Article

Vocational Domain Identification with Machine Learning and Natural Language Processing on Wikipedia Text: Error Analysis and Class Balancing

Maria Nefeli Nikiforos*, Konstantina Deliveri, Katia Lida Kermanidis and Adamantia Pateli

Department of Informatics, Ionian University, Corfu, 49132, Greece; p15deli@ionio.gr (K.D.); kerman@ionio.gr (K.L.K.); pateli@ionio.gr (A.P.)

* Correspondence: nefeli.nikiforos@ionio.gr (M.N.N)

Abstract: Highly-skilled migrants and refugees finding employment in low-skill vocations, despite professional qualifications and educational background, has become a global tendency, mainly due to the language barrier. Employment prospects for displaced communities are mostly decided by their knowledge of the sublanguage of the vocational domain they are interested in working. Common vocational domains include agriculture, cooking, crafting, construction, and hospitality. The increasing amount of user-generated content in wikis and social networks provides a valuable source of data for data mining, Natural Language Processing and machine learning applications. This paper extends the contribution of the authors' previous research on automatic vocational domain identification by further analyzing the results of the machine learning experiments with the domain-specific textual data set, considering 2 research directions: a. predictions analysis and b. data balancing. Wrong predictions analysis and the features that contributed to misclassification, along with correct predictions analysis and the features that were the most dominant, contributed to the identification of a primary set of terms for the vocational domains. Data balancing techniques were applied on the data set to observe their impact on the performance of the classification model. A novel 4-step methodology is proposed in this paper for the first time, consisting of successive applications of SMOTE oversampling on imbalanced data. Data oversampling obtains better results than data undersampling in imbalanced data sets, while hybrid approaches perform reasonably well.

Keywords: Natural Language Processing; Social Text Mining; Machine Learning; Vocational Domain Identification; Vocational Language; Error Analysis; Class Balancing

1. Introduction

1.1. Vocational Domains for Migrants and Refugees

Migrant employees face multiple challenges deriving from discrimination due to country of origin, nationality, culture, sex, etc. [1–4], while for women in particular, finding employment in high-skill vocations, besides teaching and nursing, is observed to be especially difficult [1,2,5]. A deciding factor regarding the prospects of employment for displaced communities, like migrants and refugees, is the knowledge of not the language of their host country in general, but specifically of the sublanguage of the vocational domain they are interested in working. As a result, highly-skilled migrants and refugees finding employment in low-skill vocations, despite their professional qualifications and educational background, has become a global tendency, with the language barrier being one of the most important factors [1–4,6].

The scope of vocational domains for displaced communities and analyses on their situation in the host country and in their country of origin were examined in recent literature, considering the impact on their work-life balance [2–4]. Both high-skill and low-skill vocations in hospitality, cleaning, manufacturing, retail, crafting, and agriculture consist the most common vocational domains in which migrants and refugees seek and

 doi:10.20944/preprints202304.0907.v1

find employment, according to the findings of the recent research [1–3,6–8]. It is also important to note that unemployment usually affects more the displaced communities than the natives [9]. Overworking, however, due to low-paid jobs, is thriving, as migrants and refugees struggle to increase their earnings, in their efforts to maintain living standards, afford childcare and be able to send remittances to remaining family in their country of origin [2,7].

1.2. Wikipedia and Social Networks

With the expansion of the user base of wikis and social networks in the last decade, user-generated content has increased in great amounts. This content provides a valuable source of data for various tasks and applications in data mining, Natural Language Processing (NLP) and machine learning. Wikipedia[1] is an open data wiki covering a wide scope of topic-related articles written in many languages [10]. Wikipedia's content generation is a constant collective process derived from the collaboration of its users [11]; as of April 2023, there are approximately 6.6 million Wikipedia articles written in English.

1.3. Class Imbalance Problem

Imbalanced data sets, in regards of class distribution among their examples, present several challenges in data mining and machine learning. More specifically, the number of examples representing the class of interest is considerably smaller than the ones of the other classes. As a result, standard classification algorithms have a bias towards the majority class, and consequently they tend to misclassify the minority class examples [12]. Most commonly, the class imbalance problem is related to binary classification, though it is not uncommon to present in multi-class problems (like in this paper); since there are more than one minority class, it is more challenging to solve. The class imbalance problem is an issue that affects various aspects of real-world applications that are based on classification due to the fact that the minority class examples are the most difficult to obtain from real data, especially from user-generated content from wikis and social networks, leading a large community of researchers to examine ways to address it [12–18].

1.4. Contribution

This paper extends the contribution of the authors' previous research [19] by exploring the various potential directions deriving from it. The results of the machine learning experiments with the domain-specific textual data set created and preprocessed as described in [19] were further processed and analyzed, considering 2 research directions: a. predictions analysis and b. data balancing.

More specifically, regarding the predictions analysis, important conclusions were drawn from examining which examples were classified wrongly for each class (wrong predictions) by the *Gradient Boosted Trees* model which managed to classify correctly most of the examples, as well as which distinct features contributed to their misclassification. In the same line of thought, regarding the correctly classified examples (correct predictions), the examination of the features that were the most dominant leading to the correct classifications for each class contributed to the identification of a primary set of terms highlighting the terminology of the vocational domains.

Regarding the data balancing direction, oversampling and undersampling techniques, both separately and in combination as a hybrid approach, were applied on the data set, in order to observe their impact (positive or negative) on the performance of the *Random Forest* and *AdaBoost* model. A novel methodology is proposed in this paper for the first time, consisting of successive applications of SMOTE oversampling on imbalanced data, balancing the data considering which is the minority class each time, in 4 steps. By running the experiments following this methodology, the impact of every class distribution, from completely imbalanced to completely balanced, on the performance of the machine learning

[1] https://en.wikipedia.org/wiki/Main_Page

model can be examined thoroughly. The process of the class balancing direction enabled the comparison of the performance of this model with balanced data to the performance of the same model with imbalanced data from the previous research [19]. The findings derived from the machine learning experiments of this paper are in accordance with those of the relevant literature [12,17], in terms that data oversampling obtains better results than data undersampling in imbalanced data sets, while hybrid approaches perform reasonably well.

1.5. Structure

The structure of this paper is the following. Section 2 presents past related work on: a. domain identification on textual data, b. data scraping from social text, and c. data oversampling, undersampling and hybrid approaches. Section 3 describes the stages of data set creation and preprocessing, and the feature extraction process. Section 4 presents the research direction of predictions analysis, including both wrong and correct predictions of the *Gradient Boosted Trees* model. Section 5 presents the research direction of data balancing, including the novel methodology for successive SMOTE oversampling, as well as experiments with data undersampling and a hybrid approach. Section 6 concludes the paper, discusses the most important findings and draws directions for future work.

2. Related Work

In this Section, recent literature on domain identification on textual data, including news articles, technical text, open data and Wikipedia articles is presented. Research on data scraping from social text, including social networks and Wikipedia is also described. Finally, findings of related work regarding data oversampling, undersampling and hybrid approaches are also analyzed.

2.1. Domain Identification on Textual Data

Domain identification performed on textual data, including news articles, social media posts, and social text data sets in general, remains an open problem and a very challenging task for researchers. The vast domain diversity along with their particular sublanguage and terminology present several challenges when undertaking domain identification on textual and linguistic data.

2.1.1. News Articles

Regarding domain identification on news articles, Hamza et al. [20] built a data set containing news articles written in Urdu and annotated with 7 domains as classes according to their topic. Their feature set consisted of unigrams, bigrams and Term Frequency - Inverse Document Frequency (TF-IDF). Following the stages of preprocessing, namely stopwords removal and stemming, they performed text classification to the 7 domains by employing 6 machine learning models; Multi-Layered Perceptron (MLP) reached the highest accuracy of 91.4%. Their findings showed that stemming did not affect positively the performance of the models, however stopwords removal had worsened it. Another paper by Balouchzahi et al. [21] attempted domain identification on fake news articles written in English and annotated with 6 domains according to their topic. Their ensemble of RoBERTa, DistilBERT and BERT managed up to 85.5% F1 score.

2.1.2. Technical Text

There are certain researchers who performed domain identification on technical text. Hande et al. [22] classified scientific articles in 7 computer science domains by using transfer learning with BERT, RoBERTa and SciBERT. They found that the ensemble reaches its best performance when the weights are taken into account. In the research of Dowlagar & Mamidi [23], experiments with BERT and XLM-ROBERTa with a Convolutional Neural Network (CNN) on a multilingual technical data set obtained better results in comparison with experiments with Support Vector Machines (SVM) with TF-IDF and CNN. By selecting

 doi:10.20944/preprints202304.0907.v1

the textual data written in Telugu from same data set, Gundapu & Mamidi [24] obtained up to 69.9% F1 score with CNN and a Self-Attention based Bidirectional Long Short-Term Memory (BiLSTM) network.

2.1.3. Open Data & Wikipedia Articles

Regarding domain identification on open data, Lalithsena et al. [25] performed automatic topic identification by using Map-Reduce combined with manual validation by humans on several data sets from Linked Open Data. In order to designate distinct topics, they used specialized tags for the annotation.

