Recognising innovative companies by using a diversified stacked generalisation method for website classification

Marcin Michał Mironczuk¹ · Jarosław Protasiewicz¹

Published online: 22 June 2019
© The Author(s) 2019

Abstract

In this paper, we propose a classification system which is able to decide whether a company is innovative or not, based only on its public website available on the internet. As innovativeness plays a crucial role in the development of myriad branches of the modern economy, an increasing number of entities are expending effort to be innovative. Thus, a new issue has appeared: how can we recognise them? Not only is grasping the idea of innovativeness challenging for humans, but also impossible for any known machine learning algorithm. Therefore, we propose a new indirect technique: a diversified stacked generalisation method, which is based on a combination of a multi-view approach and a genetic algorithm. The proposed approach achieves better performance than all other classification methods which include: (i) models trained on single datasets; or (ii) a simple voting method on these models. Furthermore, in this study, we check if unaligned feature space improves classification results. The proposed solution has been extensively evaluated on real data collected from companies' websites. The experimental results verify that the proposed method improves the classification quality of websites which might represent innovative companies.

Keywords Innovative detection · Text classification · Ensemble learning · Multiple views · Multi-view learning

1 Introduction

It is argued that innovativeness plays a crucial role in the development of various branches of the modern economy, such as medicine, agriculture, security, defence, industry. Providing advanced products and services depends on smooth and efficient cooperation between companies and research units. As these two groups may have some difficulties in finding appropriate partners or information, we have created an information platform, Inventorum which aims to establish connections between them. In order to prevent users from being overwhelmed or distracted by useless data, the system recommends only innovations, projects, partners, and experts that suit them. Such an

approach requires both preparations of relevant information and its careful selection. In this paper, we focus on finding potentially innovative companies on the internet, which should help to find more participants of Inventorum, i.e. information consumers. The system selectively gathers data from the internet. Then, each acquired website consisting of a group of web pages is analysed to establish whether it is related to an innovative company or not. This study evaluates classification of such websites based on two groups, namely (i) domains that represent innovative companies and (ii) others. For simplicity, we name these groups as (i) innovative domains or websites and (ii) non-innovative domains or websites. This classification does not mean that internet domains are innovative or not, but does indicate whether the companies represented in these domains are innovative or not.

In reality, an innovative business is one that offers innovative products or services, employs professionals or scientists that are the best in their profession, invests in research and development projects, cooperates with other creative firms or top champions [9, 36, 71]. The above factors are disputable except one which is an innovative product or service. Oslo Manual [48] defines innovative as something new or substantially enhanced and concurrently beneficial for consumers. Although it may be barely an

✉ Marcin Michał Mironczuk
marcin.mironczuk@opi.org.pl

Jarosław Protasiewicz
jaroslaw.protasiewicz@opi.org.pl

¹ Laboratory of Intelligent Information Systems, National Information Processing Institute, al. Niepodległości 188 b, 00-608 Warsaw, Poland

idea that triggers organisational, processing, or production change resulting in an invention, implementation and diffusion [35]. Let us imagine how an individual can evaluate a company based on its website. At first, he or she should look for innovations or legal protections. If these appear on the website and belong to this firm, it may be innovative. Nonetheless, not all entities possess innovations, but they yet may be innovative because they might work on something ground-breaking. Thus, the individual analysing the company's web page must look for the inconclusive factors mentioned above, i.e. the particular types of people, investments and cooperation.

We must mention that some attempts were made to construct models for assessing company innovativeness based on inquiry tools, e.g. a survey to benchmark organisational readiness for digital innovation [43] or a questionnaire to find a relationship between cooperation with innovative partners and internal innovations [53]. However, these techniques require conducting surveys on companies under examination. We want to develop an automatic model utilising machine learning techniques. As we can imagine, this task is not easy even for an expert, so the question is how it can be realised by a classification algorithm meant to substitute the expert's work. Firstly, our imaginary algorithm has to retrieve innovation entities such as a person or a cooperating company. Then, it should assess whether these objects are relevant to the firm under examination. More specifically, the algorithm needs (i) to judge whether potential innovations are actually innovative; (ii) to assess whether persons are unique professionals; and (iii) to determine if related companies are innovative. According to our knowledge, these tasks are impossible to realise by any machine learning algorithm because the algorithm is not able to judge directly, for example, if a product/service or a professional is innovative or not without evaluating almost all possibilities across the world. Thus, an indirect technique is required which we propose below.

Although the concept of "an innovative website" is an abstract idea similar to concepts such as what constitutes spam, pornography, or phishing [1, 4, 6, 39, 52, 68, 74, 75], it has never been adequately addressed by known machine learning techniques. Unfortunately, innovativeness is difficult to grasp and cannot be recognised by identifying typical words, phrases, origin addresses, or similar, as in spam detection. Innovativeness is something that we must determine by context, so we propose an original procedure, which assumes that it is possible to create a machine learning classification model that can determine whether a company is innovative or not based on automatic analysis of its website. It analyses a data set from three different perspectives. The first perspective considers

only a company's description localised on its website. This text most likely contains descriptions of innovative products or services, and some notes about the most crucial professionals. The second perspective analyses links included on the company's website which may contain information about cooperating companies. The third perspective utilises all text in the website. It may be important if our assumptions regarding the localisation of particular information on the website are inappropriate.

In our previous study [47], we proposed an experimental Naïve Bayes (NB) classification committee for categorisation of companies' websites into the *innovative* or *non-innovative* group. The committee was based on a voting idea and diversified feature spaces, feature weights, and two models of features distribution, i.e. Bernoulli and Multinomial [19, 44]. We showed that the committee, as well as the selected individual NB classifiers, are able to resolve the defined classification task sufficiently. Furthermore, we noted that the proposed ensemble procedure may require some improvement to provide better results. Therefore, in this study, we explore this task more deeply. The novelties of this paper are as follows:

1. We investigate a novel classification task, i.e. recognition of Polish innovative companies' websites.
2. We propose a novel improved classification approach based on multiple data sources to accomplish the classification task mentioned above.

In the recent literature, we did not find any related works that would include the topics mentioned above. In this study, we propose a new classification committee, which is an extended version of our previous proposition [47]. The primary enhancement relies on the application of a stacked generalisation method instead of a simple voting procedure. The objectives of the study are as follows:

1. To demonstrate that the new proposed method can improve the decision process and produce better overall classification quality than the previously proposed voting committee.
2. To demonstrate that a genetic algorithm can create an unaligned feature space that improves the classification results of the simple voting procedure.

Both objectives are realised based on empirical experiments and obtained results.

This paper is structured as follows. Section 2 summarises previous and related works. Section 3 presents an overview of the proposed classification system of innovative websites. Section 4 describes the evaluation process of the proposed system and the results obtained during experiments. Finally, Section 5 concludes the findings.

2 Related works vs. proposed approach

A review of recent literature indicates that there are some works concerning detection of innovative themes on the internet [7, 50, 51]. However, the analysed approaches are insufficient to solve the problem presented in this study owing to the reason that they do not offer a direct solution to recognise whether a company is innovative or not, based only on its public website available on the internet. Thus, we constructed a classification model that can recognise innovative websites on the internet with sufficient quality. We propose a method related to ensemble learning. Ensemble learning refers to procedures employed to train multiple classifiers by combining their outputs and considering them as a “committee” of decision makers. Various approaches accomplish this learning concept, for example, bagging, boosting, AdaBoost (adaptive boosting), stacked generalisation, mixtures of experts, and voting based methods [2, 5, 11, 14, 40, 42, 58, 73]. The presented approach is closely related to stacked generalisation [66, 67, 70]. Stacking is an ensemble learning technique, where a set of models is constructed from bootstrap samples of a data set, and then their outputs as a hold-out data set are used as inputs to a meta-model. The set of base models is called level-0, whereas the meta-model is considered as level-1. The task of the level-1 model is to combine the set of level-0 outputs so as to classify the target correctly, thereby correcting any mistakes made by the level-0 models [60].

There are some differences between our approach and the approach mentioned above. Usually, typical classification methods, as well as classification tasks, utilise only one data source to construct a classification model. These data include information describing a target class, i.e. *innovative* vs. *non innovative*, *spam* vs. *no spam*, etc. In our case, three data sources represent different aspects of the destination concept that is the *innovative company*. Thus, we can state that the classification is based on multiple sources. In our first research [47], we combined these sources and proposed a classification method to obtain satisfactory classification indicators. However, the simple classification, which utilised each data set separately, produced insufficient results. Appendix B covers additional analysis of the classification results achieved by such an approach. The empirical results suggest that a solitary classifier trained on a single dataset may be not able to obtain a satisfactory classification quality.

