Keyword Categorization using Statistical Methods

Dominika Krasňanská , Silvia Komara, Mária Vojtková

University of Economics in Bratislava, Faculty of Economic Informatics, Dolnozemska cesta 1b, Bratislava, Slovakia

Abstract **– Keyword analysis is a way to gain insight into market behaviour. It is a detailed analysis of words and phrases that are relevant to the selected area. Keyword analysis should be the first step in any search engine optimization, as it reveals what keywords users enter into search engines when searching the Internet. The keyword categorization process takes up almost half of the total analysis time, as it is not automated. There is currently no known tool in the online advertising market that facilitates keyword categorization. The main goal of this paper is to streamline the process of keyword analysis using selected statistical methods of machine learning applied in the categorization of a specific example.**

Keywords **– Keyword analysis, Machine learning, Python programming language, Linear support vector classifier.**

1. Introduction

Web analytics is one of the youngest areas of data analytics, and it has been developing since 1995. This activity is aimed at monitoring selected indicators of the website to improve the website.

Avinash Kaushik [7], one of the most influential figures in the field of web analytics, has defined web analytics as the analysis of quantitative and qualitative data from the company's website and competitors, to continuously improve the user experience of current and potential customers, which is later reflected in the form of desirable business results in the offline and online environments.

One of the areas of web analytics is, among other things, Search Engine Optimization (SEO) [9]. It is a set of techniques that optimize websites in order to penetrate the first positions in organic (unpaid) search results in search engines [15].

The essence of website optimization is to identify the keywords that lead to increased traffic to a particular site. Many online marketing companies are currently dealing with the development of keyword analysis. The process of analysing keywords is well known [8]. It consists of four main steps: the collection of keywords, their cleaning, the categorization of keywords, and the subsequent interpretation of the results flowing from the analysis. Developing keyword analysis is time-consuming. Each analysis is specific, and the time spent processing it is individual depending on the area that is analysed. The standard time for performing a keyword analysis begins at 50 hours, in some cases goes up to 120 hours.

Categorization requires the most time in keyword analysis. The categorization process takes up approximately half of the total analysis time, as this process is not automated. There is currently no known tool in the online advertising market that facilitates keyword categorization. For that very reason, we decided to use statistical methods designed to categorize keywords in the paper [10], which will result in streamlining and saving time on the analysis in practice.

We used an Excel spreadsheet from Microsoft and several online marketing tools such as Search Console, Google Analytics, Marketing Miner, Ahrefs, Google Ads, Collabim, and various whisperers such as Google Keyword Planner or Ubersuggest's free keyword tool to process and analyze keywords. From the various possible statistical methods, we applied selected machine learning methods in the Python programming language in the integrated development environment of Jupyter, using several libraries.

DOI: 10.18421/TEM103-47
https://doi.org/10.18421/TEM103-47

Corresponding author: Maria Vojtkova,
University of Economics in Bratislava, Faculty of Economic Informatics, Slovakia.
Email: maria.vojtkova@euba.sk

Received: 25 May 2021.
Revised: 29 July 2021.
Accepted: 06 August 2021.
Published: 27 August 2021.

2. Machine Learning

Machine learning is a sub-area of artificial intelligence dealing with the methods and algorithms that allow a program to learn and then respond adequately to various input values without being explicitly programmed for these tasks. The program works only based on information that has been learned. Machine learning algorithms use elements of mathematical statistics, methods of statistical analysis, and data mining [2], [11].

The learning process begins with observation and the examination of data, perhaps gaining experience. The program will subsequently find patterns in them and, based on them, improve its decisions in the future. The main intention is for the computer to learn independently, without human assistance, and to be able to apply what it has learned in practice. For this reason, machine learning takes place in two phases. The first of these is the learning phase, also called the training phase, during which a mathematical model is created, which we train for a selected task on a training dataset. Afterward, in the second phase, we deploy the trained model in a production environment. Based on how the specific learning process takes place, machine learning algorithms can be divided into three groups [11]:

1. Supervised machine learning – this group includes most of the currently used machine learning algorithms: the support vector machine (SVM) method, neural networks (sometimes referred to as deep machine learning), the Bayesian classifier, and decision trees.
2. Unsupervised machine learning – used for data that has not been pre-classified: clustering and association.
3. Reinforcement learning – it is a particular category of machine learning algorithms: Q-learning algorithm or Deep Q-learning.

