THE 18TH EDITION OF THE INTERNATIONAL CONFERENCE
EUROPEAN INTEGRATION
REALITIES AND PERSPECTIVES

The European Citizen and
Public Administration

Text Codification for Statistical Production using Machine Learning

Helda Curma[1], Valentina Sinaj[2]

Abstract: **Objective** The main objective of this research is the usage and evaluation of Machine Learning algorithms for automatic text codification in statistical production process. **Prior Work** This is an evolving area of research due to rapid changes in technology as well as the new data ecosystem. The paper will build on previous research done on text classifications techniques. **Approach** In this paper Machine Learning algorithms will be used and evaluated for text codification. Natural Language Processing and classification algorithms will be implemented in Python. **Results** Machine Learning is powerful in the process of automation and modernization of statistical production lifecycle. Machine Learning algorithms performance is different for text classification. Data pre-processing and balance on the training data set are important to achieve good results. **Implications** This study shows that machine learning can be used in automating part of the statistical codification process. The results of this paper will serve the work of Albanian administration and more specifically statistical production. **Value** This research is a contribution to the usage of Machine Learning for the modernization of the codification process. It will serve as an initial work towards improving the timeliness and lowering statistical production costs

Keywords: Algorithms; Data; Classification

JEL Classification: C88

1. Introduction

The digitization of numerous contemporary sectors relies heavily on text classification. Text classification, which is also sometimes referred to as text tagging or text categorization, is the process of putting text into distinct, well-organized groups by assigning labels or classes. Many operations, including survey analysis, sentiment analysis, customer service, document summarizing, and more, can be automated with the use of text classifications.

In the statistical production process open-text replies, that need to be classified, are involved. Traditionally this process has been done by professional people (codifiers), making the process difficult, slow, and dependent on the knowledge of these individuals. This process is difficult sometimes also due to the very short or incomprehensible answers.

[1] PhD in progress, University of Tirana, Albania, Address: Place, "Mother Tereza" Tirana, Albania, Corresponding author: hmitre@instat.gov.al.
[2] Professor, University of Tirana, Albania, Address: Place, "Mother Tereza" Tirana, Albania, E-mail: sinajv@yahoo.com.

Over the years, researchers have used a variety of rule-based coding techniques, in an effort to partially automate coding. For instance, the relevant code in the dictionary was allocated if the text response contained a term that matched an entry in a prepared dictionary.

More recently Machine Learning approaches can be used to facilitate this process while limiting the need for human intervention. The capacity to scale and precisely extracts particular information from vast amount of textual data is the most major benefit of text categorization via Natural Language Processing (NLP). A variety of business processes can be automated by using high-quality classifiers and encoders, making the procedure quick and efficient.

There are many classification schemes involved in official statistics production process. The codification using these schemes is challenging due to the large number of codes which are also nested hierarchically.

In this paper will be explored the possibility to use Machine Learning for automatic codification of economic activities of enterprises, based on the NACE Rev.2 classification (Eurostat, 2008). This classification consists of a five level hierarchic codes, with more than six hundred codes on the more detailed level of codification (four-digit level). Table 1 presents the NACE Rev.2 classification for Group 01.1 covering the economic activities dealing with growing of non-perennial crops.

Table 1. NACE Rev.2 Classification, Group 01.1: Growing of non-perennial crops

Division	Group	Class	Description	ISIC Rev.4
			SECTION A — AGRICULTURE, FORESTRY AND FISHING	
01			Crop and animal production, hunting and related service activities	
	01.1		Growing of non-perennial crops	
		01.11	Growing of cereals (except rice), leguminous crops and oil seeds	0111
		01.12	Growing of rice	0112
		01.13	Growing of vegetables and melons, roots and tubers	0113
		01.14	Growing of sugar cane	0114
		01.15	Growing of tobacco	0115
		01.16	Growing of fibre crops	0116
		01.19	Growing of other non-perennial crops	0119

Natural Language Processing techniques and three Machine Learning algorithms will be used on open ended replies on economic activity. Python will be used as a programming language as it is most appropriate to deal with machine learning for automating these processes. It is more straightforward and consistent than other programming languages due to its simplicity, flexibility, and powerful libraries.

This article is organised as follows: Section 2 will give background information on automatic text codification through text pre-processing and machine learning algorithms. In Section 3 the solution approach will be explained by giving more information on the input data and machine learning classifiers used. Section 4 will present the main results of this work, while Section 5 will summarise the main conclusions of this paper.