The stacked generalisation method splits one data source into several n -bootstrap samples to learn n classification functions (level-0), and then uses one classification function to classify the remainder of the data (level-1). In our case, we learn one classification function per a data source (level-0), and the results of these classifications are used to

build the level-1 classifier. Thus, we name this method: diversified stack generalisation. Thanks to this approach, we obtain a single three-dimensional feature space in the case when we have three data sources. The validation of this method is simple, especially when we have a small number of learning examples and an unbalanced number of positive and negative examples.

Moreover, in our previous work, we assumed an aligned feature space, i.e. each classifier in the committee used the same feature space and the number of features was the same for all classifiers. In this study, we evaluate the case in which there are unaligned feature spaces, i.e. the number of features is not the same for each classifier. In this manner, we check whether there are other combinations of features that may improve classification results.

3 Innovative website recognition system

This section presents the proposed classification system for innovative websites and explains the general aspects of the proposed system at a high level of abstraction. Figure 1 depicts a flow chart outlining its structure. Section 1 presents in detail how the proposed method works.

Below, we investigate how the proposed system works based on a simple use case. This use case explains how a single company website, which is a collection of HTML documents, is processed. We can generalise this case to n -websites. The system (Fig. 1) considers a company's website as an input. A *pre – processing* phase processes the HTML documents and creates three different data sets. The first data set ($Data\ set_1 (L_D)$) contains a textual description of the company. This description is extracted from the first (index) page of the website. The second data set ($Data\ set_2 (L_L)$) involves the link labels that were extracted from the index page. The last data set ($Data\ set_3 (L_B)$) consists of a so-called *big document*. The *big document* is constructed based on all of the collected HTML documents, i.e. it is the conjunction of several appropriately selected documents from the website. This representation is a concatenation of the n best documents from a given website as determined by the Okapi BM25 search system [33, 44, 47, 57] (for more technical details, see Appendix A). The textual data from each data set is classified by an appropriate classification function. We use the NB learning method to create each function. The results of the classification are merged (D_{MLE} - is a matrix which stores the collected decisions) and are utilised by a *meta – classifier* (γ_{ML}). We used and tested several learning methods, such as k nearest neighbours (k-nn), Naïve Bayes (NB), support vector machine (SVM), and decision trees (DT) [2, 17, 19, 24, 34, 44] to create the final decision classification function. The result provides

Fig. 1 General overview of the diversified stacked generalisation architecture

information on whether a given website describes an innovative business or a non-innovative company.

4 Empirical evaluation of the solution

4.1 The data set and evaluation method

We decided to create an original test set due to the lack of such data in other works. We labelled 2,747 real websites (509 as innovative websites, 2,238 as non-innovative

websites), and created three sets as follows: $|L_D| = 2,747$ description examples described by $|F_D| = 140,699$ features that also include information about logotypes, $|L_L| = 2,747$ examples of the link labels described by $|F_L| = 140,271$ features that also include information about logotypes, and $|L_B| = 2,747$ examples of the *big documents* described by $|F_L| = 663,015$ features.

It is worth mentioning that to build the data set, we used annual rankings of leading companies on the market, e.g. Forbes rankings and awards such as Business Gazelle. We assumed that these rankings are highly reliable and utilised this data to collect company websites as learning data sets describing innovative companies. Opposite examples i.e. non-innovative internet sites, were obtained from a collected set of common company websites.

The classification quality is evaluated according to the 10-fold cross-validation procedure and measured by error, accuracy, precision, recall, and F-measure. In fact, it is the F1 measure that assumes an equal balance of precision and recall [44, 61, 64].

Furthermore, we applied statistical tests to verify whether the F -score distributions of different models for meta-learning classification (γ_{ML}) and voting classification (γ_{Voting}), are identical without assuming that they follow the normal distribution. The tests' procedures varied depending on the type of feature space.

When the feature space was aligned (Section 4.2), we compared the results produced by the voting method and each meta-learning method by using two tests. First, Friedman's aligned rank test verifies the null hypothesis H_{F0} stating that the performances of experimental models are equal [18, 22], assuming that the statistical significance, α is equal to 0.05. Second, the Wilcoxon signed-rank statistic test using the Bonferroni correction of p-values for multiple testing (corrected pairwise tests) [20, 49, 59] tests the null hypothesis, H_{W0} stating that the median difference between pairs of experimental models is zero, assuming that α is equal to 0.05.

Conversely, when the feature space was unaligned (Section 4.3), we compared the results coming from the voting procedure and the best meta-learning method for the aligned feature space with results coming from the same models but for the unaligned feature space. For this purpose, we used the Wilcoxon rank-sum test [20, 49]. The null hypothesis, H_{WR0} of the Wilcoxon rank-sum test is that the median difference between pairs of experimental models is zero, and α is equal to 0.05.

We used the Scamp R-project package [13] to execute Friedman's aligned rank test, and the standard Stat package [55] to perform the Wilcoxon signed-rank test with Bonferroni correction, and the Wilcoxon rank-sum test (Wilcoxon Mann-Whitney test).

4.2 Aligned space of features

Simple diversified classification committee γ_{Voting} Our previous experiments [47] were intended to compare the best NB classifiers (γ_{NB1} , γ_{NB2} , and γ_{NB3}) with γ_{Voting} . The γ_{Voting} classification function is based on a simple voting process. For each classifier, we choose the class label of the highest weight, and then we select the most common label from all the decisions. We recall some of those results below to provide background regarding the applied meta-learning techniques and justify the differences between the simple voting and meta-learning methods. Figure 2 shows the comparison of results for each classification method mentioned above.

Our previous research indicated that methods for feature selection have a significant influence on the classifiers' performance. For example, we observed that the γ_{NB1} classifier produced the best outcomes when the χ^2 feature selection method was used, whereas the γ_{NB2} and the γ_{NB3} classifiers performed best when using the *Fisher* feature selection method.

Figure 2 compares the values of precision, recall, and F-measure of γ_{Voting} and the best three NB classifiers. Based on these results, we can conclude that the performance of the classifiers depends not only on the feature selection method but also on the number of input features. More specifically,

the committee outperforms every single classifier when the number of features varies between 600 and 4,000. However, further increase in the number of features leads to decrease of its F-measure. Although the higher number of the feature in the γ_{NB1} classifier with the χ^2 feature selection method leads to better performance than the committee, precision and recall values inform us that the increase in the number of features causes over-fitting of almost all classifiers and the committee. The over-fitting, manifested by the precision increase and the recall decrease, represents an incorrect performance of the classifiers because many innovative companies are not selected; it is better for the purpose of this study that some non-innovative companies are labelled as innovative (lower precision) provided a high number of truly innovative businesses are found from the internet datasets. Moreover, we applied an information retrieval system in the γ_{NB2} classifier, which led to high-quality performance of the classifier when the number of features was very high.

It can be concluded that the simple committee produces the firmest decisions, and concurrently, the proposed information retrieval system improves the classification quality and prevents from over-fitting.

Figure 3 presents a visualisation of the data points in the constructed three-dimensional feature space that was built based on the values representing the probabilities of

Fig. 2 Precision, Recall, and F-measure achieved by the best NB classifiers γ_{NB1} , γ_{NB2} , γ_{NB3} and the γ_{Voting} classifier, i.e. the committee [47]

Fig. 3 Plot of the feature values that were received from each classification function, i.e. γ_{NB1} , γ_{NB2} , γ_{NB3} (10,100 features, 473 training innovative examples, and 1,999 training non-innovative examples)

the innovative class of every single classifier function, i.e. γ_{NB1} , γ_{NB2} , γ_{NB3} .

It can be observed (see Fig. 3) that classes are well separated. The presented results indicate that we can use other supervising classification method(s) Γ_{ML} to create classification function(s) based on such a created feature space. In our case, we refer to this type of learning as meta-learning classification.