The Python programming language uses eight basic machine learning algorithms. In this paper, we decided to use the support vector machine algorithm to categorize keywords [1], [14] and compare it with the algorithms of logistic regression and Naive Bayes [6].

3. The Process of Keyword Analysis with a Specific Example

Keyword Collection

The first step in keyword analysis is to gather keywords. The area of analysis depends on the focus of the offered assortment of the given customer. In our case, we decided to analyse the area of jewellery for an anonymous customer. The selected area is enormous, so the keyword analysis took us more than 100 hours. When collecting keywords, we used various online marketing tools, e.g., Google Search Console, Google Analytics, Marketing Miner, Ahrefs, Google Ads, Collabim, various whisperers like Google Keyword Planner or Ubersuggest's free keyword tool, the website of the customer for whom the analysis was performed, competitor websites, discussion forums, social networks, and several other resources that focus on the field of jewellery.

From all the sources mentioned above, we collected 112,117 keywords at the beginning of the keyword analysis.

Keyword Cleanup

The keyword cleanup consisted of several stages. In the first phase, duplicate keywords had to be removed. We removed duplicate keywords in the MS Excel spreadsheet, using conditional formatting with the option to mark duplicate words, which we subsequently removed. Duplicates accounted for more than 50% of the keywords collected.

In the next phase of the keyword cleanup, it was essential to find the average monthly search for each keyword. We took this step using the online marketing tool Marketing Miner, which assigned each keyword its average monthly search. We then removed the zero-search keywords. After deleting keywords with zero average monthly searches, we had 24,917 keywords available.

The next step was to remove irrelevant keywords, i.e., words unrelated to the field of jewellery. This was a time-consuming task because all 24,917 terms had to be crawled, and irrelevant keywords had to be deleted. After removing the irrelevant keywords, we had 8,731 words left for further analysis.

The last task was to eliminate the problem with diacritical signs and multi-word expressions regarding the different ways people enter the terms they want to find into the search engine. We solved the problem with diacritics by creating a macro in the MS Excel spreadsheet, whose task was to remove the diacritical marks from all the keywords. We then assigned the average monthly search as the sum of the average monthly searches for the keyword or phrase we selected as the correct one.

After removing all the issues that needed to be resolved, we obtained 5,549 keywords, from which we decided to create categories.

Keyword Categorization

The goal of categorization was to sort the keywords into categories. Keywords within one category should be as similar as possible. The categorization process itself was manual. We went word for word and assigned them to subjectively created categories. We will explain the procedure in

one of the categories called *Product*, which contains 33 subcategories. Each subcategory included several tens or thousands of keywords. For keywords within each subcategory, we assigned an average monthly search, which we then sorted, and based on them, we created individual subcategories manually, i.e., from the keywords with the highest average monthly searches to the keywords with lower average monthly searches.

4. Keyword Categorization Using Machine Learning Methods

In the commercial world, there are many applications for text classification. However, there are only binary text classifications in most cases, such as filtering e-mails into spam (yes, no) or behavioral analysis (positive vs. negative). Our task was to categorize keywords into several dozen categories. One solution could be tools for automatic document classification based on artificial intelligence (AI). The main intention is for the computer to be able to learn independently, without human assistance, and to be able to apply in practice what it has learned. For categorizing keywords, in our case, we used an empirical approach based on the principle of learning and testing, allowing for the selection of the optimal model for a given input set, i.e., machine learning with a teacher. We implemented machine learning methods in the Python programming language, specifically in the Jupyter development environment. The input consisted of 5549 keywords. The process of collecting, cleaning the duplicates, the keywords with zero average monthly searches, and irrelevant keywords was described in the previous sections of the article. For the selected category *Product*, the subcategories are listed in Table 1.

Table 1. List of subcategories of the Product variable

Category	Subcategories within the relevant category	Number of Subcategories
Product	Watches, earrings, bracelets, rings, jewelry, chains, rosary, wedding bands, pendants, neckleces, piercing, keychain, stone, sets, brooch, beads, cufflinks, accesories, jewellery boxes, components, brick, medallions, scapular, clips, cups, tie clips, pens, plug, ring bearer pillow, organizer, collection, amulet, crystal	33

After importing the file, it was necessary to import the libraries we worked with, namely those from Numpy, Matplotlib, Pandas, String, Seaborn, and Scikit-learn, also known as Sklearn.