In the paper of Nakatani et al. [26], Wikipedia structural feature and term extraction was performed, aiming to score both topic coverage and topic detailedness on web search results, relevant to the related search queries. Saxena et al. [27] built domain-specific conceptual bases using Wikipedia navigational templates. They employed a knowledge graph and then applied fuzzy logic on each article's network metrics. In the research of Stoica et al. [28], a Wikipedia article by topic extractor was created. Preprocessing included parsing the articles for lower-casing, stopwords removal and embedding generation. The extractor obtained high precision, recall and F1 score up to 90% with Random Forest, SVM and Extreme Gradient Boosting, along with cross-validation.

In the authors' previous research, Nikiforos et al. [19], automatic vocational domain identification was performed. A domain-specific textual data set from Wikipedia articles was created, along with a linguistic feature set with TF-IDF. Preprocessing included tokenization, removal of numbers, punctuation marks, stopwords and duplicates, and lemmatization. 5 vocational domains, in which displaced communities, like migrants and refugees, commonly seek and find employment, were considered as classes. Machine learning experiments were performed with Random Forest combined with AdaBoost, and Gradient Boosted Trees, with the latter obtaining the best performance with up to 99.93% accuracy and 100% F1-score.

In Table 1, the performance of the related work mentioned in this subsection is shown, in terms of evaluation metrics such as accuracy and F1 score, and considering the data sets and models that procured the best results for each research paper.

Table 1. Performance per research paper. Data sets and models of related work with the best results.

Paper	Classes	Model	Performance
Hamza et al. [20]	7 domains of Urdu news	MLP	Accuracy: 91.4%
Balouchzahi et al. [21]	6 domains of English fake news	Ensemble: RoBERTa, DistilBERT, BERT	F1 score: 85.5%
Hande et al. [22]	7 computer science domains of scientific articles	Ensemble: BERT, RoBERTa, SciBERT	Accuracy: 92%, F1 score: 98%
Dowlagar & Mamidi [23]	7 multilingual technical domains	BERT, XLM-ROBERTa, CNN	F1 score (macro): 80.3%
Gundapu & Mamidi [24]	6 Telugu technical domains	CNN, BiLSTM	F1 score: 69.9%
Stoica et al. [28]	3 topic domains of Wikipedia	BERT, Random Forest, Extreme Gradient Boosting	F1 score: 90%
Nikiforos et al. [19]	5 vocational domains of English Wikipedia	Gradient Boosted Trees	Accuracy: 99.9%, F1 score: 100%

2.2. Social Text Data Scraping

Data scraping and analysis of textual data from social networks and Wikipedia have been attempted in recent research. "Data analysis is the method of extracting solutions to the problems via interrogation and interpretation of data" [29]. Despite the development of

 doi:10.20944/preprints202304.0907.v1

numerous web scrapers and crawlers, social data scraping and analysis of high quality still present a challenging task.

2.2.1. Social Networks

Several web scrapers were developed with Python. Scrapy, by Thomas & Mathur [29], scraped textual data from Reddit[2] and stored them in CSV files. Another scraper, by Kumar & Zymbler [30], scraped the Twitter API[3] to download tweets regarding particular airlines, which then were used as input for sentiment analysis and machine learning experiments with SVM and CNN, reaching up to 92.3% accuracy.

2.2.2. Wikipedia

Other web crawlers, more focused on Wikipedia data, were built. iPopulator by Lange et al. [10] used Conditional Random Fields (CRF) and crawled Wikipedia to gather textual data from the first paragraphs of Wikipedia articles, and then used them to populate an infobox for each article. iPopulator reached up to 91% average extraction precision with 1,727 infobox attributes. Cleoria and a MapReduce parser were used by Hardik et al. [11] to download and process XML files, aiming to evaluate the link-ability factor of Wikipedia pages.

In the authors' previous research, Nikiforos et al. [19], a web crawler was developed by using the Python libraries BeautifulSoup4 and Requests. It scraped Wikipedia's API, downloading textual data from 57 articles written in English, considering their relevance to 5 vocational domains in which refugees and migrant commonly seek and find employment as a criterion. The aim was to extract linguistic information concerning these domains, and perform machine learning experiments for domain identification.

2.3. Data Oversampling & Undersampling

Data sampling, either oversampling or undersampling, is one of the proposed solutions to mitigate the class imbalance problem. Resampling techniques practically change the class distribution in imbalanced data sets by creating new examples for the minority class(es) (oversampling), or removing examples from the majority class (undersampling), or doing both (hybrid) [12,16].

Several researchers proposed data undersampling techniques. Lin et al. [13] proposed 2 undersampling strategies in which a clustering technique is applied during preprocessing; the number of clusters of the majority class were made equal to the number of data points of the minority class. In order to represent the majority class, cluster centers and nearest neighbors of the cluster centers were used by the 2 strategies, respectively. They performed experiments on 44 small-scale and 2 large-scale data sets, resulting in the second strategy approach, combined with a single multilayer perceptron and a C4.5 decision tree, performing better compared to 5 state-of-the-art approaches. Anand et al. [14] introduced an undersampling technique and evaluated it by performing experiments on 4 real biological imbalanced data sets. Their technique improved the model sensitivity compared to weighted SVMs and other models in the related work for the same data. Yen & Lee [15] proposed cluster-based undersampling approaches to define representative data as the training set, aiming to increase the classification accuracy for the minority class in imbalanced data sets. García & Herrera [16] presented evolutionary undersampling, a taxonomy of methods which considered the nature of the problem and then applied different fitness functions to achieve both class balance and high performance. Their experiments with numerous imbalanced data sets showed that evolutionary undersampling performs better than other state-of-the-art undersampling models when the imbalance is increased.

---

[2] https://www.reddit.com/
[3] https://developer.twitter.com/en

 doi:10.20944/preprints202304.0907.v1

Other researchers experimented with data oversampling and hybrid approaches. Shelke et al. [18] examined class imbalance on text classification tasks with multiple classes, addressing sparsity and high dimensionality of textual data. After applying a combination of undersampling and oversampling techniques on the data, they performed experiments with multinomial Naïve Bayes, k-Nearest Neighbor, and SVMs. They concluded that the effectiveness of resampling techniques is highly data dependent, while certain resampling techniques achieve better performance when combined with specific classifiers. Lopez et al. [12] provided an extensive overview of class imbalance mitigating methodologies, namely data sampling, algorithmic modification and cost-sensitive learning. They discussed the most significant challenges of using data intrinsic characteristics in classification problems with imbalanced data sets; small disjuncts, lack of density in the training set, class overlapping, noisy data identification, borderline instances, and the data set shift between the training and the test distributions. Their experiments on imbalanced data lead to important observations on the reaction of the machine learning algorithms on data with these intrinsic characteristics. One of the most notable approaches is that of Chawla et al. [17]. They proposed a hybrid approach for classification on imbalanced data which achieved better performance compared to exclusively undersampling the majority class. Their oversampling method, also known as SMOTE, produces synthetic minority class examples. Their experiments were performed with C4.5, Ripper and Naive Bayes, while their method was evaluated with the area under the Receiver Operating Characteristic curve (AUC) and the Receiver Operating Characteristic (ROC) convex hull strategy. The SMOTE oversampling method is used in this paper to balance the data set (Section 5).

3. Data Set Creation & Preprocessing

The data set which was used in the authors' conference paper [19] was created by scraping 57 articles written in English from Wikipedia's API[4] with Python (BeautifulSoup4[5], Requests[6]). The criterion for selecting these specific articles was their relevance to 5 vocational domains considered as the most common for refugee and migrant employment in Europe, Canada and the United States of America [1,2,6–8].

The initial textual data set comprised of 6,827 sentences extracted from the 57 Wikipedia articles. The data set was preprocessed in 4 stages, namely:

1. Initial preprocessing & Tokenization;
2. Numbers & Punctuation marks removal;
3. Stopwords removal;
4. Lemmatization & Duplicates removal.

The data set was tokenized initially to 6,827 sentences and to 69,062 words; the sentences were to be used as training-testing examples, and the words as unigram features. Numbers, punctuation marks and special characters were removed. Stopwords (conjunctions, articles, adverbs, pronouns, auxiliary verbs, etc.) were also removed. Finally, lemmatization was performed to normalize the data without reducing the semantic information, and 912 duplicate sentences and 58,393 duplicate words were removed. For more details on these stages of preprocessing refer to [19].

Resulting from the preprocessing stages, the text data set comprised of 5,915 sentences (examples) and 5 classes, ready to be used in machine learning experiments. For each sentence, the domain which was most relevant to each article's topic, as shown in Table 2, was considered as its class, resulting in 5 distinct classes, namely: A. Agriculture, B. Cooking, C. Crafting, D. Construction, and E. Hospitality. The distribution of the sentences to these 5 classes is shown in Figure 1.

[4] https://pypi.org/project/wikipedia/
[5] https://pypi.org/project/beautifulsoup4/
[6] https://pypi.org/project/requests/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

Table 2. Wikipedia articles which were scraped to create the data set. By domain categorization.