Meta-learning classification We have tested some learning methods in the Γ_{ML} category to determine the final meta-learned classifier γ_{ML} (Fig. 6). Figures 4 and 5, and Table 1 show the results of the classification experiments in respect to different meta-classifiers γ_{ML} and the features number in the range of 600–10,100. More specifically, there were tested six meta-classifiers. Three based on k-nn considering different numbers of neighbours and three other algorithms (NB, SVM, and DT). Figure 4 depicts precision, recall, and F-measures achieved by these meta-classifiers. All meta-classifiers γ_{ML} work better than γ_{Voting} when the number of features is average or high. However, their quality is lower than γ_{Voting} at the lowest range of features. The best meta-classifiers are based on DT, SVM, and k-nn with three or nine neighbours. By the contrast, the lowest enhancements are produced by meta-classifiers based on NB and k-nn with one neighbour.

Additionally, we calculated the mean values of precision, recall, F-measure, error, and accuracy for each meta-classifier γ_{ML} and γ_{Voting} in the specified range of features, i.e. from 600 to 10,100. The results are included in Table 1. The γ_{Voting} has two values for each indicator. The first value (marked by *) is calculated for the entire range of features, whereas the second (marked by **) analyses the number of features in the range of 600 to 4,500. The average results prove the above conclusions. Moreover, they show clearly that γ_{Voting} performs better when the number of values is low. However, when the number of features increases, all meta-classifiers γ_{ML} improve. Additionally, Fig. 5 presents the differences of F-scores between γ_{Voting} and each meta-classifier γ_{ML} . Thus, the improvements introduced by the new methods are clearly apparent.

Furthermore, Friedman's aligned rank test indicated that the F-scores of the seven best classification models, for which the number of features ranges from 600 to 10,100 (see Table 1), were significantly different ($p - value < 2.2e - 16$). Thus, we can reject the hypothesis, H_{F0} . In addition, we achieved the following p -values for multiple pairwise comparisons made using the Wilcoxon signed-rank test (1) $p - value = 1.5e - 13$ for γ_{Voting}^* vs. γ_{ML} , k-nn, k=1, (2) $p - value = 5.4e - 16$ for γ_{Voting}^* vs. γ_{ML} , k-nn, k=3, (3) $p - value = 5.2e - 16$ for γ_{Voting}^* vs. γ_{ML} , k-nn, k=9, (4) $p - value = 3.8e - 16$ for γ_{Voting}^* vs. γ_{ML} , SVM, (5) $p - value = 1.1e - 15$ for γ_{Voting}^* vs. γ_{ML} , NB, and (6)

Fig. 4 Precision, Recall and F-measure of the different meta-classifiers γ_{ML}

p -value = $4.2e - 16$ for γ_{Voting}^* vs. γ_{ML} , DT. The test showed that there are significant differences between the pairs compared, suggesting that these pairs are independent. As a consequence, we can reject the null hypothesis, H_{W0} .

4.3 Unaligned space of features

Although each classifier in the committee uses a different feature space, the number of features is the same for all classifiers. This is probably a suboptimal solution because it is apparent that the classifiers produce the best results in distinct ranges of feature numbers (Fig. 2). Thus, the optimal system may involve an adaptive selection of the feature numbers for each classifier in the committee.

As finding an optimal feature space is similar to selecting the best population among many possible solutions, we decided to apply a genetic algorithm to solve this problem. We conducted several computational experiments to verify this assumption. They involved choosing an optimal feature

space by the genetic algorithm for the γ_{Voting} classifier, as well as the meta-classifier γ_{ML} based on the SVM. Table 2 lists the best and average values of precision, recall, F-measure, error, and accuracy obtained in the experiments.

Firstly, note that the γ_{Voting} classifier utilised approximately 4,500 features because we previously observed that when the number of features is higher than this value over-fitting may occur. Thus, the genetic algorithm searches optimal solutions without the risk of creating an over-fitted model. For example, the combination of 3,258 features for the γ_{NB1} , 812 features for the γ_{NB2} and 4,092 features for the γ_{NB3} functions produce the optimal solution for the γ_{Voting} classifier (see the first row in Table 2). The average outcomes of the genetic algorithm in the case of the γ_{Voting} model were calculated by using a 10-fold cross-validation procedure. When we compare these results (see the second row in Table 2) with outcomes covered in Table 1, we note that there are some combinations of features that slightly improve the classification quality. Moreover, the Wilcoxon

Fig. 5 Comparison of the difference between F-measures of the classifier based on γ_{Voting} and meta-classifiers γ_{ML}

signed-rank test compared γ_{Voting} for the unaligned feature space (Table 2) with γ_{Voting}^{**} for the aligned feature space (Table 1). It indicated that $p - value < 2.2e - 16$. The test showed that there is a difference between the pair compared. We conclude that this pair is independent. As a consequence, we can reject the null hypothesis, H_{WR0} .

Secondly, note that the maximum number of features for the meta-classifier γ_{ML} , which is based on SVM, was equal to approximately 20,000 (the third and fourth rows in Table 2). This boundary has been selected experimentally with consideration of the problems of over-fitting and acceptable performance. For instance, if we take into account the F-measure, the following combinations of features are optimal: 14,207 features for γ_{NB1} , 16,081 for γ_{NB2} , and 16,283 for γ_{NB3} . The average number of

indicators for the γ_{ML} models were achieved by using a 10-fold cross-validation procedure. Note that the genetic model selected more features than the specified upper boundary, i.e. 10,100. Furthermore, the Wilcoxon signed-rank test compared γ_{ML} for the unaligned feature space (Table 2) with γ_{ML} , SVM for the aligned feature space (Table 1). Since the test indicated that $p - value < 2.2e - 16$, there is a difference between the pair compared, suggesting that this pair is independent. As a consequence, we can reject the null hypothesis, H_{WR0} .

Finally, we tested the meta-classifiers γ_{ML} with the genetic algorithm again, but we assumed that the maximum number of features should not exceed the value of 10,100 (the rows marked * in Table 2). The meta-classifiers γ_{ML} performed best when the genetic algorithm selected the

Table 1 The comparison of the mean values of basic indicators for various types of learning methods (* or without a mark - the averages are calculated for number of features in the range of 600 to 10,100; ** - the averages calculated for number of features in the range of 600 to 4,500)

Experiment name	Precision	Recall	F-measure	Error	Accuracy
γ_{Voting}^*	0.924	0.588	0.715	0.087	0.913
γ_{Voting}^{**}	0.865	0.625	0.724	0.088	0.912
γ_{ML} , k-nn, k=1	0.81	0.763	0.786	0.077	0.923
γ_{ML} , k-nn, k=3	0.865	0.768	0.813	0.065	0.935
γ_{ML} , k-nn, k=9	0.896	0.756	0.82	0.061	0.939
γ_{ML} , SVM	0.878	0.775	0.823	0.061	0.939
γ_{ML} , NB	0.797	0.765	0.781	0.08	0.92
γ_{ML} , DT	0.87	0.783	0.823	0.062	0.938

Table 2 The comparison of basic indicators for different type of feature space size

Experiment name		Precision	Recall	F-measure	Error	Accuracy
γ_{Voting}	The best outcome	0.882	0.692	0.775	0.0743	0.9257
	The average outcomes	0.877	0.686	0.77	0.0759	0.924
γ_{ML} based on SVM	The best outcome	0.962	0.886	0.922	0.0277	0.9723
	The average outcomes	0.941	0.857	0.897	0.0364	0.9636
	The best outcome*	0.908	0.831	0.868	0.047	0.953
	The average outcomes*	0.886	0.769	0.823	0.0611	0.9389

following combinations of features: 8,662 for γ_{NB1} , 8,022 for γ_{NB2} , and 9,481 for γ_{NB3} . Interestingly, in most cases, the algorithm searched for the optimal solutions in the range when the feature numbers were higher than 10,100. Moreover, we compared γ_{ML} for the unaligned feature space (see rows marked * in Table 2) with γ_{ML} , SVM for the aligned feature space (Table 1) by using the Wilcoxon signed-rank test. Since the test indicated that $p - value = 0.57$, there is no difference between the pair compared. Therefore, we conclude that this pair is dependent. As a consequence, we cannot reject the null hypothesis, H_{WR0} .

5 Conclusions

Nowadays, owing to the growing need for innovativeness, modern companies use highly complex processes to develop increasingly advanced products in order to gain a competitive edge in the market. Such strategies require sophisticated knowledge and skills, as well as high organisational and technical culture. These attributes characterise businesses that either employ in-house researchers or outsource their services. Thus, suitable matching between the firms and the researchers is crucial for boosting innovativeness. One of the aspects of this process is the identification of innovative companies that can be based on their internet websites.