In our case, the input (x) was represented by the *Keyword* variable, and the output variable (y) was *Product*, more precisely, individual subcategories within this category, of which there were a total of 33. We removed keywords that were not in any of the subcategories. Subsequently, it was necessary to transform the categorical variables within the *Product* variable into numerical variables. We performed the transformation by creating a new column called *Category_id* with the character of a numeric variable applying several commands, more precisely, a string of commands using the String library. At present, computers can already work very well with numeric variables, but not that great with text data. One of the most commonly used techniques for processing text data is a technique called Term Frequency - Inverse Document Frequency, abbreviated TF-IDF, so-called vectorization of documents [16].

The TF-IDF technique consists of two indicators. The first of them is TF (the number of documents or words), and the second indicator represents IDF (the number of inverse documents). T represents an expression (word) and, D defines a document which, in our case is a keyword. N is a corpus, which represents the whole set of documents, i.e., in our case, all keywords. In this way, we can determine the number of words in a document, that is, the number of words within a keyword [3].

However, the frequency is also affected by the length of the document, i.e., the length of the keyword. For example, a widespread word like "is" (meaning "to be") may appear more than once in a document. However, if we consider two documents, one with a word count of 100 and the other with a word count of 10,000, there is a high probability that the word "is" will appear more often in a longer document than in a shorter one. However, based on this assumption, we cannot claim that a longer document is more important than a shorter document. It is for this reason that we perform the so-called normalization of the values of the obtained frequencies. Our goal was to vectorize documents, i.e., transform the text into numeric variables. If we perform a text transformation (vectorization of documents), we cannot only consider the words in the relevant document. If we do this, then the length of the vector will be different for both documents, and it will not be possible to calculate the similarity. We have to do this by vectorizing the documents in the dictionary. The dictionary is a list of all possible words in the corpus. After vectorizing, we check the number of words. If the word does not exist in the

document, then the value of TF is 0. If all the words in the document were the same, the value of the indicator TF would be 1. The final value of the normalized indicator TF is, therefore, in the interval [0,1]. It follows from the above description that the value of the TF indicator is individual for each document.

TF $(tf_{t,d})$ indicates the frequency of the word (t) within the document (d), and in our case, within the keyword. This is the ratio of the number of occurrences of the word in the keyword $(n_{t,d})$ compared to the total number of words in the keyword ($\sum n_{t,d}$). Each keyword has its own TF $(tf_{t,d})$, which is given by the relationship:

$$tf_{t,d} = \frac{n_{t,d}}{\sum n_{t,d}} \quad (1)$$

Within keywords, these can be various prepositions, conjunctions, or expressions that are not important in the analysis. These words are called stopwords, i.e., words that often appear in keywords with almost no meaning. However, by finding these words in the corpus and normalizing them, we will achieve much better results.

In connection with the transformation, i.e., by vectorizing keywords, defining the document frequency (DF) is necessary. The document frequency (df_t) represents the number of documents (n_t), i.e., keywords in which the selected word is present. The relationship for calculating DF is as follows:

$$df_t = \frac{n_t}{\sum n_{t,d}} \quad (2)$$

Our main goal was to get the so-called informativeness, i.e., the determination of expression weights. Then the Inverse Document Frequency (IDF) indicator is used. IDF is the inverse of a document's frequency that determines the informativeness of a word. If we calculate the IDF value, then for the most common words, such as stopwords, this value will be very low. Words that rarely occur in the analysis will have a higher IDF value than words that occur more often.

We calculate the inverse value of the document number as follows:

$$idf(t) = (\frac{N}{df_t}), \quad (3)$$

where

$idf(t)$ is the inverse value of the document frequency *(t),*

N is the corpus, i.e., the total number of documents

df_t is the number of documents with expression *(t).*

In the case of huge documents, a problem with the IDF indicator can occur. For this reason, the IDF indicator must be logarithmic:

$$idf(t) = \ln(\frac{N}{df_t + 1}) \quad (4)$$

Suppose a word in the database is not in the dictionary or corpus; its DF value will be 0. Since we cannot divide by zero, we will add 1 to the denominator for this reason.

Subsequently, we transformed a text document using the *CountVectorizer* function in Python to the so-called token matrix. The result was a token matrix that contained 3722 rows and 2061 columns. By applying a token matrix, we can express every single keyword through a numeric value (Figure 1.).