Agriculture *10 articles*	**Cooking** *17 articles*	**Crafting** *11 articles*	**Construction** *7 articles*	**Hospitality** *12 articles*
Agriculture	Al dente	Anvil	Building	Bellhop
Glossary of agriculture	Al forno	Blacksmith	Building design	Casino hotel
Farm	Baking	Bladesmith	construction	Check-in
Farmer	Charcuterie	Coppersmith	Carpentry	Concierge
Environmental impact of agriculture	Chef	Forge	Construction	Doorman(profession)
History of agriculture	Chef's uniform	Goldsmith	Constructor worker	Hostel
Intensive farming	Chocolate	Gunsmith	Glossary of construction costs	Hotel
Plant breeding	Cooking	Locksmithing	Home construction	Hotel manager
Subsistence agriculture	Cooking school	Metalsmith		Maid
Sustainable agriculture	Cooking weights and measures	Silversmith		Receptionist
	Cuisine	Whitesmith		Resort
	Denaturation(food)			Tourism
	Garde manger			
	List of cooking techniques			
	Mise en place			
	Outdoor cooking			
	Outline of food preparation			

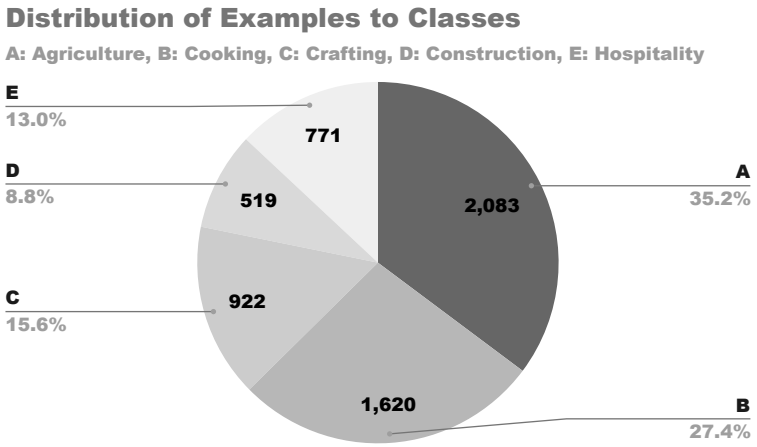

Figure 1. Final distribution of sentences as training-test examples to the 5 classes.

A RapidMiner Studio (version 9.10) process, as shown in Figure 2, was used to extract the feature set with TF-IDF and taking into consideration the feature occurrences by pruning features which occur rarely (below 1%) or very often (above 30%), resulting in 109 unigram features. For more details on the operators and parameters of the feature extraction process refer to [19] and RapidMiner Documentation[7].

Resulting from the feature extraction process, the final text data set comprised of 5,915 examples, 109 features and *class* as *label*.

4. Predictions Analysis

The best results on domain identification were obtained with a *Gradient Boosted Trees*[8] model and are shown in Table 3. For more information on the operators and parameters of

[7] https://docs.rapidminer.com/

[8] https://docs.rapidminer.com/9.10/studio/operators/modeling/predictive/trees/gradient_boosted_trees.html

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

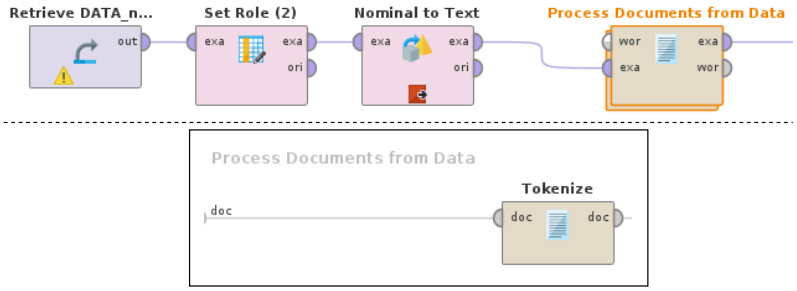

Figure 2. Feature extraction. The *Tokenize* operator is nested in the *Process Documents from Data* operator.

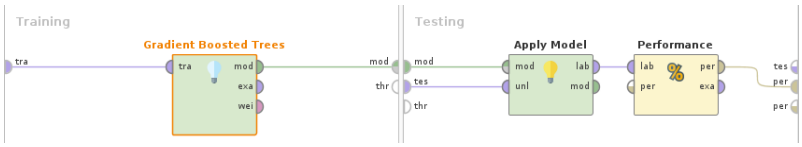

Figure 3. Setup of machine learning experiment with *Gradient Boosted Trees*. The depicted process is nested in a *Cross Validation* operator (10-fold cross validation).

the RapidMiner Studio (version 9.10) experiment with *Gradient Boosted Trees*, as shown in Figure 3, refer to [19] and RapidMiner Documentation.

Table 3. Machine learning experiment results with *Gradient Boosted Trees*. Accuracy: 99.93%.

Class	Precision	Recall	F1 score
A	100%	100%	100%
B	100%	99.94%	99.97%
C	99.78%	99.89%	99.83%
D	99.81%	99.61%	99.70%
E	99.87%	100%	99.93%

With regards to the high performance of this machine learning model, it is of interest to examine which examples were classified wrongly for each class, as well as which distinct features contributed to their misclassification. In the same line of thought, regarding the correctly classified examples, the examination of the features that were the most dominant leading to the correct predictions would contribute to the identification of a primary set of terms highlighting the terminology of the vocational domains.

4.1. Wrong Predictions

The *Gradient Boosted Trees* model showed high performance regarding all classes (Table 3), with precision ranging from 99.78% to 100%, recall from 99.61% to 100%, F1 score from 99.70% to 100%, while misclassifying a total of 4 examples.

The *Explain Predictions*[9] operator is used to identify which features are the most dominant in forming predictions. A model and a set of examples, along with the feature set, are considered as input, in order to produce a table highlighting the features that most strongly support or contradict each prediction, also containing numeric details. For each example, a neighboring set of data points is generated, by using correlation to define the local feature weights in that neighborhood. The operator can calculate model-specific, though model-agnostic global feature weights, deriving directly from the explanations. *Explain Predictions* is able to work with all data types and data sizes, and can be applied for both classification and regression problems.

[9] https://docs.rapidminer.com/10.1/studio/operators/scoring/explain_predictions.html

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

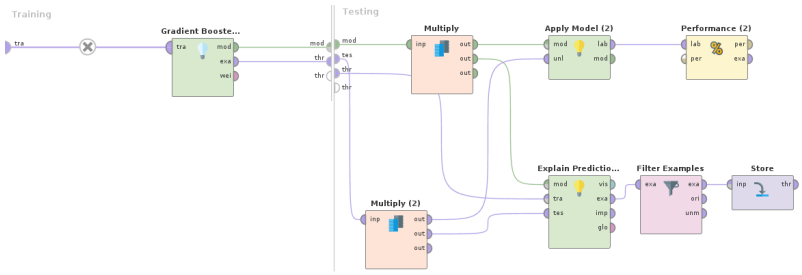

Figure 4. Setup of process to identify wrong and correct predictions with *Explain Predictions* operator and *Filter Examples* operator. The depicted process is nested in a *Cross Validation* operator (10-fold cross validation).

In this case, in which the machine learning model (*Gradient Boosted Trees*) uses supervised learning, all supporting local explanations add positively to the weights for correct predictions, while all contradicting local explanations add positively to the weights for wrong predictions. Regarding the parameters for this operator, the *maximal explaining attributes* were set to 3 and the *local sample size* was left at the default (500). The *sort weights* parameter was set to true, along with descending *sort direction* of the weight values, in order to apply sorting to the resulting feature weights supporting and contradicting the predictions.

The *Filter Examples*[10] operator selects which examples are kept and which are removed. In this case, only the misclassified examples (wrong predictions) were kept. Regarding the *condition class* parameter for this operator, it was set to wrong_predictions, in order to keep only those examples, where the class and prediction were different, meaning that the prediction is wrong.

In order to identify the misclassified examples, a RapidMiner Studio (version 9.10) process, as shown in Figure 4, was designed and executed.

The 4 misclassified examples were the following:

1. WP1: building edifice structure roof wall standing permanently house factory;
2. WP2: typically whitesmiths product required decorative finish fire grate coldworking screw lathed machine;
3. WP3: organic food;
4. WP4: traditional vernacular building method suit local condition climate dispensed favour generic cookie cutter housing type.

In Table 4, detailed information is provided for these wrong predictions. Class is the real class of the example, while prediction is the wrongly predicted class for the example. Confidence, with values ranging from 0 to 1, is derived from feature weights regarding both class and prediction.

Table 4. Wrong predictions of *Gradient Boosted Trees*. *Class* is the real class of the example, and *Prediction* is the wrongly predicted class for the example. *Confidence*, ranging from 0 to 1, and derived from feature weights regarding both *Class* and *Prediction* is shown in the last 2 columns.

No.	Class	Prediction	Confidence (Class)	Confidence (Prediction)
WP1	D	C	0.14	0.41
WP2	C	D	0.12	0.48
WP3	B	C	0.11	0.55
WP4	D	E	0.17	0.31

The features which contributed to the wrong predictions for each class are shown in Table 5. The effect of the value for each feature is denoted, considering whether it supports,

[10] https://docs.rapidminer.com/10.1/studio/operators/blending/examples/filter/filter_examples.html

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

contradicts or is neutral to the prediction. The typical value for the specific feature for each class is also provided.