In this study, we developed the novel diversified stacked generalisation method for recognition of websites that may indicate innovative companies. The main idea behind this approach is that we consider an entire data source as one bootstrap sample and learn one classification function for each data source. In our case, we consider three data sources, which thus represent three bootstrap samples. Then, the classification results from each bootstrap sample are used to build the final meta-classifier.

Because we consider three data sources, a single three-dimensional feature space is achieved. The system has been verified experimentally regarding various combinations of the feature numbers and meta-classification methods. A genetic algorithm selects the combinations of features. We experimentally confirmed that the simple voting classifier

performs better when the number of features is low, but as features increase, all meta-classifiers improve. The proposed system is robust to over-fitting, especially when big documents are considered.

In conclusion, the most important findings of this work are as follows:

- We can recognise innovative companies with accurate precision based on analysis of a textual content of companies' websites. For this purpose, we may use machine learning techniques, such as those applied in the proposed method.
- The proposed diversified stacked generalisation method (the multi-sources classification model), based on the meta-classifier outperforms the voting approach. The results were improved by 11% in terms of the F-score mean.
- We may utilise a genetic algorithm to construct the unaligned feature space. More specifically, we can create various combinations of feature numbers originating from the different data sources, and in this manner improve the classification results. The results of the simple voting procedure were improved by 4.6% in terms of F-score mean. On the other hand, we did not notice a significant improvement in the results of a meta-classifier that utilises the aligned feature space in comparison to a meta-classifier which uses the unaligned feature space.
- The proposed method creates non-over-fitted models over some ranges of feature numbers.
- We achieve a well separated three-dimensional feature space that is simple to visualise.

In further work, we will attempt to better evaluate our method by using other classification approaches and data sets. For instance, we believe that the method could be applied to the problem of multilingual text classification. Preliminary experiments indicate significant classification improvements in this field [54].

In our current approach, the training set is created by experts. We believe that through these data the classification system contains unique knowledge, which allows the system to reject non-innovative companies.

Another issue is the model robustness to fake innovativeness such as using buzzwords or just advertising on a website. Since classification models are constructed in a supervised way, they are as robust as the training set is correct and general. The system may classify companies improperly if unknown features, which indicate innovativeness, appear in real data. On the other hand, we must underline that the proposed system is not resistant to poisoning attacks by including corrupted data in a training dataset. A poisoning attack is the issue of data or applications security, and it was beyond the study. It may be the topic of further research.

Finally, we argue that complex classification tasks require more sophisticated approaches than applying a single model. The diversified stacked generalisation method integrates many views (classifiers) on the same problem, and in this manner, categorises data sufficiently.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Appendix A: Formal description of the system

This section presents the formal description of the proposed classification system for innovative websites. Figure 6 depicts a flow chart outlining its detailed structure. We use *function interfaces* to illustrate how the system works. The interfaces are considered in terms of computer science.

Figure 6 shows a detailed overview of the classification system for innovative websites. The framework consists of two main phases, namely a learning phase and an applied phase. The learning phase includes (i) a *processing₁* function, which is responsible for acquisition of learning data, (ii) the NB methods Γ_{NB1} , Γ_{NB2} , and Γ_{NB3} that create (iii) the classification functions γ_{NB1} , γ_{NB2} , and γ_{NB3} [19, 44], (iv) a *createMetaLearnExamples* function that constructs a new feature space and new learning examples for a meta-learning classification process, and (v) a method Γ_{ML} that creates the final meta-classifier. On the other hand, the applied phase consists of (i) a *processing₂* function that collects unlabelled data for classification, (ii) three classification functions γ_{NB1} , γ_{NB2} , and γ_{NB3} , (iii) a *createDataToClassification* function that prepares the received results from the classifier functions for final classification, and (iv) the final classification functions γ_{Voting} and γ_{ML} that are based on the simple voting classifier and the meta-classifier respectively.

A.1 Realisation of the standard and meta-learning phases

Realisation of the *processing₁* function If we do not have the classification function(s) (the first condition in Fig. 6, i.e. *is learning process?* is *yes*), then, in the first step, we implement the interface of the *processing₁* : $(R_L, n_p, n_n, q) \mapsto (L_D, L_L, L_B)$ function (Fig. 6, Notation 7). The notations from 1 to 6 describe the required elements for the realisation of this function.

Notation 1 Let C_L be a set (a labels set) of the two elements c_L and $\neg c_L$, where c_L is a positive class label, for example, $c_L = \text{"innovative website"}$, and $\neg c_L$ is a negative class label, for example, $\neg c_L = \text{"non innovative website"}$.

Notation 2 Let R_L be a set (a learning data set to crawl) of the triples $(idWebsite, h, c_L \text{ or } \neg c_L)$, where *idWebsite* is a crawled website identifier, for example, *idWebsite* = 1, *h* is a host of website, for example, *h* = *http://www.mostostal.waw.pl*, and $c_L \text{ or } \neg c_L$ is a label class as above.

Notation 3 Let F_L be a set of the features extracted from links (their labels) of the first pages of the given websites, i.e. $F_L = \{f_{l1}, \dots, f_{ln}\}$, where $f_{li} \in F_L$ is a feature (phrase, n-gram, and additional information, e.g. whether a given website contains appropriate logotypes), for example, $f_{li} = \text{contact}$, $f_{li} = \text{contain innovative logotype}$, etc.

Notation 4 Let F_D be a set of the features extracted from descriptions of the first pages of the given websites, i.e. $F_D = \{f_{d1}, \dots, f_{dn}\}$, where $f_{di} \in F_D$ is a feature (phrase, n-gram, and additional information e.g. whether a given website contains appropriate logotypes), for example, $f_{di} = \text{our company}$, etc.

Notation 5 Let F_B be a set of the features extracted from *big documents* that were constructed for all websites. For each website, we build a special text representation called a *big document*. This representation is a concatenation of the *n* best documents from a given website received by the Okapi BM25 search system [33, 44, 47, 57]. Then $F_B = \{f_{b1}, \dots, f_{bn}\}$, where $f_{bi} \in F_B$ is a feature (phrase, n-gram, name entity recognition label, noun phrase), for example, $f_{di} = \text{our company}$, $f_{bi} = \text{person}$, $f_{bi} = \text{city}$, $f_{bi} = \text{venture capital}$, etc.

Notation 6 Let L_L , L_D , and L_B be sets (learning data sets) of the *n* - *tuples* (the learning examples), such as $(idWebsite, w_{f11}, \dots, w_{f1n}, c_L \text{ or } \neg c_L)$ for the L_L set, $(idWebsite, w_{fd1}, \dots, w_{fdn}, c_L \text{ or } \neg c_L)$ for the

Fig. 6 Detailed overview of the diversified stacked generalisation architecture

L_D set, and $(idWebsite, w_{fb1}, \dots, w_{fbn}, c_L \text{ or } \neg c_L)$ for the L_B set, where w_{fli} , w_{fdi} , and w_{fbi} are the weights of the features from the F_L , F_D and F_B sets for the given $idWebsite$ respectively, and c_L , $\neg c_L$ are defined above.

Notation 7 Let $processing_1 : (R_L, n_p, n_n, q) \mapsto (L_D, L_L, L_B)$ be an interface of the processing function, which is based on the R_L, n_p, n_n, q parameters received from the L_D, L_L, L_B learning sets. The n_p is the amount of documents that are taken to construct a *big document* for the *innovative website*, where n_n is the number of documents used to construct a *big document* for the *non innovative website*, and q is a query sent to the Okapi BM25 search system, and R_L, L_D, L_L , and L_B are defined above. Also assume that $|F_D \cap F_L \cap F_B| \simeq 0$, i.e. L_D, L_L , and L_B are described by diversified disjunction (independent) sets of features.

Realisation of the learning functions After creating all necessary data sets mentioned above, we build the classification models. Additionally, if meta-learning is required, i.e. *is meta-learning process?* is *yes*, we learn meta-learning classification function(s) by using several classification methods, such as k-nn, NB, SVM, and DT. We call this meta-learning, because the output results of several classification functions are the inputs for the other classification function(s). The notations from 8 to 12 describe the required elements for the realisation of this phase.