Each keyword represents a numeric value. For example, the first line in Figure 1. defines the first term found in the first keyword in the numerical expression. The first keyword in the keyword analysis is the keyword "18-carat gold chains." This keyword is expressed in the first four lines of the output below. The first number in parentheses defines the keyword's ranking in the keyword analysis. The second number defines the column within the token matrix in which the expression or word can be found.

```
print(word_count)
```

(0, 11)	1	:	:
(0, 637)	1	(3713, 787)	1
(0, 1947)	1	(3713, 1980)	1
(0, 1433)	1	(3714, 2025)	1
(1, 11)	1	(3714, 1980)	1
(1, 632)	1	(3715, 1947)	1
(1, 1371)	1	(3715, 1433)	1
(2, 1433)	1	(3715, 125)	1
(2, 18)	1	(3715, 1987)	1
(2, 631)	1	(3716, 1095)	1
(2, 1943)	1	(3716, 2055)	1
(3, 30)	1	(3717, 1373)	1
(3, 1942)	1	(3717, 2055)	1
(3, 1427)	1	(3718, 2025)	1
(4, 42)	1	(3718, 2055)	1
(4, 1588)	1	(3719, 1486)	1
(5, 44)	1	(3719, 2057)	1
(5, 608)	1	(3720, 1095)	1
(5, 1373)	1	(3720, 2059)	1
(6, 46)	1	(3721, 1947)	1
(6, 2025)	1	(3721, 1433)	1
(7, 1588)	1	(3721, 280)	1
(7, 47)	1	(3721, 997)	1
(8, 40)	1	(3721, 733)	1
(8, 1032)	1	(3721, 546)	1
		(3721, 2058)	1

Figure 1. Token matrix display
Source: own processing

After this process, we can calculate the IDF value. The value of the IDF indicator is inversely proportional to the frequency of words in the keyword analysis. Once we know the IDF value for each word, we can calculate the total TF-IDF value for each word in the analysis:

$$tf - idf_{t,d} = tf_{t,d}\ x\ \ln(\frac{N}{df_t + 1}) \quad (5)$$

where

$tf_{t,d}$ is the number of occurences *t* in *d,*

df_t is the number of keywords in the corpus containing *t,*

N is the total number of keywords, i.e., overall corpus.

We calculated the TF-IDF values for each individual word, i.e., an expression found in a keyword. For example, for the first keyword "18 carat gold chain" in the keyword analysis, the TF-IDF value is shown in Figure 2.

	TF-IDF
18 carat	0.642620
gold	0.590395
chains	0.363749
	0.325821

Figure 2. TF-IDF values for the keyword "18 carat gold chains"
Source: own processing

The next step in the analysis was to divide the data into a training and validation set using the Pandas library, and in our case, we used a 2:1 ratio. We used the training set for learning and the validation set for evaluation.

After all the above transformations, we applied the three selected machine learning methods:

- Linear support vector classifier,
- Multinomial logistic regression,
- Multinomial Naive Bayes.

These are methods that are suitable for categorizing texts and are recommended by several authors who deal with this issue [8], [10].

Linear Support Vector Machine

In our case, the method of support vectors gave the best results; therefore, we will deal with it in more detail. The principle of this method is to find the optimal superstructure that maximizes the range between classes. The goal of the support vector method is to divide the data into output classes [1]. In our case, it is about splitting keywords into the appropriate subcategories within the *Product* category.

The support vector method is one of the machine learning algorithms that is used quite often today. This method represents a binary classifier, meaning that it only operates with a class 0 and 1 scenario. In other words, it is not possible to create a multi-class classification scenario. However, there are several methods that allow the usage of the support vector method in classifying multiple classes [4], [5]. One of these methods is the one-against-one method. When applying the method of support vectors in the classification of several classes, it is, therefore, necessary to divide the data into several files using the previously mentioned one-against-one method. When applying the one-against-one method, the classifier is trained for each pair of classes, which allows us to constantly compare. The principle of this method is to divide a data set with several variations into the binary classification form, while the number of classifiers is determined according to the following formula:

$$k = \frac{N(N-1)}{2}, \tag{6}$$

where

k is the number of classifiers,

N is the number of classes, i.e., the number of subcategories within the Product category.

In our case, the classification of keywords into any of the 33 subcategories exists according to the relationship of (6) 528 classifiers. In 528 binary problems, we determined the optimal superstructure that divided the data into one of the classes. The result of applying the model of linear support vectors was the inclusion of keywords into individual subcategories within the *Product* category.