Table 5. Features that contributed to the wrong predictions of *Gradient Boosted Trees*. *Effect* denotes whether the specific value of the specific feature supports, contradicts or is neutral to the prediction. *Typical Value* is the typical value for the specific feature for each class.

No.	Feature(s)	Value(s)	Effect(s)	Typical Value
WP1	building	1	Neutral	D: 0 C: 0 and some 1
WP2	typically fire product	1 0.66 0.54	Neutral Contradict	C & D: 0 and some 1
WP3	food organic	0.50 0.86	Support	B: 0 & C: 1 B: 0 & C: 0 and some 1
WP4	local method type	0.56 0.47 0.46	Support	D: 0 & E: 0 and some 1

4.2. Correct Predictions

The *Gradient Boosted Trees* model managed to classify correctly most of the examples. Regarding class A, it is of particular interest that all of its examples were classified correctly, while none of the examples of the other classes were classified wrongly to class A. Consequently, it is of significance to identify and examine which features were the most dominant leading to the correct predictions for each class, thus contributing to the identification of a primary set of terms for the vocational domains.

In order to identify the correctly classified examples, the same RapidMiner Studio (version 9.10) process, as in wrong predictions (Figure 4), was used. The only difference with the previous experiment is that the *condition class* parameter for the *Filter Examples* operator was set to correct_predictions, in order to keep only those examples, where the class and prediction were the same, meaning that the prediction is correct.

Confidence, with values that can be from 0 to 1, was derived from feature weights for each class; for class A ranging from 0.49 to 0.55, for class B ranging from 0.37 to 0.55, for class C ranging from 0.48 to 0.55, for class D ranging from 0.47 to 0.55, and for class E ranging from 0.54 to 0.55. The features which were the most dominant leading to the correct predictions are shown in Table 6 in a descending order, along with the global weights which were calculated for each one of them.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

Table 6. Global weights per feature (descending order). Features with higher weight were more dominant for the correct predictions of this model than features with lower weight.

No.	Feature	Weight	No.	Feature	Weight
1	farmer	0.037	56	time	0.018
2	world	0.036	57	usually	0.018
3	blacksmith	0.034	58	cocoa	0.018
4	using	0.034	59	grain	0.017
5	produce	0.033	60	material	0.017
6	human	0.032	61	chef	0.017
7	developed	0.031	62	growing	0.017
8	plant	0.03	63	process	0.017
9	yield	0.029	64	water	0.017
10	food	0.029	65	form	0.017
11	project	0.029	66	industry	0.016
12	temperature	0.029	67	fat	0.016
13	environmental	0.028	68	field	0.016
14	ingredient	0.028	69	found	0.016
15	america	0.028	70	domesticated	0.015
16	technique	0.028	71	product	0.015
17	united	0.027	72	sometimes	0.015
18	design	0.027	73	europe	0.015
19	heat	0.027	74	crop	0.015
20	system	0.027	75	source	0.015
21	quality	0.026	76	anvil	0.014
22	iron	0.026	77	variety	0.014
23	breeding	0.026	78	various	0.013
24	local	0.025	79	livestock	0.013
25	vegetable	0.025	80	tourism	0.013
26	typically	0.025	81	farm	0.013
27	increase	0.025	82	construction	0.013
28	land	0.024	83	practice	0.013
29	cost	0.024	84	building	0.013
30	agricultural	0.024	85	people	0.012
31	sustainable	0.024	86	natural	0.012
32	common	0.023	87	example	0.012
33	called	0.023	88	level	0.012
34	service	0.023	89	animal	0.012
35	period	0.023	90	organic	0.012
36	cuisine	0.022	91	soil	0.011
37	trade	0.022	92	resort	0.011
38	production	0.022	93	cooking	0.011
39	operation	0.022	94	meat	0.011
40	country	0.022	95	especially	0.01
41	farming	0.022	96	population	0.01
42	include	0.021	97	fire	0.01
43	global	0.021	98	hotel	0.01
44	effect	0.021	99	modern	0.009
45	increased	0.021	100	century	0.009
46	type	0.02	101	change	0.009
47	agriculture	0.02	102	chocolate	0.009
48	method	0.02	103	metal	0.008
49	fertilizer	0.02	104	including	0.008
50	amount	0.02	105	steel	0.008
51	baking	0.02	106	smith	0.007
52	tool	0.019	107	text	0.007
53	oven	0.019	108	management	0.006
54	worker	0.018	109	due	0.006
55	hot	0.018			

 doi:10.20944/preprints202304.0907.v1

4.3. Discussion

Regarding the wrong predictions analysis, the 4 misclassified examples were successfully identified (WP1-WP4), as shown in Table 4. More specifically, 2 examples of class D, namely WP1 and WP4, were wrongly classified to classes C and E, respectively, while 1 example of class C, WP2, was misclassified to class D, and 1 example of class B, WP3, was misclassified to class C. It is observed that for all wrong predictions, the confidence for class, which is the real class of the examples, ranges from 0.11 to 0.17 and is significantly lower than the confidence for prediction, which is the wrongly predicted class of the examples and ranges from 0.31 to 0.55. This indicates that these examples quite diverge from the other examples of their class. By examining Tables 5 and 6, this observation is explained as described below.

For WP1, the value for the *building* feature is 1, while typically for examples of D (class) is 0 and of C (prediction) mostly 0 and sometimes 1. Considering that *building* is the only most dominant feature of WP1, with an assigned feature weight of 0.013, its overall impact on the prediction being neutral is expected.

For WP2, the value for the *typically* feature is 1, for the *fire* feature is 0.66, and for the *product* feature is 0.54, while typically the values of all these features for examples of both C (class) and D (prediction) is mostly 0 and sometimes 1. Considering that *typically* is the most dominant feature of WP2, with an assigned feature weight of 0.025 which is high, its overall impact on the prediction being neutral is expected. The *fire* and *product* features contradict the prediction, though due to their quite low feature weights of 0.01 and 0.015, respectively, their effect on the prediction is insignificant.

For WP3, the value for the *food* feature is 0.50 and for the *organic* feature is 0.86, while typically for examples of B (class) is 0 for both features and of C (prediction) is 1 for the *food* feature, and mostly 0 and sometimes 1 for the *organic* feature. Considering that *food* is the most dominant feature of WP3, with an assigned feature weight of 0.029 which is high, its overall impact on the prediction being positive (support) is expected. The *organic* feature also supports the prediction, though due to its quite low feature weight (0.012) its effect on the prediction is insignificant.

For WP4, the value for the *local* feature is 0.56, for the *method* feature is 0.47, and for the *type* feature is 0.46, while typically the values of all these features for examples of D (class) is 0 and E (prediction) is mostly 0 and sometimes 1. Considering that *local* is the most dominant feature of WP4, with an assigned feature weight of 0.025 which is high, its overall impact on the prediction being positive (support) is expected. The *method* and *type* features also support the prediction, with quite high feature weights of 0.02 for both, having a significant effect on the prediction.

Overall, it becomes evident that the main factor that leads the *Gradient Boosted Trees* model to misclassify the examples is their lack of dominant features supporting the real class more than the prediction, in terms of feature weight.

Regarding the correct predictions analysis, it is observed that the confidence for the correct predictions for all classes was considerably high, the lower being for class B in a range from 0.37 to 0.55 and the highest for class E in a range from 0.54 to 0.55. This means that the model could classify the examples of class E more confidently compared to the examples of the other classes.

Additionally, the most dominant features, in terms of feature weights, leading to the correct predictions for each class were identified successfully and sorted in a descending order, as shown in Table 6; features with higher weight were more dominant for the correct predictions of this model than features with lower weight. 51 features, which are about half of the 109 features of the extracted feature set, have the highest feature weights, ranging from 0.02 up to 0.037. This indicates that the feature extraction process, as described in Section 3 and [19], performed quite well, producing a robust feature set with great impact on the correct predictions. Finally, it is also observed that among these features, terms relevant to all of the vocational domains are included, thus consisting a primary set of terms for the vocational domains.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

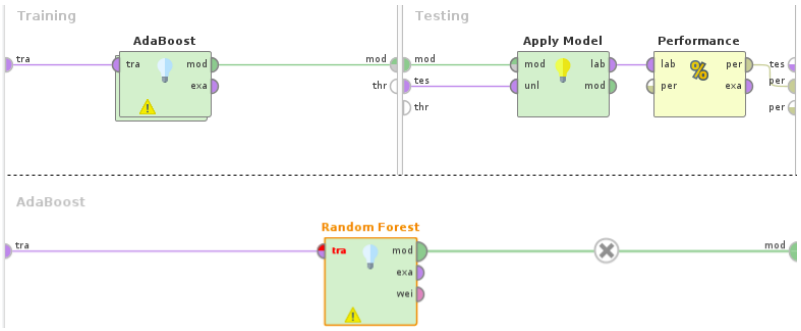

Figure 5. Setup of machine learning experiment with *Random Forest* and *AdaBoost*. The *Random Forest* operator is nested in the *AdaBoost* operator. The depicted process is nested in a *Cross Validation* operator (10-fold cross validation).