2. Automatic Text Codification

2.1. Text Pre-Processing

The initial step in text codification is data preparation, which entails preparing the raw data for future analysis by cleaning and modifying it. Stop-word elimination, stemming, lemmatization, and the elimination of special characters and punctuation are tasks that fall under this step. These processes are required to lower the data's dimensionality and get rid of noise and redundancy.

Unlike humans, machines cannot understand free text. Unstructured text needs to be cleaned in order to be useful by any machine learning algorithm. According to Loper at al 2008, pre-processing can enhance the quality of datasets in general and for text classification in particular. The pre-processing step can "clean" the dataset of "noise" (for example, by fixing spelling mistakes, cutting down on duplicate characters, and disambiguating acronyms). In some circumstances, the dataset's quality for text classification tasks can be enhanced by applying pre-processing techniques such stop-word removal, punctuation mark removal, word stemming, and word lemmatization.

2.2. Machine Learning Algorithms for text codification

Text classification has been gaining power due to developments in the fields of text mining and natural language processing (NLP) and some practical applications go beyond simple task of categorisation/classification into summarisation and evaluation of open answers to specific questions (Gasparetto et al. 2022). Whenever labelled data are available, the best technique to deal with text codification is the supervised learning one.

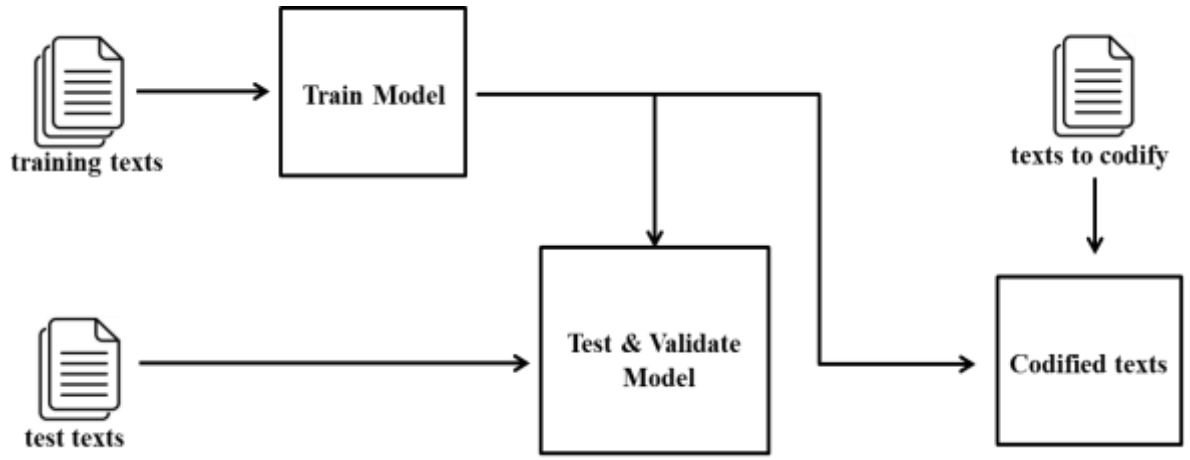

Figure 1. Automatic TEXT CODIFICATION PROCESS

Figure 1 presents the automatic text codification process, starting form splitting the labelled data into training and test texts until classifying new text based on the classification model built.

3. Solution Approach

In this section will be explained the approach used to solve the problem of automatic codification for NACE classification, based on labeled data from free-text replies.

3.1. Input Data

Input data is prepared from previously conducted surveys. The data set contains approximately 40,000 records. As explained also in Figure 1 the data set is divided in two parts, one for training and another for testing. The scenario used to divide the dataset was 70% of the rows to be used for training the model, while 30% for testing it. This dataset contains two columns: Activity description based on free text replies and NACE Rev.2 code. The initial dataset was larger, but not balanced; there were many economic activities with very few records. In this paper, to evaluate the potential of using machine learning for automatic text codification, these cases were not taken into consideration. More than 70 different codes for economic activities are part of the dataset used for the classification process. In Table 2 is presented the structure of the dataset used.

