassmeh, O.; Tarawneh, A.S.; Alhasanat, M.B.; Eyal Salman, H.S.; Prasath, V.S. Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review. *Big Data* **2019**, *7*, 221–248. [CrossRef]
32. Keller, J.M.; Gray, M.R.; Givens, J.A. A fuzzy k-nearest neighbor algorithm. *IEEE Trans. Syst. Man Cybern.* **1985**, *SMC-15*, 580–585.
33. Derrac, J.; García, S.; Herrera, F. Fuzzy nearest neighbor algorithms: Taxonomy, experimental analysis and prospects. *Inf. Sci.* **2014**, *260*, 98–119. [CrossRef]
34. Wankhede, S.B. Analytical study of neural network techniques: SOM, MLP and classifier-a survey. *IOSR J. Comput. Eng.* **2014**, *16*, 86–92. [CrossRef]
35. Al-Shammari, E.T. Lemmatizing, Stemming, and Query Expansion Method and System. U.S. Patent 8,473,279, 25 June 2013.

36. Solorio-Fernández, S.; Carrasco-Ochoa, J.A.; Martínez-Trinidad, J.F. A review of unsupervised feature selection methods. *Artif. Intell. Rev.* **2020**, *53*, 907–948.
37. StiIhle, L.; Wold, S. Analysis of variance (ANOVA). *Chemom. Intell. Lab. Syst.* **1989**, *6*, 259–272.
38. Cai, J.; Luo, J.; Wang, S.; Yang, S. Feature selection in machine learning: A new perspective. *Neurocomputing* **2018**, *300*, 70–79.
39. Meera, S.; Sundar, C. A hybrid metaheuristic approach for efficient feature selection methods in big data. *J. Ambient. Intell. Hum. Comput.* **2020**, *12*, 3743–3751.
40. Guyon, I.; Gunn, S.; Nikravesh, M.; Zadeh, L.A. *Feature Extraction: Foundations and Applications*; Springer: Berlin, Germany, 2008; Volume 207.
41. Wang, D.; Xu, J. Principal component analysis in the local differential privacy model. *Theor. Comput. Sci.* **2020**, *809*, 296–312. [CrossRef]
42. Balakrishnama, S.; Ganapathiraju, A. Linear discriminant analysis-a brief tutorial. *Inst. Signal Inf. Process.* **1998**, *18*, 1–8.
43. Kunang, Y.N.; Nurmaini, S.; Stiawan, D.; Zarkasi, A. Automatic features extraction using autoencoder in intrusion detection system. In Proceedings of the 2018 International Conference on Electrical Engineering and Computer Science (ICECOS), Pangkal, Indonesia, 2–4 October 2018; pp. 219–224.
44. Wang, Y.; Yao, H.; Zhao, S. Auto-encoder based dimensionality reduction. *Neurocomputing* **2016**, *184*, 232–242. [CrossRef]
45. Jung, Y.M. Principal component analysis based two-dimensional (PCA-2D) correlation spectroscopy: PCA denoising for 2D correlation spectroscopy. *Bull. Korean Chem. Soc.* **2003**, *24*, 1345–1350.
46. Reddy, G.T.; Reddy, M.P.K.; Lakshmanna, K.; Kaluri, R.; Rajput, D.S.; Srivastava, G.; Baker, T. Analysis of dimensionality reduction techniques on big data. *IEEE Access* **2020**, *8*, 54776–54788. [CrossRef]
47. Beaulieu, M.M.; Gatford, M.; Huang, X.; Robertson, S.; Walker, S.; Williams, P. Okapi at TREC-5. *Nist Spec. Publ. SP* **1997**, *143–166*, 500238.
48. Rastin, N.; Jahromi, M.Z.; Taheri, M. A Generalized Weighted Distance k-Nearest Neighbor for Multi-label Problems. *Pattern Recognit.* **2020**, *114*, 107526.
49. Hassanat, A.B.; Abbadi, M.A.; Altarawneh, G.A.; Alhasanat, A.A. Solving the problem of the K parameter in the KNN classifier using an ensemble learning approach. *arXiv* **2014**, arXiv:1409.0919.
50. Thongsuwan, S.; Jaiyen, S.; Padcharoen, A.; Agarwal, P. ConvXGB: A new deep learning model for classification problems based on CNN and XGBoost. *Nucl. Eng. Technol.* **2021**, *53*, 522–531. [CrossRef]
51. Shilong, Z. Machine Learning Model for Sales Forecasting by Using XGBoost. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE), Guangzhou, China, 15–17 January 2021; pp. 480–483.
52. Feng, X.; Ma, G.; Su, S.F.; Huang, C.; Boswell, M.K.; Xue, P. A multi-layer perceptron approach for accelerated wave forecasting in Lake Michigan. *Ocean Eng.* **2020**, *211*, 107526. [CrossRef]
53. Brownlee, J. How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification. Available online: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/ (accessed on 11 March 2020).
54. Banda, J. Framework for Creating Large-Scale Content-Based Image Retrieval System (CBIR) for Solar Data Analysis. Ph.D Thesis, Montana State University-Bozeman, Bozeman, Montana, 2011.