5. Data Balancing

Another machine learning experiment on domain identification was performed with a *Random Forest*[11] and *AdaBoost*[12] model. The results of this experiment are shown in Table 7. For more information on the operators and parameters of the RapidMiner Studio (version 9.10) experiment with *Random Forest* and *AdaBoost*, as shown in Figure 5 refer to [19] and RapidMiner Documentation[13].

Table 7. Machine learning experiment results with *Random Forest* and *AdaBoost*. Accuracy: 62.33%.

Class	Precision	Recall	F1 score
A	49.06%	97.60%	65.29%
B	91.52%	51.30%	65.74%
C	95.05%	41.65%	57.92%
D	91.67%	31.79%	47.20%
E	98.21%	35.54%	52.19%

Regarding the model's accuracy of 62.33%, it is important to bear in mind that, despite being quite lower than the accuracy of the *Gradient Boosted Trees* model (99.93%), it is significantly above the randomness baseline by 42.33%, considering that the randomness for a 5-class problem is at 20%.

Examining the model's results (Table 7) more closely, it was noted that, despite its precision for classes B, C, D, and E being high, ranging from 91.52% to 98.21%, the recall for these classes was low, ranging from 31.79% to 51.30%. Also considering its low precision (49.06%) and high recall (97.60%) for class A, this examination highlighted that a lot of the examples were classified wrongly to class A. As a result, it becomes evident that the *Random Forest* and *AdaBoost* model tended to classify most of the examples to class A. Due to the fact that the examples of class A consist the majority of the examples in the data set (35.20%, Figure 1), this tendency can be attributed to the imbalance of data.

Consequently, it is of interest to examine whether applying data balancing techniques on the data set (oversampling and undersampling), has any impact, positive or negative, on the performance of the *Random Forest* and *AdaBoost* model.

5.1. Data Oversampling

As a first step towards addressing data imbalance, SMOTE oversampling [17] was applied in a successive manner on the data set, balancing the data by oversampling, regarding which is the minority class each time. Consequently, a RapidMiner Studio (version

[11] https://docs.rapidminer.com/9.10/studio/operators/modeling/predictive/trees/parallel_random_forest.html
[12] https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/ensembles/AdaBoost.html
[13] https://docs.rapidminer.com/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

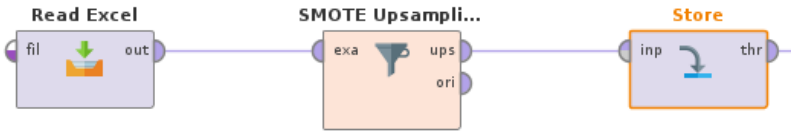

Figure 6. Setup of *SMOTE Upsampling*.

9.10) process, as shown in Figure 6, was designed and executed 4 times. The 4 derived oversampled data sets were then used as input for the machine learning experiments with *Random Forest* and *AdaBoost*.

The *SMOTE Upsampling*[14] operator practically applies the Synthetic Minority Oversampling Technique, as defined in the paper by Chawla et. al. [17]. More specifically, the algorithm considers only the examples of the minority class, and the k nearest neighbours for each example are searched. Then a random example and a random nearest neighbour for this example is selected, resulting in the creation of a new example, on the line between the two examples.

Regarding the parameters for this operator, the *number of neighbours* was left at the default (5), while *normalize* and *round integers* were set to true, and *nominal change rate* was set to 0.5, in order to make the distance calculation solid. The *equalize classes* parameter was set to true to draw the necessary amount of examples for class balance, along with *auto detect minority class* set to true to automatically upsample the class with the least occurrences.

The set of machine learning experiments with successive applications of SMOTE oversampling as described below, follows a novel methodology, proposed in this paper for the first time. The methodology steps are the following:

1. Detect the minority class;
2. Resample the minority class with SMOTE oversampling;
3. Run the machine learning experiment;
4. Repeat steps 1-3 until the data set is balanced (no minority class exists).

By running the experiments following this methodology, the impact of every class distribution, from completely imbalanced to completely balanced, on the performance of the machine learning model can be examined thoroughly.

In the first machine learning experiment, class D is the minority class, with its examples representing merely the 8.8% of the data set (Figure 1). After applying SMOTE, class D represented 27.9% of the data set with 2,083 examples (Figure 7). The results of the *Random Forest* and *AdaBoost* with *SMOTE* are shown in Table 8.

Table 8. Machine learning experiment results with *Random Forest* and *AdaBoost* with *SMOTE*. Accuracy: 66.01%.

Class	Precision	Recall	F1 score
A	94.30%	69.13%	79.77%
B	92.67%	49.20%	64.27%
C	94.47%	38.94%	55.14%
D	46.65%	99.33%	63.48%
E	98.19%	35.28%	51.90%

In the second machine learning experiment, class E is the minority class, with its examples representing the 10.3% of the data set (Figure 7). After applying SMOTE, class E represented 23.7% of the data set with 2,083 examples (Figure 8). The results of the *Random Forest* and *AdaBoost* with *SMOTE* (2 times) are shown in Table 9.

[14] https://docs.rapidminer.com/10.1/studio/operators/extensions/Operator%20Toolbox/blending/smote.html

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

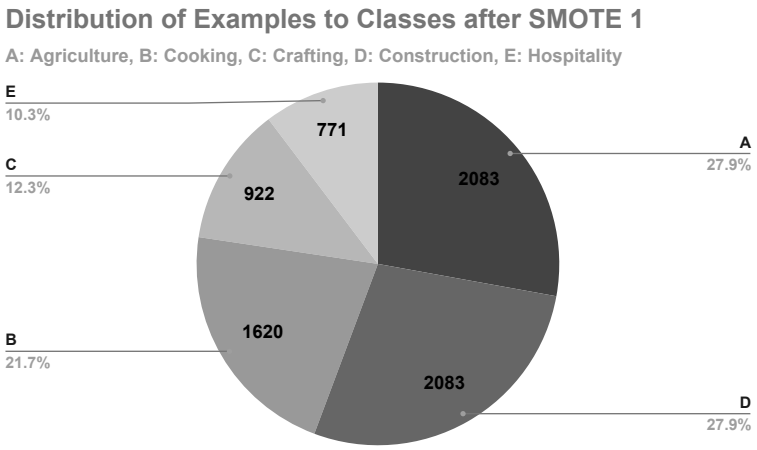

Figure 7. Distribution of examples to classes after applying SMOTE (1 time).

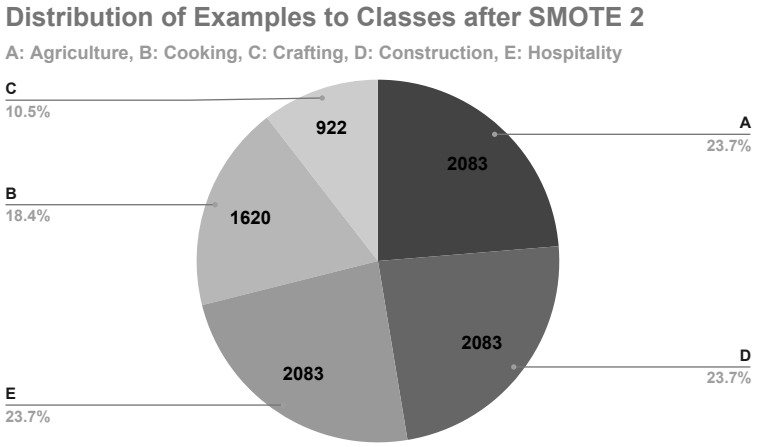

Figure 8. Distribution of examples to classes after applying SMOTE (2 times).

Table 9. Machine learning experiment results with *Random Forest* and *AdaBoost* with *SMOTE* (2 times). Accuracy: 65.72%.

Class	Precision	Recall	F1 score
A	93.46%	67.93%	78.67%
B	92.20%	47.41%	62.62%
C	94.51%	37.31%	53.49%
D	60.50%	72.35%	65.89%
E	48.57%	83.68%	61.46%

In the third machine learning experiment, class C is the minority class, with its examples representing the 10.5% of the data set (Figure 8). After applying SMOTE, class C represented 20.9% of the data set with 2,083 examples (Figure 9). The results of the *Random Forest* and *AdaBoost* with *SMOTE* (3 times) are shown in Table 10.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

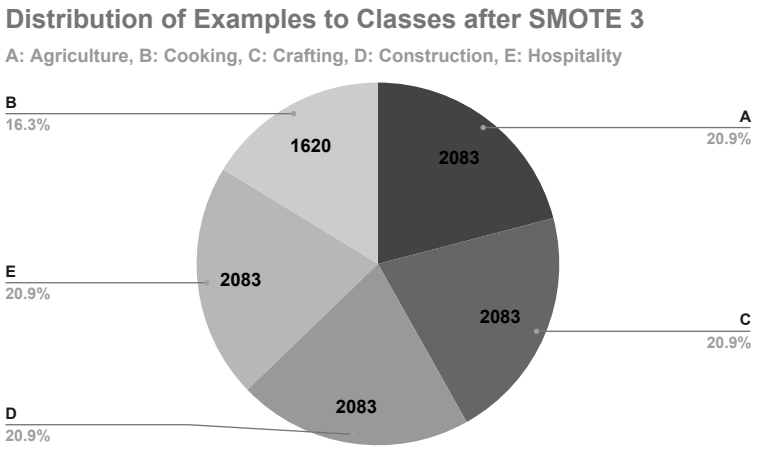

Figure 9. Distribution of examples to classes after applying SMOTE (3 times).