Notation 8 Let Γ_{NB1} , Γ_{NB2} , and Γ_{NB3} be the classification methods that create the γ_{NB1} , γ_{NB2} , and γ_{NB3} classification functions based on the L_D, L_L , and L_B learning sets, respectively (the $\Gamma_{NB1} : L_D \mapsto \gamma_{NB1}$, $\Gamma_{NB2} : L_L \mapsto \gamma_{NB2}$, and $\Gamma_{NB3} : L_B \mapsto \gamma_{NB3}$ functions, respectively). Further assume that γ_{NB1} and γ_{NB2} are NB classifiers based on the Bernoulli distribution model and γ_{NB3} is an NB classifier based on the multinomial distribution model.

Notation 9 Let $R_{\gamma_{NB1}}$, $R_{\gamma_{NB2}}$, and $R_{\gamma_{NB3}}$ be the sets (the classification result sets) of the triples $(idWebsite, w_{cL, \gamma_{NB1}}, w_{\neg cL, \gamma_{NB1}})$, $(idWebsite, w_{cL, \gamma_{NB2}}, w_{\neg cL, \gamma_{NB2}})$, and $(idWebsite, w_{cL, \gamma_{NB3}}, w_{\neg cL, \gamma_{NB3}})$ respectively, where $idWebsite$, γ_{NB1} , γ_{NB2} , γ_{NB3} , c_L and $\neg c_L$ are defined above, $w_{cL, \gamma_{NB1}}$, $w_{cL, \gamma_{NB2}}$, and $w_{cL, \gamma_{NB3}}$ are weights, i.e. the probabilities of the c_L class computed by the γ_{NB1} , γ_{NB2} , and γ_{NB3} classification functions respectively, and $w_{\neg cL, \gamma_{NB1}}$, $w_{\neg cL, \gamma_{NB2}}$, and $w_{\neg cL, \gamma_{NB3}}$ are also weights, i.e. the probabilities of the $\neg c_L$ class computed by the $\gamma_{NB1} : L_D \mapsto R_{\gamma_{NB1}}$, $\gamma_{NB2} : L_L \mapsto R_{\gamma_{NB2}}$, and $\gamma_{NB3} : L_B \mapsto R_{\gamma_{NB3}}$ classification functions, respectively.

Notation 10 Let D_{MLE} be the set (the meta-learning examples set) of the quadruples $(idWebsite, w_{cL, \gamma_{NB1}}, w_{cL, \gamma_{NB2}}, w_{cL, \gamma_{NB3}}, c_L)$, where $idWebsite$, $w_{cL, \gamma_{NB1}}$, $w_{cL, \gamma_{NB2}}$, $w_{cL, \gamma_{NB3}}$, and c_L are defined above.

Notation 11 Let $createMetaLearnExamples : (R_{\gamma_{NB1}}, R_{\gamma_{NB2}}, R_{\gamma_{NB3}}) \mapsto D_{MLE}$ be the interface of a function to create the learning examples for the meta-learning classifier Γ_{ML} , which is based on the $R_{\gamma_{NB1}}, R_{\gamma_{NB2}}$, and $R_{\gamma_{NB3}}$ parameters and receives the D_{MLE} learning sets. The $R_{\gamma_{NB1}}, R_{\gamma_{NB2}}, R_{\gamma_{NB3}}$, and D_{MLE} are defined above.

Notation 12 Let Γ_{ML} be the classification method that creates the γ_{ML} classification function based on the D_{MLE} meta-learning set (the $\Gamma_{ML} : D_{MLE} \mapsto \gamma_{ML}$ function). Also, we assume that we consider learning methods, such as k-nn ($k = 1, k = 3$, and $k = 9$), NB classifier based on the Normal distribution model, SVM, and DT.

A.2 Realisation of the applied phase

After creating all necessary classification models and the meta-learning model, we can realise the applied phase. The notations from 13 to 18 describe the required elements for realisation of this phase.

Notation 13 Let R_C be a set (the data set to crawl) of the tuples $(idWebsite, h)$, where $idWebsite$ and h are defined above.

Notation 14 Let D_L, D_D and D_B be a set (a crawled data set for classification) of the $n - tuples$, such as $(idWebsite, w_{fd1}, \dots, w_{fdn})$ for the D_L set, $(idWebsite, w_{fd1}, \dots, w_{fdn})$ for the D_D set, and $(idWebsite, w_{fb1}, \dots, w_{fbn})$ for the D_B set, where w_{fli} , w_{fdi} , and w_{fbi} are defined above.

Notation 15 Let $processing_2 : (R_C, n, q) \mapsto (D_D, D_L, D_B)$ be an interface of the processing function, which is based on the R_C, n, q parameters and receives the D_D, D_L, D_B data sets for classification. The variable n is the number of documents used to construct a *big document*. R_C, D_D, D_L , and D_B are defined above.

Notation 16 Let D_C be the set (the classified data set) of the quadruples $(idWebsite, w_{cL, \gamma_{NB1}}, w_{\neg cL, \gamma_{NB1}}, w_{cL, \gamma_{NB2}}, w_{\neg cL, \gamma_{NB2}}, w_{cL, \gamma_{NB3}}, w_{\neg cL, \gamma_{NB3}})$, where $idWebsite$, c_L , $\neg c_L$, $w_{cL, \gamma_{NB1}}$, $w_{cL, \gamma_{NB2}}$, $w_{cL, \gamma_{NB3}}$, $w_{\neg cL, \gamma_{NB1}}$, $w_{\neg cL, \gamma_{NB2}}$, and $w_{\neg cL, \gamma_{NB3}}$ are defined above.

Notation 17 Let D'_C be the set (the meta data set for classification) with the same structure as D_{MLE} from Notation 10.

Notation 18 Let R_{Voting} and R_{ML} be the sets (the results data sets of the appropriate classification functions) of the tuples $(idWebsite, c_L \text{ or } \neg c_L)$ where $idWebsite$, c_L , and $\neg c_L$ are defined above.

The γ_{Voting} classification function is based on the simple voting process. For each classifier, we take the class label of the highest weight, and then we select the most common label from all the decisions. The γ_{ML} meta-classification function is one of the classification functions learned by the classification methods mentioned above.

Appendix B: Experiment results

This appendix covers the results of additional experiments, which show the performance of various solitary classification models. They are trained on single datasets by using various algorithms. The recent review articles analysing state-of-the-art classification algorithms [23, 62, 72] report that methods, such as Support Vector Machine (SVM), Naive Bayes (NB), Random Forest (RF), Artificial Neural Network (ANN), Decision Tree (DT), are the most

frequently applied to text classification. Therefore, in the experiments, we took into account these algorithms. In addition, we paid our attention to less known methods, for example, Nearest Shrunken Centroids, Generalized Linear Model, eXtreme Gradient Boosting, Model Averaged Naive Bayes Classifier or Naive Bayes Classifier with Attribute Weighting in our document classification experiments. It was based on our intuition and experience that these tools may produce competitive results.

The experiments allow us to show how differs the performance of our diversified stacked generalisation method from simple classifiers in the task of websites classification. The selected algorithms are enumerated in Table 3. We had available three separate data sets, namely (i) companies description (L_D); (ii) link labels (L_L); (iii) big documents (L_B). They were created from the web pages of companies, as it is described in Section 4.1. We trained classification models on each data set and for each algorithm separately by using the 10-fold cross-validation procedure.