We used various metrics to assess the success of the classification [12]. The "accuracy" metric is based on empiricism and does not distinguish between the number of correct predictions of different classes [13]. It is defined as:

$$accuracy = \frac{tp + tn}{tp + fp + fn + tn} \tag{7}$$

in the case of the classification of the classes "positive" and "negative", then:

tp represents the number of true positive cases,

tn represents the number of true negative cases,

fp represents the number of predicted positive cases,

fn represents the number of predicted negative cases.

Accuracy approximates the probability of the correct classification results.

In addition to the rate of "accuracy" in the field of text classification, information extraction, natural language processing, etc., we also use metrics such as "specificity" (precision) and metrics called "sensitivity" (recall), which are defined based on the relationships:

$$precision = \frac{tn}{tn + fp} \tag{8}$$

$$recall = \frac{tp}{tp + fn} \tag{9}$$

where precision estimates the predictive power of an algorithm and sensitivity estimates the efficiency of an algorithm, specifically a class. From the metrics

mentioned above, it is then possible to determine a metric called "*F*-score," which in the case of $\beta = 1$ calculates the balanced sets, if $\beta > 1$ prefers the precision, otherwise the recall. We calculate the *F*-score metric according to the relation:

$$F - score = \frac{(\beta^2 + 1) \,.\, accuracy \,.\, recall}{\beta^2 \,.\, accuracy + precision} \qquad (10)$$

The used model of linear support vectors classified the keywords into individual subcategories, in our case, very well. The accuracy rate shown in the figure below for the linear support vector model was 96.8294 %.

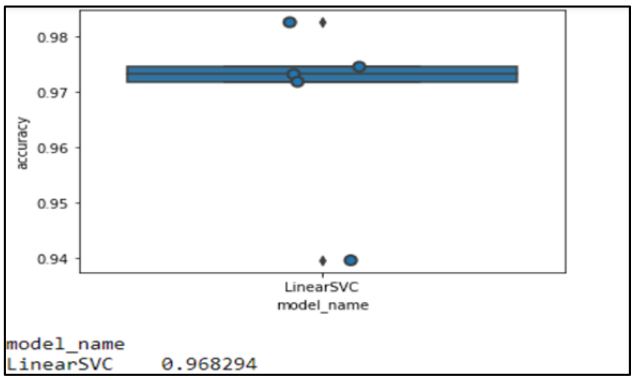

Figure 1. Box-plot, A visual view of the model accuracy rate
Source: own processing

Subsequently, we were interested in the accuracy of the keyword classification within the individual subcategories. We showed the degree of accuracy of classifying keywords into individual categories through the classification report shown in Figure 4.

There are measures expressing accuracy in the columns. More specifically, in the first column there is precision, in the second column there is recall, and in the third column there is a measure of the F1-score. The last column contains the number of keywords in each category within the validation set, which is used to evaluate the model. In the subcategories of cufflinks, rosary, and accessories, the model of linear support vectors included all the keywords correctly, i.e., into the categories in which they were originally located. Of course, the accuracy rates in each category are affected by the number of keywords within that subcategory. The overall accuracy rate for all the subcategories within the *Product* category reached a 96 % accuracy rate for precision and an accuracy rate for recall of 97 %.

	precision	recall	f1-score	support
chains	0.99	0.96	0.97	80
rings	0.98	0.99	0.99	170
jewelry	0.98	0.99	0.99	124
bracelets	0.98	0.97	0.98	145
watches	0.99	1.00	0.99	187
earings	0.96	0.96	0.96	162
cufflinks	1.00	1.00	1.00	9
neckleces	0.96	1.00	0.98	53
stone	0.96	1.00	0.98	26
crystal	0.00	0.00	0.00	1
pendants	0.99	1.00	0.99	73
wedding rings	0.92	0.97	0.94	35
sets	0.95	0.90	0.92	20
piercing	0.94	0.97	0.95	30
brooches	1.00	0.75	0.86	8
rosary	1.00	1.00	1.00	15
collection	0.00	0.00	0.00	2
beads	0.83	1.00	0.91	5
accessories	1.00	1.00	1.00	14
keychains	0.98	0.93	0.95	44
components	0.67	1.00	0.80	4
jewelry boxes	0.67	0.67	0.67	3
tie clips	0.00	0.00	0.00	1
medallions	0.60	0.75	0.67	4
amulet	0.00	0.00	0.00	1
organizer	0.00	0.00	0.00	1
plug	0.00	0.00	0.00	2
clips	0.00	0.00	0.00	0
pens	0.00	0.00	0.00	1
cups	0.25	1.00	0.40	1
scapular	1.00	0.67	0.80	3
ring pillow	0.00	0.00	0.00	1
brick	1.00	0.75	0.86	4
avg / total	0.96	0.97	0.97	1229