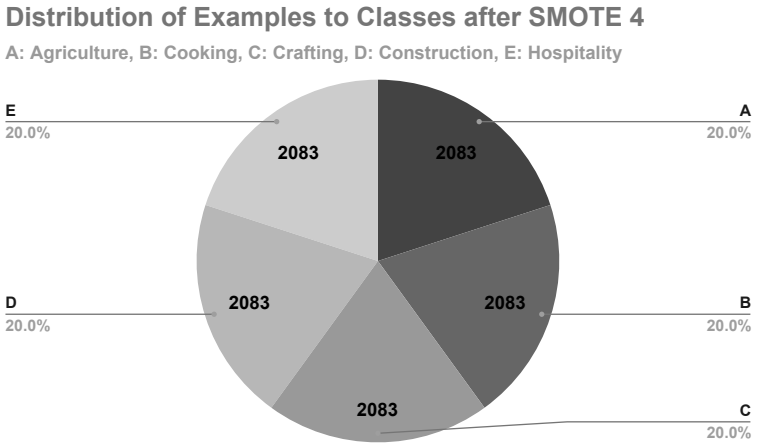

Figure 10. Distribution of examples to classes after applying SMOTE (4 times).

Table 10. Machine learning experiment results with *Random Forest* and *AdaBoost* with *SMOTE* (3 times). Accuracy: 64.35%.

Class	Precision	Recall	F1 score
A	95.14%	66.73%	78.44%
B	91.72%	46.48%	61.69%
C	38.44%	96.54%	54.98%
D	89.68%	58.38%	70.72%
E	95.48%	49.64%	65.32%

In the fourth machine learning experiment, class B is the minority class, with its examples representing the 16.3% of the data set (Figure 9). After applying SMOTE, class B represented 20% of the data set with 2,083 examples (Figure 10). The results of the *Random Forest* and *AdaBoost* with *SMOTE* (4 times) are shown in Table 11.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

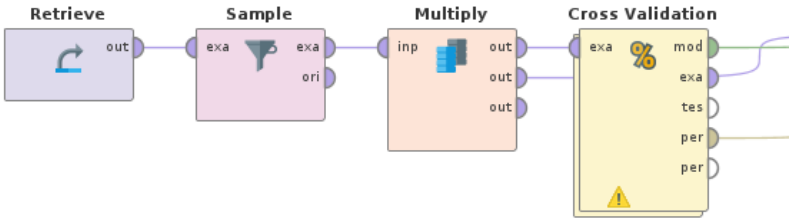

Figure 11. Setup of machine learning experiment with *Random Forest* and *AdaBoost* with *Sample*. The *Random Forest* operator is nested in the *AdaBoost* operator, which is nested in a *Cross Validation* operator (10-fold cross validation).

Table 11. Machine learning experiment results with *Random Forest* and *AdaBoost* with *SMOTE* (4 times). Accuracy: 64.09%.

Class	Precision	Recall	F1 score
A	95.52%	65.53%	65.29%
B	40.79%	85.84%	65.74%
C	58.84%	62.17%	57.92%
D	90.00%	58.33%	47.20%
E	96.20%	48.58%	52.19%

5.2. Data Undersampling

In another set of experiments, undersampling was applied on the data set, balancing the data by undersampling the classes represented by the most examples. Consequently, a RapidMiner Studio (version 9.10) process, as shown in Figure 11, was designed and executed. The derived undersampled data set was then used as input for the machine learning experiments with *Random Forest* and *AdaBoost*.

The *Sample*[15] operator has basic principles common to the *Filter Examples* operator, taking a set of examples as input and procuring a subset of it as output. However, while *Filter Examples* follows previously specified conditions, *Sample* is centered on the number of examples and class distribution in the resulting subset, producing samples in a random manner.

Regarding the parameters for this operator, *sample* was set to absolute, in order for it to be created consisting of an exact number of examples. The *balance data* parameter was set to true, in order to define different sample sizes (by number of examples) for each class, while the class distribution of the sample was set with *sample size per class*. Examples of classes A and B were reduced to 1,183 for each one, which is the mean of the number of all examples in the data set. The sample sizes for each class are shown in Figure 12. The results of this experiment are shown in Table 12.

Table 12. Machine learning experiment results with *Random Forest* and *AdaBoost* with *Sample*. Accuracy: 62.84%.

Class	Precision	Recall	F1 score
A	94.34%	66.27%	77.85%
B	41.77%	96.53%	58.30%
C	94.94%	44.79%	60.86%
D	88.57%	41.81%	56.80%
E	96.40%	41.63%	58.14%

A hybrid approach combining data oversampling and undersampling was also tested. In this experiment, both the *SMOTE Upsampling* operator and the *Sample* operator were

[15] https://docs.rapidminer.com/10.1/studio/operators/blending/examples/sampling/sample.html

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 April 2023 doi:10.20944/preprints202304.0907.v1

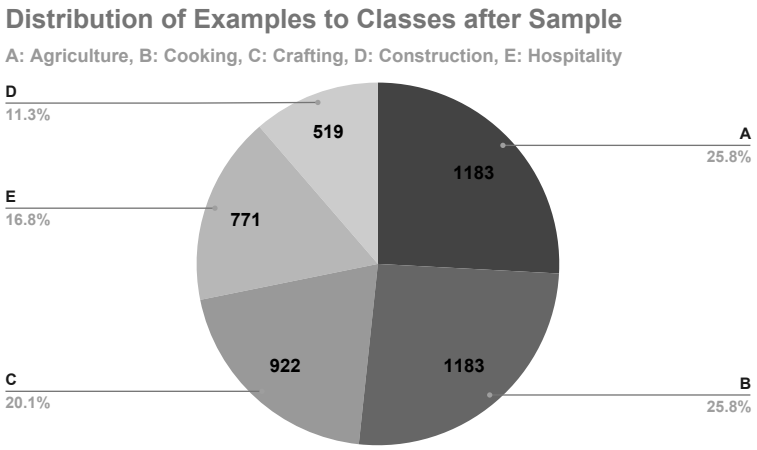

Figure 12. Distribution of examples to classes after applying Sample.

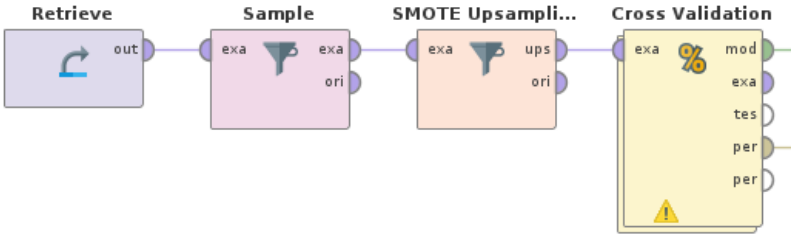

Figure 13. Setup of machine learning experiment with *Random Forest* and *AdaBoost* with *Sample* and *SMOTE Upsampling*. The *Random Forest* operator is nested in the *AdaBoost* operator, which is nested in a *Cross Validation* operator (10-fold cross validation).

applied on the data set, balancing the data by undersampling the classes represented by the most examples and oversampling the classes represented by the least examples, respectively. Consequently, a RapidMiner Studio (version 9.10) process, as shown in Figure 13, was designed and executed. The derived undersampled data set was then used as input for the machine learning experiments with *Random Forest* and *AdaBoost*.

Regarding the parameters for *Sample* and *SMOTE Upsampling*, they were set in the same way as in the previous experiments. After applying them, examples of classes A and B were reduced to 1,183 for each one, which is the mean of the number of all examples in the data set, while examples of class D were added resulting also in 1,183 for this class. The sample sizes for each class are shown in Figure 14. The results of this experiment are shown in Table 13.

Table 13. Machine learning experiment results with *Random Forest* and *AdaBoost* with *Sample* and *SMOTE Upsampling*. Accuracy: 63.35%.