Tables 4, 5 and 6 covers the values of precision, recall, f-measure, error, and accuracy of all eighteen models trained on (L_D), (L_L), (L_B) data sets respectively. The best results produced an Averaged Naive Bayes Classifier irrespectively of a data set. Moreover, a Naive Bayes Classifier with Attribute Weighting gained the best classification quality on the data set containing companies description. We must remember that these are the best models among solitary

Table 3 Algorithms which are used to train simple classifiers in the additional experiments

L.p	Algorithm name	Package name	Method name	Citations
1	k-Nearest Neighbors	base	knn	[26, 56]
2	Support Vector Machines with Linear Kernel	kernlab	svmLinear	[27, 37]
3	Support Vector Machines with Radial Basis Function Kernel	kernlab	svmRadial	[27, 37]
4	Boosted Generalized Linear Model	mboost	glmboost	[12, 28, 29, 31, 32]
5	Nearest Shrunken Centroids	pamr	pam	[25]
6	Optimal Weighted Nearest Neighbor Classifier	snn	ownn	[65]
7	Stabilized Nearest Neighbor Classifier	snn	snn	[65]
8	Random Forest	randomForest	rf	[10, 41]
9	Linear Support Vector Machines with Class Weights	e1071	svmLinearWeights	[46]
10	AdaBoost Classification Trees	fastAdaboost	adaboost	[15]
11	Cost-Sensitive C5.0	C50	C5.0Cost	[38]
12	Generalized Linear Model	glmnet	glmnet	[21, 63]
13	Logistic Model Trees	RWeka	LMT	[30, 69]
14	DeepBoost	deepboost	deepboost	[45]
15	eXtreme Gradient Boosting	xgboost	xgbLinear	[16]
16	Multilayer Perceptron Network with Weight Decay	keras	mlpKerasDecay	[3]
17	Model Averaged Naive Bayes Classifier	bnclassify	manb	[8]
18	Naive Bayes Classifier with Attribute Weighting	bnclassify	awnb	

Each row contains a common name of a particular algorithm; a corresponding package name and a method name in R software; a reference to literature describing the algorithm version which is implemented

Table 4 Classification results on the data set containing companies description (L_D)

Algoorythm name	Feature range	Precision	Recall	Fmeasure	Error	Acc
k-Nearest Neighbors*	600 – 10000	0.894	0.0129	0.0252	0.1835	0.8165
	600 – 4500	0.82	0.0148	0.0291	0.1832	0.8168
Support Vector Machines with Linear Kernel	600 – 12000	0.7328	0.3709	0.4918	0.1417	0.8583
	600 – 4500	0.747	0.3791	0.5012	0.139	0.861
Support Vector Machines with Radial Basis Function Kernel	600 – 12000	0.674	0.5741	0.6199	0.1303	0.8697
	600 – 4500	0.668	0.5527	0.6049	0.1338	0.8662
Boosted Generalized Linear Model	600 – 12000	0.7799	0.2367	0.3632	0.1538	0.8462
	600 – 4500	0.7811	0.2384	0.3653	0.1535	0.8465
Nearest Shrunkn Centroids	600 – 12000	0.6149	0.7024	0.6533	0.1386	0.8614
	600 – 4500	0.564	0.7156	0.6293	0.1558	0.8442
Optimal Weighted Nearest Neighbor Classifier	600 – 12000	0.6969	0.0229	0.0439	0.1836	0.8164
	600 – 4500	0.5818	0.0345	0.0649	0.1841	0.8159
Stabilized Nearest Neighbor Classifier	600 – 12000	0.9163	0.0079	0.0396	0.1843	0.8157
	600 – 4500	0.9163	0.0212	0.0396	0.1827	0.8173
Random Forest*	600 – 2000	0.7708	0.0044	0.0174	0.1847	0.8153
Linear Support Vector Machines with Class Weights	600 – 12000	0.7076	0.4806	0.5713	0.1334	0.8666
	600 – 4500	0.6861	0.5104	0.5852	0.1342	0.8658
AdaBoost Classification Trees*	600 – 1500	0.7007	0.3785	0.4915	0.1451	0.8549
Cost-Sensitive C5.0	600 – 12000	0.7018	0.362	0.4771	0.1469	0.8531
	600 – 4500	0.7117	0.3661	0.4821	0.1453	0.8547
Generalized Linear Model	600 – 12000	0.8213	0.3347	0.4658	0.1366	0.8634
	600 – 4500	0.8329	0.4682	0.5985	0.1158	0.8842
Logistic Model Trees*	600 – 6000	0.6386	0.4655	0.5384	0.1479	0.8521
	600 – 4500	0.6473	0.4719	0.5458	0.1455	0.8545
DeepBoost*	600 – 1500	0.7023	0.353	0.4695	0.1477	0.8523
eXtreme Gradient Boosting*	600 – 1500	0.6935	0.4119	0.5168	0.1427	0.8573
Multilayer Perceptron Network with Weight Decay	600 – 12000	0.2766	0.1793	0.1992	0.2518	0.7482
	600 – 4500	0.2218	0.2362	0.2221	0.2973	0.7027
Model Averaged Naive Bayes Classifier	600 – 12000	0.7121	0.8082	0.7511	0.1017	0.8983
	600 – 4500	0.5962	0.8203	0.6896	0.1369	0.8631
Naive Bayes Classifier with Attribute Weighting	600 – 12000	0.8023	0.6789	0.7202	0.098	0.902
	600 – 4500	0.6545	0.7738	0.7081	0.1189	0.8811

The sign, '**' indicates that we were not able to finish all computations for the whole range of feature, i.e. 600 – 12000. During computations, memory overhead occurred or the time of calculation was too long

Table 5 Classification results on the data set containing link labels (L_L)

Algorithm name	Feature range	Precision	Recall	Fmeasure	Error	Acc
k-Nearest Neighbors*	600 – 10000	0.794	0.0494	0.0875	0.1796	0.8204
	600 – 4500	0.7181	0.0882	0.1523	0.1759	0.8241
Support Vector Machines with Linear Kernel	600 – 12000	0.7694	0.3804	0.5068	0.136	0.864
	600 – 4500	0.7597	0.3907	0.5116	0.1355	0.8645
Support Vector Machines with Radial Basis Function Kernel	600 – 12000	0.6516	0.4887	0.5529	0.1426	0.8574
	600 – 4500	0.6233	0.3846	0.4679	0.1577	0.8423
Boosted Generalized Linear Model	600 – 12000	0.7637	0.213	0.3331	0.158	0.842
	600 – 4500	0.7689	0.2144	0.3352	0.1575	0.8425
Nearest Shrunk Centroids	600 – 12000	0.6967	0.4098	0.5086	0.1447	0.8553
	600 – 4500	0.6284	0.485	0.5472	0.1488	0.8512
Optimal Weighted Nearest Neighbor Classifier	600 – 12000	0.7647	0.1009	0.1638	0.1752	0.8248
	600 – 4500	0.6555	0.1908	0.2921	0.1685	0.8315
Stabilized Nearest Neighbor Classifier	600 – 12000	0.7046	0.0567	0.1753	0.1806	0.8194
	600 – 4500	0.7096	0.134	0.2035	0.1732	0.8268
Random Forest*	600 – 2000	0.8472	0.0103	0.0204	0.1837	0.8163
Linear Support Vector Machines with Class Weights	600 – 12000	0.7857	0.364	0.4876	0.138	0.862
	600 – 4500	0.7304	0.4471	0.5535	0.1334	0.8666
AdaBoost Classification Trees*	600 – 1500	0.6568	0.3569	0.4625	0.1537	0.8463
Cost-Sensitive C5.0	600 – 12000	0.7221	0.3124	0.4337	0.1504	0.8496
	600 – 4500	0.709	0.3211	0.4407	0.1506	0.8494
Generalized Linear Model	600 – 12000	0.8732	0.2936	0.4232	0.14	0.86
	600 – 4500	0.8265	0.4222	0.5579	0.1237	0.8763
Logistic Model Trees*	600 – 6000	0.6552	0.3751	0.4770	0.1524	0.8476
	600 – 4500	0.6605	0.3774	0.4803	0.1513	0.8487
DeepBoost*	600 – 1500	0.7341	0.2731	0.3981	0.153	0.847
eXtreme Gradient Boosting*	600 – 1500	0.6623	0.3582	0.4649	0.1528	0.8473
Multilayer Perceptron Network with Weight Decay	600 – 12000	0.2517	0.2843	0.2564	0.2988	0.7012
	600 – 4500	0.22	0.3231	0.2532	0.3427	0.6573
Model Averaged Naive Bayes Classifier	600 – 12000	0.8266	0.5968	0.6806	0.1036	0.8964
	600 – 4500	0.6893	0.6889	0.6862	0.1174	0.8826
Naive Bayes Classifier with Attribute Weighting	600 – 12000	0.7776	0.5083	0.6035	0.1218	0.8782
	600 – 4500	0.6848	0.6097	0.6389	0.1264	0.8736

The sign, '**' indicates that we were not able to finish all computations for the whole range of feature, i.e. 600 – 12000. During computations, memory overhead occurred or the time of calculation was too long

Table 6 Classification results on the data set containing big documents (L_B)