Table 2. Structure and example of the dataset used

Activity description	NACE Code
chrome production	0729
EXTRACTION AND UTILIZATION OF CHROMIUM	0729
STONE WORKS	2370

Automatic text codification includes Natural Language Processing (NLP) as the free text needs to be pre-processed before training a machine learning model. Machine learning models cannot use raw text directly. In this paper, NLTK is used for text pre-processing. NLTK, the Natural Language Toolkit is a suite of open source program modules, tutorials, and problem sets, providing ready-to-use computational linguistics courseware; it covers symbolic and statistical natural language processing (Bird et al. 2009). During the pre-processing phase, blank rows are removed from the dataset. All the data is transformed in lowercase as Python is case-sensitive. Afterward, word tokenization is performed on the dataset to split paragraphs and sentences into words. Lemmatization is used as it performs morphological analysis of the words by combining several inflected forms into a single unit for analysis. Scikit-learn library is used as it provides state-of-the-art implementations of many well-known machine learning algorithms (Pedregosa 2011). To quantify the importance of words in a text reply amongst a collection of text replies, Term Frequency-Inverse Document Frequency (TF-IDF) is used.

3.2. Machine Learning Classifiers

Three different classifiers are used to train and test the model: Naive Bayes, Support Vector Machine and Random Forest. The three of them are based on supervised machine learning techniques and can be used on our dataset as we already have labelled data that can be used to train and test the model. The dataset is partitioned in training and testing subsets. The same TF-IDF matrix architecture is used for the different classifiers.

3.2.1. Naive Bayes

Classification algorithms that are built on the Bayes' Theorem are known as naive Bayes classifiers. These algorithms are all based on the idea that every pair of features being classified is independent of the other. Machine learning tasks involving text codification and text analysis have made extensive use of naive Bayes classifiers (Albon 2018). Based on a hypothesis defined for our data, the Bayes' Theorem will calculate the probability that the hypothesis will occur by multiplying the probable chances. This way the hypothesis will occur true given certain scenarios. After this the product will be divided by the probability that the defined scenario will occur. As we are codifying texts, the hypothesis is that the text belongs to Category C. The evidence is the occurrence of the word W. The texts that need to be codified contain many words and the formula is presented as follows:

Equation 1: Formula for naïve Bayes classifiers

$$P(C|T) = P(C) * \frac{P(T|C)}{P(T)}$$

In this formula:

- P(C|T) is the probability of Category C given text T
- P(C) is the prior probability of Category C
- P(T|C) is the probability of text T given Category C. It is calculated as the product of each word in the text T given Category C.
- P(T) is the probability of text T.

P(T|C) is calculated as the product of each word in the text T given Category C, while P(T) normalisation factor makes sure that all possible class probabilities total is 1.

A new text is categorised by calculating the probability of the text belonging to each category, and the category with the highest probability is chosen.

3.2.2. Support Vector Machine

Support Vector Machine (SVM) is a machine learning algorithm that can be used for classification problems. SVM use linear models to implement nonlinear boundaries for the different categories (Berry et al. 2010). This is accomplished by using nonlinear mappings to convert a given instance space into a linearly separable one. To maximize the separation between training samples for two categories, SVM builds a separating hyperplane based on so-called support vectors. This transformation is generally achieved by using a kernel function that is in charge of mapping input features to a higher-dimensional space. The basic formula for Support Vector Machines for text classification is as follows:

Equation 2 Formula for Support Vector Machines for text classification

$$\boldsymbol{f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x_i, x_{i'})}$$

In this formula as explained by Albon, 2018:

- β_0 is the bias
- S is the set of all support vector observations
- α are the model parameters to be learned

- (x_i , $x_{i'}$) are pairs of two support vector observations, x_i and $x_{i'}$
- K is a kernel function that compares the similarity between x_i and $x_{i'}$

In text categorisation, the input features may be the word frequencies or presence of specific words in a text. The kernel function calculates the similarity between the texts based on the word features and maps them to a higher-dimensional space where the texts can be separated by a hyperplane. SVM target is to find a hyperplane that maximizes the distance between two categories.

3.2.3. Random Forest

Random Forest is a supervised machine learning algorithm that can be used for classification and regression problems. Random Forest for text classification is based on ensemble learning, which combines different models to improve the reliability and accuracy. The Random Forest is composed of Decision Trees. In order to create every Decision Tree, from the training data is used a random subset of records and also a random subset of features. In this way over fitting is reduced and the model in general is improved. Every Decision Tree is trained separately while the final prediction is made by aggregating the predictions of all the decision. In summary, the theory behind Random Forest for text classification is based on the concept of ensemble learning using decision trees (Fürnkranz, 2011). In the case of text classification every decision tree is trained to predict the class of a text reply based on a subset of features of the text.