Class	Precision	Recall	F1 score
A	95.29%	66.69%	78.46%
B	39.79%	95.52%	56.17%
C	93.62%	44.58%	60.39%
D	83.27%	57.23%	67.83%
E	97.52%	40.73%	57.46%

5.3. Discussion

Regarding the machine learning experiments' results with *Random Forest* and *AdaBoost* with *SMOTE* oversampling, it is observed that the accuracy and overall performance, as shown in Tables 8, 9, 10, and 11, have improved compared to those of *Random Forest*

 doi:10.20944/preprints202304.0907.v1

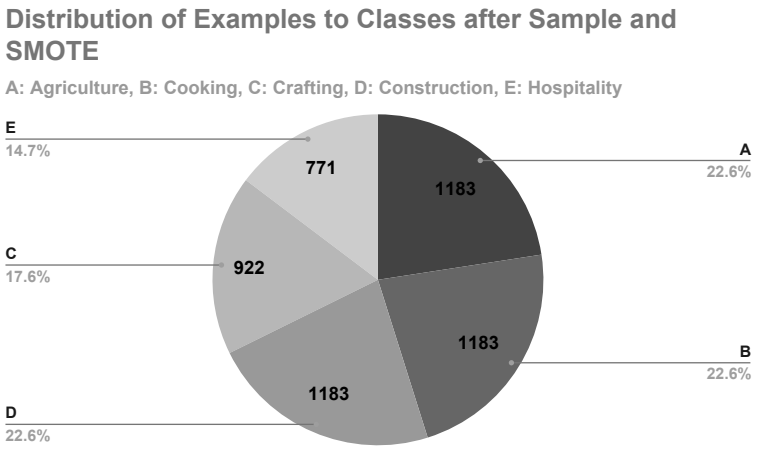

Figure 14. Distribution of examples to classes after applying Sample and SMOTE.

and *AdaBoost* with imbalanced data, as shown in Table 7. More specifically, accuracy has increased from 62.33% up to 66.01%, and F1 score increased from 65.29% up to 79.77% for class A, maintained up to 65.74% for class B, maintained up to 57.92% for class C, increased from 47.20% up to 70.72% for class D, and increased from 52.19% up to 65.32% for class E. It is also noteworthy that, despite the overall performance of the model getting slightly worse with each iteration (each added SMOTE oversampling), it is still significantly better than the performance of the experiment with completely imbalanced data; even the lowest accuracy (64.09%), which is that of the fourth machine learning experiment with SMOTE, is quite higher than the accuracy (62.33%) of the experiment with completely imbalanced data. Additionally, the values of precision, recall and F1 score seem to be distributed more evenly among the classes with each iteration, thus mitigating any emerging bias of the model towards one particular class. Another important observation from these experiments is that, in a classification task where 1 of the 5 vocational domains may be considered as the class of interest, e.g. for trying to exclusively detect articles of a specific vocational domain from a corpus to filter relevant content, the application of SMOTE oversampling for the class of interest would have a positive effect on the results of this classification task.

Regarding the machine learning experiments' results with *Random Forest* and *AdaBoost* with *Sample*, it is observed that the accuracy and overall performance, as shown in Table 12, have improved slightly compared to those of *Random Forest* and *AdaBoost* with imbalanced data, as shown in Table 7. More specifically, accuracy has increased from 62.33% to 62.84%, and F1 score increased from 65.29% to 77.85% for class A, reduced from 65.74% to 58.30% for class B, increased from 57.92% to 60.86% for class C, increased from 47.20% to 56.80% for class D, and increased from 52.19% to 58.14% for class E. Comparing to the results obtained with SMOTE oversampling (Tables 8, 9, 10, and 11), undersampling has worse performance in terms of accuracy and class precision, recall and F1 score.

Regarding the machine learning experiments' results with *Random Forest* and *AdaBoost* with *Sample* and *SMOTE* oversampling (hybrid approach), it is observed that the accuracy and overall performance, as shown in Table 13, have marginally improved compared to those of *Random Forest* and *AdaBoost* only with *Sample* (Table 12). More specifically, accuracy has increased from 62.84% to 63.35%, and F1 score increased from 77.85% to 78.46% for class A, reduced from 58.30% to 56.17% for class B, reduced from 60.86% to 60.39% for class C, increased from 56.80% to 67.83% for class D, and reduced from 58.14% to 57.46% for class E. In any case, the performance of this experiment is better than that of the experiment with completely imbalanced data. Overall, these experiments indicate that when applying both data undersampling and oversampling in a hybrid approach, the results are better than only applying undersampling, but worse than only applying oversampling for this data set.

 doi:10.20944/preprints202304.0907.v1

The findings derived from the machine learning experiments of this paper are in accordance with those of the relevant literature [12,17], in terms that data oversampling obtains better results than data undersampling in imbalanced data sets, while hybrid approaches perform reasonably well.

6. Conclusions

Displaced communities, like migrants and refugees, face multiple challenges in seeking and finding employment in high-skill vocations in their host country, deriving from discrimination. Unemployment and overworking phenomena usually affect more the displaced communities than the natives. A deciding factor for their prospects of employment is the knowledge of not the language of their host country in general, but specifically of the sublanguage of the vocational domain they are interested in working. Consequently, more and more highly-skilled migrants and refugees worldwide are finding employment in low-skill vocations, despite their professional qualifications and educational background, with the language barrier being one of the most important factors. Both high-skill and low-skill vocations in agriculture, cooking, crafting, construction, and hospitality, among others, consist the most common vocational domains in which migrants and refugees seek and find employment, according to the findings of the recent research.

In the last decade, due to the expansion of the user base of wikis and social networks user-generated content has increased exponentially, providing a valuable source of data for various tasks and applications in data mining, Natural Language Processing and machine learning. However, minority class examples are the most difficult to obtain from real data, especially from user-generated content from wikis and social networks, creating a class imbalance problem that affects various aspects of real-world applications that are based on classification. Especially for multi-class problems, like the one addressed in this paper, it is more challenging to solve.

This paper extends the contribution of the authors' previous research [19] on automatic vocational domain identification by further processing and analyzing the results of the machine learning experiments with the domain-specific textual data set, considering 2 research directions: a. predictions analysis and b. data balancing.

Regarding the predictions analysis direction, important conclusions were drawn from identifying successfully and examining the 4 misclassified examples (WP1-WP4) for each class (wrong predictions) by the *Gradient Boosted Trees* model, which managed to classify correctly most of the examples, as well as which distinct features contributed to their misclassification. An important finding is that the misclassified examples quite diverge from the other examples of their class, since for all wrong predictions, the confidence for class, which is the real class of the examples, is significantly lower (from 0.11 to 0.17) than the confidence for prediction (from 0.31 to 0.55), which is the wrongly predicted class of the examples. More specifically, the feature values of WP1-WP4 are the main factor for their misclassification, by being either neutral or supporting more the wrong over the correct prediction. Even when they contradict the wrong prediction, like the features of WP2 and WP3, they do not have a significant effect due to their feature weights being quite low. In conclusion, the main factor that leads the *Gradient Boosted Trees* model to misclassify the examples is their lack of dominant features supporting the real class more than the prediction, in terms of feature weight.

In the same line of thought, the examination of the correctly classified examples (correct predictions) resulted in several findings. The confidence for the correct predictions for all classes was considerably high, the lower being for class B (from 0.37 to 0.55) and the highest for class E (from 0.54 to 0.55), meaning that the model could classify the examples of class E more confidently compared to the examples of the other classes. Additionally, the most dominant features, in terms of feature weights, leading to the correct predictions for each class were identified successfully and sorted in a descending order; features with higher weight were more dominant for the correct predictions of this model than features with lower weight. Another important finding concerning the most dominant features is

 doi:10.20944/preprints202304.0907.v1

the fact that about half of the features of the extracted feature set have the highest feature weights (from 0.02 up to 0.037), therefore indicating that the feature extraction process, as described in Section 3 and [19], performed quite well, producing a robust feature set with great impact on the correct predictions. It is important to note that among these features, terms relevant to all of the vocational domains are included, thus consisting a primary set of terms for the vocational domains.

Regarding the data balancing direction, oversampling and undersampling techniques, both separately and in combination as a hybrid approach, were applied on the data set, in order to observe their impact (positive or negative) on the performance of the *Random Forest* and *AdaBoost* model. A novel methodology is proposed in this paper for the first time, consisting of successive applications of SMOTE oversampling on imbalanced data, balancing the data considering which is the minority class each time, in 4 steps. By running the experiments following this methodology, the impact of every class distribution, from completely imbalanced to completely balanced, on the performance of the machine learning model can be examined thoroughly. The process of the class balancing direction enabled the comparison of the performance of this model with balanced data to the performance of the same model with imbalanced data from the previous research [19].

More specifically, the machine learning experiments' results with *Random Forest* and *AdaBoost* with *SMOTE* oversampling have obtained significantly improved overall performance and accuracy (up to 66.01%) compared to those of *Random Forest* and *AdaBoost* with imbalanced data, while maintaining or surpassing the achieved F1 score per class. A major finding is that, despite the overall performance of the model getting slightly worse with each iteration (each added SMOTE oversampling), it is still significantly better than the performance of the experiment with completely imbalanced data; even the lowest accuracy (64.09%), which is that of the fourth machine learning experiment with SMOTE, is quite higher than the accuracy (62.33%) of the experiment with completely imbalanced data. Moreover, the values of precision, recall and F1 score seem to be distributed more evenly among the classes with each iteration, thus mitigating any emerging bias of the model towards one particular class. Another important finding is that, in a classification task where 1 of the 5 vocational domains may be considered as the class of interest, e.g. for trying to exclusively detect articles of a specific vocational domain from a corpus to filter relevant content, the application of SMOTE oversampling for the class of interest would have a positive effect on the results of this classification task.