Algorithm name	Feature range	Precision	Recall	Fmeasure	Error	Acc
k-Nearest Neighbors*	600 – 10000	0.8426	0.018	0.035	0.183	0.817
	600 – 4500	0.8017	0.0205	0.0397	0.1829	0.8171
Support Vector Machines with Linear Kernel	600 – 12000	0.8292	0.3077	0.4465	0.1397	0.8603
	600 – 4500	0.8515	0.3621	0.5056	0.1296	0.8704
Support Vector Machines with Radial Basis Function Kernel	600 – 12000	0.3833	0.0424	0.0734	0.1894	0.8106
	600 – 4500	0.4098	0.0572	0.0935	0.1889	0.8111
Boosted Generalized Linear Model	600 – 12000	0.4881	0.0325	0.0609	0.1857	0.8143
	600 – 4500	0.5738	0.0343	0.0646	0.1836	0.8164
Nearest Shrunk Centroids	600 – 12000	0.8917	0.1864	0.3081	0.155	0.845
	600 – 4500	0.9128	0.181	0.3019	0.155	0.845
Optimal Weighted Nearest Neighbor Classifier	600 – 12000	0.8497	0.0214	0.0415	0.1825	0.8175
	600 – 4500	0.8078	0.0284	0.0544	0.1819	0.8181
Stabilized Nearest Neighbor Classifier	600 – 12000	0.8971	0.0031	0.0358	0.1849	0.8151
	600 – 4500	0.8971	0.0083	0.0358	0.1843	0.8157
Random Forest*	600 – 2000	1	0.0108	0.0212	0.1833	0.8167
Linear Support Vector Machines with Class Weights	600 – 12000	0.8251	0.3373	0.4776	0.1363	0.8637
	600 – 4500	0.8761	0.3456	0.4949	0.1306	0.8694
AdaBoost Classification Trees*	600 – 1500	0.5829	0.2004	0.2982	0.1747	0.8253
Cost-Sensitive C5.0	600 – 12000	0.5445	0.0443	0.0961	0.184	0.816
	600 – 4500	0.5316	0.0517	0.1178	0.1841	0.8159
Generalized Linear Model	600 – 12000	0.9224	0.3496	0.5051	0.1259	0.8741
	600 – 4500	0.9459	0.382	0.5442	0.1186	0.8814
Logistic Model Trees*	600 – 6000	0.6993	0.2547	0.3728	0.1589	0.8411
	600 – 4500	0.7233	0.2602	0.3819	0.1560	0.8440
DeepBoost*	600 – 1500	0.6667	0.0118	0.0232	0.1842	0.8158
eXtreme Gradient Boosting*	600 – 1500	0.5315	0.1493	0.2331	0.1820	0.8170
	600 – 1500	0.5315	0.1493	0.2331	0.1820	0.8170
Multilayer Perceptron Network with Weight Decay	600 – 12000	0.2339	0.2508	0.2342	0.2939	0.7061
	600 – 4500	0.2366	0.2438	0.2365	0.2877	0.7123
Model Averaged Naive Bayes Classifier	600 – 12000	0.9735	0.4352	0.601	0.1069	0.8931
	600 – 4500	0.9684	0.4158	0.5817	0.1108	0.8892
Naive Bayes Classifier with Attribute Weighting	600 – 12000	0.7015	0.2622	0.3611	0.1576	0.8424
	600 – 4500	0.7253	0.4148	0.5072	0.1402	0.8598

The sign, '**' indicates that we were not able to finish all computations for the whole range of feature, i.e. 600 – 12000. During computations, memory overhead occurred or the time of calculation was too long

classifiers trained on a single dataset. If we compare these results with the outcomes of the diversified stacked generalisation method, which is proposed in this study, we can easily notice this method gives classification models that perform better than single models. These results are observable when a features space in this method is either aligned (Table 1) or unaligned (Table 2).

Also, we must note that the time of creating models by each training algorithm was not under our consideration.

However, we noted the computation time of each model, and we present this data in the open data repository¹.

References

1. Adebawale M, Lwin K, Sánchez E, Hossain M (2018) Intelligent web-phishing detection and protection scheme using integrated

¹<http://doi.org/10.5281/zenodo.2537998>

- features of images, frames and text Expert Systems with Applications
2. Aggarwal CC (2018) Machine learning for text springer. <https://doi.org/10.1007/978-3-319-73531-3>
 3. Allaire J, Chollet F keras: R Interface to 'Keras' (2018). <https://CRAN.R-project.org/package=keras>. R package version 2.2.4
 4. Almeida TA, Silva TP, Santos I, Hidalgo JMG (2016) Text normalization and semantic indexing to enhance instant messaging and sms spam filtering. *Knowl-Based Syst* 108:25–32
 5. Asim Y, Shahid AR, Malik AK, Raza B (2017) Significance of machine learning algorithms in professional blogger's classification. *Computers & Electrical Engineering*
 6. Barushka A, Hajek P (2018) Spam filtering using integrated distribution-based balancing approach and regularized deep neural networks. *Appl Intell* 48(10):3538–3556. <https://doi.org/10.1007/s10489-018-1161-y>
 7. Benaim M (2018) From symbolic values to symbolic innovation: Internet-memes and innovation. *Res Policy* 47(5):901–910
 8. Bojan M, Concha B, Pedro L (2018) bncclassify: Learning Discrete Bayesian Network Classifiers from Data. <https://CRAN.R-project.org/package=bncclassify>. R package version 0.4.1
 9. Brattström A, Frishammar J, Richtner A, Pflueger D (2018) Can innovation be measured? a framework of how measurement of innovation engages attention in firms. *Journal of Engineering and Technology Management*
 10. Breiman L, Cutler A (2007) Random forests-classification description. Department of Statistics, Berkeley, vol 2
 11. Brown G (2010) Encyclopedia of Machine Learning, chap. Ensemble Learning. Springer, Boston, pp 312–320
 12. Buehlmann P, Hothorn T (2007) Boosting algorithms: regularization, prediction and model fitting (with discussion). *Stat Sci* 22(4):477–505
 13. Calvo B, Santafe G (2015) scmamp: Statistical comparison of multiple algorithms in multiple problems. *The R Journal Accepted for publication*
 14. Catal C, Nangir M (2017) A sentiment classification model based on multiple classifiers. *Appl Soft Comput* 50:135–141
 15. Chatterjee S (2016) fastAdaboost: a Fast Implementation of Adaboost. <https://CRAN.R-project.org/package=fastAdaboost>. R package version 1.0.0
 16. Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, Chen K, Mitchell R, Cano I, Zhou T, Li M, Xie J, Lin M, Geng Y, Li Y (2018) xgboost: Extreme Gradient Boosting. <https://CRAN.R-project.org/package=xgboost>. R package version 0.71.2
 17. Cortes C, Vapnik V (1995) Support-vector networks. *Mach Learn* 20(3):273–297
 18. Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30. <http://www.jmlr.org/papers/v7/demsar06a.html>
 19. Diab DM, Hindi KME (2017) Using differential evolution for fine tuning naïve bayesian classifiers and its application for text classification. *Appl Soft Comput* 54:183–199. <https://doi.org/10.1016/j.asoc.2016.12.043>
 20. Field A, Miles J, Field Z (2012) Discovering statistics using. R Sage Publications, Thousand Oaks
 21. Friedman J, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models via coordinate descent. *J Stat Softw* 33(1):1–22. <http://www.jstatsoft.org/v33/i01/>
 22. Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J Amer Stat Assoc* 32(200):675–701
 23. Hartmann J, Huppertz J, Schamp C, Heitmann M (2018) Comparing automated text classification methods. *International Journal of Research in Marketing*
 24. Hastie T, Tibshirani R, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer series in statistics. Springer, Berlin. <http://www.worldcat.org/oclc/300478243>
 25. Hastie T, Tibshirani R, Narasimhan B, Chu G (2014) pamr: Pam: prediction analysis for microarrays. <https://CRAN.R-project.org/package=pamr>. R package version 1.55
 26. Hechenbichler K, Schliep K (2004) Weighted k-nearest-neighbor techniques and ordinal classification. In: Discussion paper 399, SFB 386. Ludwig-Maximilians University Munich
 27. Helleputte T LiblineaR: Linear Predictive Models Based on the LIBLINEAR C/C++ Library (2017). R package version 2.10-8
 28. Hofner B, Boccuto L, Goeker M (2015) Controlling false discoveries in high-dimensional situations: Boosting with stability selection *BMC Bioinformatics* 16(144)
 29. Hofner B, Mayr A, Robinsonov N, Schmid M (2014) Model-based boosting in R: a hands-on tutorial using the R package mboost. *Comput Stat* 29:3–35
 30. Hornik K, Buchta C, Zeileis A (2009) Open-source machine learning: R meets Weka. *Comput Stat* 24(2):225–232. <https://doi.org/10.1007/s00180-008-0119-7>
 31. Hothorn T, Buehlmann P, Kneib T, Schmid M, Hofner B (2010) Model-based boosting 2.0. *J Mach Learn Res* 11:2109–2113
 32. Hothorn T, Buehlmann P, Kneib T, Schmid M, Hofner B (2018) mboost: Model-Based Boosting. <https://CRAN.R-project.org/package=mboost>. R package version 2.9-1
 33. Huang JX, He B, Zhao J (2018) Mining authoritative and topical evidence from the blogosphere for improving opinion retrieval. *Information Systems*. <https://doi.org/10.1016/j.is.2018.02.002>, <http://www.sciencedirect.com/science/article/pii/S0306437917302211>
 34. James G, Witten D, Hastie T, Tibshirani R (2014) An introduction to statistical learning: With applications in R. Springer Publishing Company, Incorporated
 35. Jha AK, Bose I (2016) Innovation research in information systems: A commentary on contemporary trends and issues. *Inf Manag* 53(3):297–306. <https://doi.org/10.1016/j.im.2015.10.007>, <http://www.sciencedirect.com/science/article/pii/S0378720615001238>. *Information Technology and Innovation: Drivers, Challenges and Impacts*
 36. Kahn KB (2018) Understanding innovation. *Bus Horizons* 61(3):453–460. <https://doi.org/10.1016/j.bushor.2018.01.011>, <http://www.sciencedirect.com/science/article/pii/S0007681318300119>
 37. Karatzoglou A, Smola A, Hornik K, Zeileis A (2004) kernlab – an S4 package for kernel methods in R. *J Stat Softw* 11(9):1–20. <http://www.jstatsoft.org/v11/i09/>
 38. Kuhn M, Quinlan R (2018) C50: C5.0 Decision Trees and Rule-Based Models. <https://CRAN.R-project.org/package=C50>. R package version 0.1.2
 39. Kumar BS, Ravi V (2016) A survey of the applications of text mining in financial domain. *Knowl-Based Syst* 114:128–147
 40. Kuncheva LI (2014) Combining pattern classifiers: methods and algorithms, 2nd edn. Wiley, New York
 41. Liaw A, Wiener M (2002) Classification and regression by randomforest. *R News* 2(3):18–22. <https://CRAN.R-project.org/doc/Rnews/>
 42. Lochter JV, Zanetti RF, Reller D, Almeida TA (2016) Short text opinion detection using ensemble of classifiers and semantic indexing. *Expert Syst Appl* 62:243–249
 43. Lokuge S, Sedera D, Grover V, Dongming X (2018) Organizational readiness for digital innovation: Development and empirical calibration of a construct. *Information & Management*. <https://doi.org/10.1016/j.im.2018.09.001>, <http://www.sciencedirect.com/science/article/pii/S0378720616303111>