Figure 4. Classification report for individual categories within the Product category
Source: own processing

Subsequently, we were interested in inaccuracies, i.e., deviations between the actual and predicted values. We displayed the deviations using the plt.subplots () command by using a confusion matrix. This matrix is used to analyze the results of the classification.

The diagonal contains the correctly assigned elements, in our case, keywords. Outside the diagonal, there are incorrectly assigned elements with respect to the classification model, i.e., keywords that the linear support vector model included in a different subcategory than initially assigned.

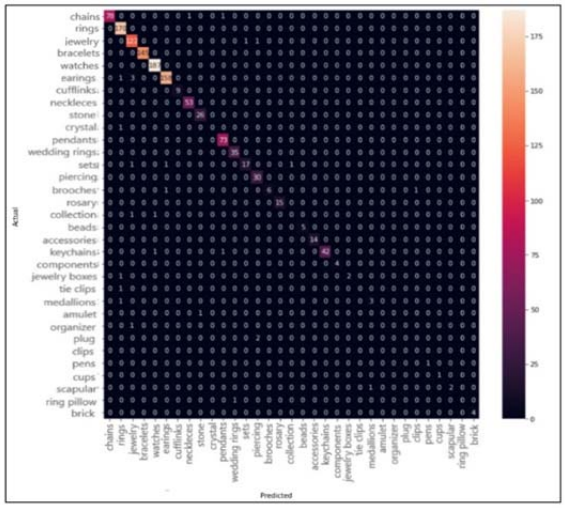

Figure 5. The difference between the actual and predicted values in the model of linear support vectors
Source: own processing

The highlighted values on the diagonal indicate the number of correctly assigned keywords in each category (fact = prediction). For example, the first box indicates the case of the category Chain, where out of a total of 80 keywords within that category, 78 keywords were correctly assigned.

All three applied models that we used proved to be suitable for the classification of text documents. We expressed the quality of the applied models through the degree of accuracy, which in percentage expresses the accuracy based on which the model classified the keywords into the relevant subcategories (Fig. 6.). The linear support vector model achieved the highest rate of accuracy, up to 96.82 %. The multinomic logistics regression model achieved the second-highest level of accuracy 91.78 %. The last, third model, which achieved the lowest accuracy rate of the three selected models, precisely at 72.19 %, is the multinomial probability classifier based on the Bayes theorem.

Based on the accuracy rate, we chose the linear support vector model as the best model.

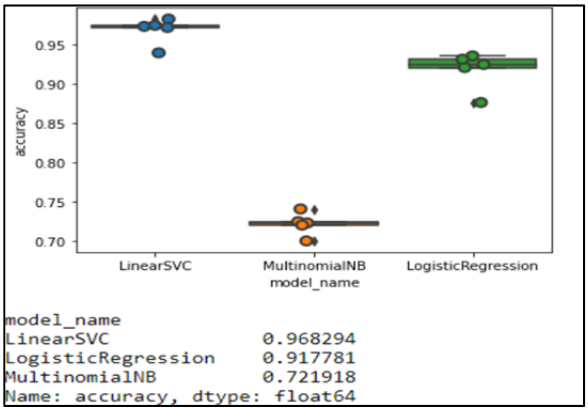

Figure 6. A visual view of the accuracy rate of all three applied models
Source: own processing

5. Conclusion

The current world population statistics published by the global provider of global statistics Worldometer show the current world population as of March 17, 2021, 7,852,736,170 inhabitants. Of this population, more than 65% have an internet connection. In the case of the Slovak Republic, we have current data from 2019, which indicates that in 2019, 76% of the population aged 12 to 79 had an internet connection, i.e., almost 8 out of 10 Slovaks in the relevant age range were online. Every single person with internet access searches for something in search engines every day.