4. Results

In order to evaluate the performance for the three different classifiers four different metrics are used. Accuracy, Precision, Recall and F1 score are calculated for each of the models. The results are presented in the Table 3. As expected, results vary considerably among classification algorithms. As we can observe Naïve Bayes has a lower performance than the other two classifiers. Random Forest has similar performance with Support Vector Machine but still, Support Vector Machine scores are better. Support Vector Machine overcomes in all four-evaluation metrics.

Table 3. Results of the Three Different Classifiers

Algorithm	Naive Bayes	SVM	RF
Accuracy Score (%)	70.71	77.99	77.243
Precision Score (%)	59.59	72.35	71.18
Recall Score (%)	76.40	77.81	74.71
F1 Score (%)	62.01	73.69	72.31

This is also shown on confusion matrixes produced for the three models, part of which is presented in Figure 2.

```
Confusion Matrix for Naive Baye:
[[ 96   0   0 ...   0   0   0]
 [  0  59   0 ...   0   0   0]
 [  0   0  85 ...   0   0   0]
 ...
 [  0   0   0 ... 209   0   1]
 [  0   0   0 ...   5  41   4]
 [  0   0   0 ...   0   0 477]]
```

```
Confusion Matrix for SVM
[[ 98   0   0 ...   0   0   0]
 [  0  60   0 ...   0   0   0]
 [  0   0  88 ...   0   0   0]
 ...
 [  0   0   0 ... 211   0   0]
 [  0   0   0 ...   1  44   6]
 [  0   0   0 ...   0   0 474]]
```

```
Confusion Matrix for RF
[[ 94   0   0 ...   0   0   0]
 [  0  61   0 ...   0   0   0]
 [  0   0  86 ...   0   0   0]
 ...
 [  0   0   0 ... 210   0   1]
 [  0   0   0 ...   3  43   3]
 [  0   0   0 ...   0   0 478]]
```

Figure 2. Part of Confusion Matrixes Generated for the Three Models

Performance of the models is evaluated more in details to understand what codes are classified correctly or incorrectly by the models. Confusion matrixes results highlight also the problems that may arise as a consequence of using a not well-balanced data set.

5. Conclusions

This work is a proof of concept that machine learning can be used for automatic text codification in statistical production, on codification of open ended text replies, for economic activities. The results of it allow identifying potential improvements to the current manual codification processes, which if implemented in the future will have an impact on decreasing the workload of codification employees and making the process less dependent on human mistakes. However, the results may not be ideal for several reasons such as the input data used, the selection of the vectorizer, or the classifiers used. More test hypotheses need to be generated in order to improve different model performance. The data cleaning and pre-processing shall be focused also in balancing the labelled dataset, which will have an impact on the overall performance. The model performance shall be evaluated also while executing hyper-parameter tuning. This study shows that for the data that were available, SVM with TF-IDF performs better, and need to be explored more on how to improve performance scores, in order to be used in the future in the statistical production chain. Ensemble techniques combining different models may be considered and tested to achieve a better performance.

References

Albon, C. (2018). *Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning.* Location: Japan: O'Reilly Media.

Berry, M. W. & Kogan, J. (Eds.). (2010). *Text mining: applications and theory*. John Wiley & Sons.

Bird, Steven, Edward Loper and Ewan Klein (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.

Eurostat, European Commission (2008), Methodologies and working papers title. *NACE Rev. 2 Statistical classification of economic activities in the European Community*, Retrieved from https://ec.europa.eu/eurostat/documents/3859598/5902521/KS-RA-07-015-EN.PDF.

Fürnkranz, J. (2011). Decision Tree. In: Sammut, C., Webb, G.I. (eds). *Encyclopedia of Machine Learning.* Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_204.

Gasparetto, A.; Marcuzzo, M.; Zangari, A. & Albarelli, A. (2022). *A Survey on Text Classification Algorithms: From Text to Predictions*. Information, 13(2), p. 83.

Loper, E.; Bird, S.; Klein, E. & Loper, E. (2009). *Natural language processing with python.* Germany: O'Reilly Media, Incorporated.

Pedregosa, F.; Varoquaux, Gael; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O. et.all. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.