The machine learning experiments' results with *Random Forest* and *AdaBoost* with *Sample* showed slightly improved overall performance and accuracy (62.84%) compared to those of *Random Forest* and *AdaBoost* with imbalanced data, while surpassing the achieved F1 score per class, except from class B. Comparing to the results obtained with SMOTE oversampling, undersampling has worse performance in terms of accuracy and class precision, recall and F1 score. The machine learning experiments' results with *Random Forest* and *AdaBoost* with *Sample* and *SMOTE* oversampling (hybrid approach) showed marginally improved overall performance and accuracy (63.35%) compared to those of *Random Forest* and *AdaBoost* only with *Sample*, while surpassing the achieved F1 score for classes A and D. However, the performance of this experiment is better than that of the experiment with completely imbalanced data. In conclusion, these experiments indicate that when applying both data undersampling and oversampling in a hybrid approach, the results are better than only applying undersampling, but worse than only applying oversampling for this data set. The findings derived from the machine learning experiments of this paper are in accordance with those of the relevant literature [12,17], in terms that data oversampling obtains better results than data undersampling in imbalanced data sets, while hybrid approaches perform reasonably well.

Potential directions for future work include the automatic extraction of domain-specific terminology, to be used as a component of an educational tool for sublanguage learning regarding specific vocational domains in host countries, aimed to help displaced communities, like migrants and refugees, overcome the language barrier. This terminology extraction

task could use the terms (features) that were identified in this paper as the most dominant for vocational domain identification. Moreover, a more vocational domain-specific data set could be created to perform a more specialized domain identification task in vocational subdomains, especially considering the set of terms identified in this paper. Experiments with a data set consisting of either more Wikipedia articles or textual data from other wikis and social networks as data sources could be performed. Using a different feature set, e.g. with n-grams and term collocations, could also be attempted. Finally, another potential direction for future work could be the application of the novel methodology of successive SMOTE oversampling proposed in this paper in combination with undersampling techniques and/or on other imbalanced data sets, in order to test its performance in different class imbalance problems.

Author Contributions: Conceptualization, K.L.K. and A.P.; methodology, M.N.N.; experiments, M.N.N.; validation, M.N.N.; formal analysis, M.N.N.; investigation, M.N.N.; resources, K.D.; data curation, K.D.; writing—original draft preparation, M.N.N.; writing—review and editing, K.L.K. and A.P.; visualization, M.N.N.; supervision, K.L.K. and A.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available in a publicly accessible repository that does not issue DOIs. Publicly available datasets were analyzed in this study. This data can be found here: https://hilab.di.ionio.gr/wp-content/uploads/2023/04/Wikipedia-Articles-for-Vocational-Domain-Identification.xlsx.

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

Abbreviations

The following abbreviations are used in this manuscript:

AUC	Area Under the Receiver Operating Characteristic curve
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
CRF	Conditional Random Field
MLP	Multi-Layered Perceptron
NLP	Natural Language Processing
ROC	Receiver Operating Characteristic curve
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency
WP	Wrong Prediction

666

References

1. Kosny, A.; Yanar, B.; Begum, M.; Al-Khooly, D.; Premji, S.; Lay, M.A.; Smith, P.M. Safe employment integration of recent immigrants and refugees. *Journal of International Migration and Integration* **2020**, *21*, 807–827.
2. Opute, J.; Hack-Polay, D.; Rigby, M. The employment situation of migrant workers and their experience of work–life pressures. In *Migration Practice as Creative Practice*; Emerald Publishing Limited, 2021.
3. Gürlek, M. Workplace ostracism, Syrian migrant workers' counterproductive work behaviors, and acculturation: Evidence from Turkey. *Journal of Hospitality and Tourism Management* **2021**, *46*, 336–346.
4. Cross, C.; Turner, T. Integration or exclusion? Assimilation of non-Irish nationals into the Irish labour market. *Journal of Contemporary European Studies* **2022**, pp. 1–13.
5. Kreyenfeld, M.; Diehl, C.; Kroh, M.; Giesecke, J. Female employment and migration in European countries: Introduction to the Special Issue. *Journal of Family Research (JFR)* **2021**, *33*, 230–251.
6. Turner, T. The jobs immigrants do: issues of displacement and marginalisation in the Irish labour market. *Work, employment and society* **2010**, *24*, 318–336.
7. Daunfeldt, S.O.; Johansson, D.; Seerar Westerberg, H. Which firms provide jobs for unemployed non-Western immigrants? *The Service Industries Journal* **2019**, *39*, 762–778.

8. Hall, M.; Greenman, E. The occupational cost of being illegal in the United States: Legal status, job hazards, and compensating differentials. *International Migration Review* **2015**, *49*, 406–442.
9. Basten, C.; Siegenthaler, M. Do immigrants take or create residents' jobs? Evidence from free movement of workers in Switzerland. *The Scandinavian Journal of Economics* **2019**, *121*, 994–1019.
10. Lange, D.; Böhm, C.; Naumann, F. Extracting structured information from Wikipedia articles to populate infoboxes. In Proceedings of the Proceedings of the 19th ACM international conference on Information and knowledge management, 2010, pp. 1661–1664.
11. Hardik, V.; Anirudh, V.; Balaji, P. Link analysis of Wikipedia documents using mapreduce. In Proceedings of the 2015 IEEE International Conference on Information Reuse and Integration. IEEE, 2015, pp. 582–588.
12. López, V.; Fernández, A.; García, S.; Palade, V.; Herrera, F. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information sciences* **2013**, *250*, 113–141.
13. Lin, W.C.; Tsai, C.F.; Hu, Y.H.; Jhang, J.S. Clustering-based undersampling in class-imbalanced data. *Information Sciences* **2017**, *409*, 17–26.
14. Anand, A.; Pugalenthi, G.; Fogel, G.B.; Suganthan, P. An approach for classification of highly imbalanced data using weighting and undersampling. *Amino acids* **2010**, *39*, 1385–1391.
15. Yen, S.J.; Lee, Y.S. Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. In Proceedings of the Intelligent Control and Automation: International Conference on Intelligent Computing, ICIC 2006 Kunming, China, August 16–19, 2006. Springer, 2006, pp. 731–740.
16. García, S.; Herrera, F. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary computation* **2009**, *17*, 275–306.
17. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* **2002**, *16*, 321–357.
18. Shelke, M.S.; Deshmukh, P.R.; Shandilya, V.K. A review on imbalanced data handling using undersampling and oversampling technique. *Int. J. Recent Trends Eng. Res* **2017**, *3*, 444–449.
19. Nikiforos, M.N.; Deliveri, K.; Kermanidis, K.L.; Pateli, A. Machine Learning on Wikipedia Text for the Automatic Identification of Vocational Domains of Significance for Displaced Communities. In Proceedings of the 2022 17th International Workshop on Semantic and Social Media Adaptation & Personalization (SMAP). IEEE, 2022, pp. 1–6.
20. Hamza, S.A.; Tahir, B.; Mehmood, M.A. Domain identification of urdu news text. In Proceedings of the 2019 22nd International Multitopic Conference (INMIC). IEEE, 2019, pp. 1–7.
21. Balouchzahi, F.; Shashirekha, H.L.; Sidorov, G. MUCIC at CheckThat! 2021: FaDo-Fake News Detection and Domain Identification using Transformers Ensembling. In Proceedings of the CLEF (Working Notes), 2021, pp. 455–464.
22. Hande, A.; Puranik, K.; Priyadharshini, R.; Chakravarthi, B.R. Domain identification of scientific articles using transfer learning and ensembles. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 2021, pp. 88–97.
23. Dowlagar, S.; Mamidi, R. Multilingual Pre-Trained Transformers and Convolutional NN Classification Models for Technical Domain Identification. In Proceedings of the Proceedings of the 17th International Conference on Natural Language Processing (ICON): TechDOfication 2020 Shared Task, 2020, pp. 16–20.
24. Gundapu, S.; Mamidi, R. Multichannel LSTM-CNN for Telugu Technical Domain Identification. In Proceedings of the 17th International Conference on Natural Language Processing, 2020, p. 11.
25. Lalithsena, S.; Hitzler, P.; Sheth, A.; Jain, P. Automatic domain identification for linked open data. In Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT). IEEE, 2013, Vol. 1, pp. 205–212.
26. Nakatani, M.; Jatowt, A.; Ohshima, H.; Tanaka, K. Quality evaluation of search results by typicality and speciality of terms extracted from wikipedia. In Proceedings of the International Conference on Database Systems for Advanced Applications. Springer, 2009, pp. 570–584.
27. Saxena, K.; Singh, T.; Patil, A.; Sunkle, S.; Kulkarni, V. Leveraging Wikipedia navigational templates for curating domain-specific fuzzy conceptual bases. In Proceedings of the Proceedings of the Second Workshop on Data Science with Human in the Loop: Language Advances, 2021, pp. 1–7.
28. Stoica, A.S.; Heras, S.; Palanca, J.; Julián, V.; Mihaescu, M.C. Classification of educational videos by using a semi-supervised learning method on transcripts and keywords. *Neurocomputing* **2021**, *456*, 637–647.
29. Thomas, D.M.; Mathur, S. Data analysis by web scraping using python. In Proceedings of the 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2019, pp. 450–454.
30. Kumar, S.; Zymbler, M. A machine learning approach to analyze customer satisfaction from airline tweets. *Journal of Big Data* **2019**, *6*, 1–16.