At present, the Internet not only serves to mediate communication and entertainment, but the possibilities of webspace have mainly begun to be used mainly by entrepreneurs, whether to promote products or services or directly for trading on the web. The current situation has been significantly affected by the COVID-19 pandemic, in which people were forced to use the Internet more often and to a much greater extent than they had been accustomed to, as all shops except grocery stores and those that sell basic necessities were closed.

Sales or purchases through online stores could only be made in cooperation with the customers themselves (on the Internet, these are visitors to the selected website). It is the visitors who satisfy their needs via the Internet by searching for various products, services, or anything else. Initially, visitors only needed to orient themselves among dozens of websites, which did not present any complications. However, there are currently more than a billion websites worldwide available for visitors to choose from.

The whole search process works by a visitor entering a word or phrase in a search engine, called a keyword, based on which the search engine will offer them several thousand results. The ranking of search engine results is very important, as more than 70% of visitors choose from the first five results offered by a search engine. For that reason, website owners try to appear in the first five positions in search results. The more visitors, the more sales, resulting in higher profits. With more than a billion web pages, it is not that easy to appear in the top 5 search results. The key to getting the best ranking is to know the keywords that visitors enter into the search engine. If the owner knows these keywords and has them implemented into their website in texts, menus, etc., there is a high probability that his website will be in the highest possible search results, i.e., in good positions.

In this paper, we decided to describe the process of keyword analysis with a specific example in the field of jewellery, and while categorizing keywords, we used selected machine learning algorithms. In all categories, the support vector model came out as the most accurate model. This model achieves the highest level of accuracy in all 23 categories, as well as in the Product category. The accuracy of the models gradually decreased in individual categories. One of the reasons may be the lower number of keywords within the categories, which means that the learning system had less information available, i.e., a lower training sample available. Finally, it should be emphasized that the choice of the model itself is different for each application and depends on several factors. In the case of keyword analysis, the support vector model has been shown to better capture the occurrence of keywords, working more efficiently in categorizing them and minimizing the margin of error as well as the ability to learn. A huge benefit in this case is not only the streamlining of the entire process of keyword analysis, but also the resulting increase in the profitability of the company.

Acknowledgements

This work was supported by the project VEGA 1/0561/21 The impact of the COVID-19 crisis on business demography and employment in the Slovak Republic and the EU.

References

[1]. Abe, S. (2005). *Support vector machines for pattern classification* (Vol. 2, p. 44). London: Springer.

[2]. Aggarwal, C. C., & Zhai, C. (Eds.). (2012). *Mining text data*. Springer Science & Business Media.

[3]. Azam, N., & Yao, J. (2012). Comparison of term frequency and document frequency based feature selection metrics in text categorization. *Expert Systems with Applications*, *39*(5), 4760-4768.

[4]. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

[5]. Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.

[6]. Feng, G., Guo, J., Jing, B. Y., & Sun, T. (2015). Feature subset selection using naive Bayes for text classification. *Pattern Recognition Letters*, *65*, 109-115.

[7]. Kaushik, A. (2009). *Web analytics 2.0: The art of online accountability and science of customer centricity*. John Wiley & Sons.

[8]. Khan, G. F., & Wood, J. (2015). Information technology management domain: emerging themes and keyword analysis. *Scientometrics*, *105*(2), 959-972.

[9]. Killoran, J. B. (2013). How to use search engine optimization techniques to increase website visibility. *IEEE Transactions on professional communication*, *56*(1), 50-66.

[10]. Li, Y. H., & Jain, A. K. (1998). Classification of text documents. *The Computer Journal*, *41*(8), 537-546.

[11]. Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

[12]. Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006, December). Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In *Australasian joint conference on artificial intelligence* (pp. 1015-1021). Springer, Berlin, Heidelberg.

[13]. Šoltés, E. (2020). Regresná a korelačná analýza s aplikáciami v softvéri SAS. *LetraEdu, Bratislava*. ISBN 978-80-89962-38-9.

[14]. Vapnik, V. (1998). The support vector method of function estimation. In *Nonlinear modeling* (pp. 55-85). Springer, Boston, MA.

[15]. Zhu, X., & Tan, Z. (2012, September). SEO keyword analysis and its application in website editing system. In *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing* (pp. 1-4). IEEE.

[16]. Wang, D., Zhang, H., Liu, R., Lv, W., & Wang, D. (2014). t-Test feature selection approach based on term frequency for text categorization. *Pattern Recognition Letters*, *45*, 1-10.