44. Manning CD, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press, New York
45. Marcous D, Sandbank Y (2017) deepboost: Deep Boosting Ensemble Modeling. <https://CRAN.R-project.org/package=deepboost>. R package version 0.1.6
46. Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2018) e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien. <https://CRAN.R-project.org/package=e1071>. R package version 1.7-0
47. Mironczuk M, Protasiewicz J (2016) A Diversified Classification Committee for Recognition of Innovative Internet Domains. Springer International Publishing, Cham, pp 368–383
48. Mortensen PS, Bloch CW et al (2005) Oslo Manual-Guidelines for collecting and interpreting innovation data: Proposed guidelines for collecting and interpreting innovation data. Organisation for economic cooperation and development OECD
49. Myles H, Douglas AW, Eric C (2014) Nonparametric statistical methods, 3rd edn. Wiley, New York
50. Nakatsuji M, Miyoshi Y, Otsuka Y (2006) Innovation Detection Based on User-Interest Ontology of Blog Community. Springer, Berlin, pp 515–528
51. Nakatsuji M, Yoshida M, Ishida T (2009) Detecting innovative topics based on user-interest ontology. Web Semant Sci Serv Agents World Wide Web 7(2):107–120
52. Obied A, Alhajj R (2009) Fraudulent and malicious sites on the web. Appl Intell 30(2):112–120
53. Pilav-Velić A, Marjanovic O (2016) Integrating open innovation and business process innovation: Insights from a large-scale study on a transition economy. Inf Manag 53(3):398–408. <https://doi.org/10.1016/j.im.2015.12.004>. <http://www.sciencedirect.com/science/article/pii/S0378720615001433>. Information Technology and Innovation: Drivers, Challenges and Impacts
54. Protasiewicz J, Mironczuk M, Dadas S (2017) Categorization of multilingual scientific documents by a compound classification system. In: International conference on artificial intelligence and soft computing. Springer, pp 563–573
55. R Core Team (2016) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna. <https://www.R-project.org/>
56. R Core Team (2018) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna. <https://www.R-project.org/>
57. Robertson SE, Walker S, Jones S, Hancock-beaulieu M, Gatford M (1994) Okapi at TREC-3. In: TREC, pp 109–126
58. Rokach L (2010) Ensemble-based classifiers. Artif Intell Rev 33(1-2):1–39
59. Salkind N (2010) Encyclopaedia of Research Design, vol 1. Sage Publications, Thousand Oaks
60. Sammut, C, Webb, GI (eds.) (2017) Stacked Generalization. Springer, Boston. pp 1173–1173, https://doi.org/10.1007/978-1-4899-7687-1_969
61. Santafe G, Inza I, Lozano JA (2015) Dealing with the evaluation of supervised classification algorithms. Artif Intell Rev 44(4):467–508. <https://doi.org/10.1007/s10462-015-9433-y>
62. Shaikh GM, Shuib NLM, Idris N, Hoo WL, Raj RG, Khowaja K, Shaikh K, Nweke HF (2019) Clinical text classification research trends: Systematic literature review and open issues. Expert Syst Appl 116:494–520. <https://doi.org/10.1016/j.eswa.2018.09.034>
63. Simon N, Friedman J, Hastie T, Tibshirani R (2011) Regularization paths for cox's proportional hazards model via coordinate descent. J Stat Softw 39(5):1–13. <http://www.jstatsoft.org/v39/i05/>
64. Sokolova M, Lapalme G (2009) A systematic analysis of performance measures for classification tasks. Inf Process Manag 45(4):427–437
65. Sun W, Qiao X, Cheng G (2015) snn: Stabilized Nearest Neighbor Classifier. <https://CRAN.R-project.org/package=snn>. R package version 1.1
66. Ting KM, Witten IH (1997) Stacked generalization: when does it work?. In: Proceedings of International Joint Conference on Artificial Intelligence. Morgan Kaufmann, pp 866–871
67. Ting KM, Witten IH (1999) Issues in stacked generalization. J Artif Intell Res 10:271–289
68. Wang Z, Gu S, Xu X (2018) GSLDA: LDA-based group spamming detection in product reviews. Appl Intell 48(9):3094–3107. <https://doi.org/10.1007/s10489-018-1142-1>
69. Witten IH, Frank E (2005) Data mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco
70. Wolpert DH (1992) Stacked generalization. Neural Netw 5:241–259
71. Zach FJ, Hill T (2017) Network, knowledge and relationship impacts on innovation in tourism destinations. Tour Manag 62:196–207
72. Zhang C, Liu C, Zhang X, Almpandis G (2017) An up-to-date comparison of state-of-the-art classification algorithms. Expert Syst Appl 82:128–150. <https://doi.org/10.1016/j.eswa.2017.04.003>
73. Zhang C, Ma Y (2012) Ensemble machine learning: methods and applications. Springer, Berlin
74. Zhang D, Yan Z, Jiang H, Kim T (2014) A domain-feature enhanced classification model for the detection of chinese phishing e-business websites. Inf Manag 51(7):845–853. <https://doi.org/10.1016/j.im.2014.08.003>
75. Zhang W, Jiang Q, Chen L, Li C (2017) Two-stage elm for phishing web pages detection using hybrid features. World Wide Web 20(4):797–813

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Marcin Michał Mirończuk (PhD) is a graduate of Białystok Technical University, Faculty of Electrical Engineering in Poland where he received MSc degree, in 2007. The PhD degree received in Białystok Technical University, Faculty of Computer Science, in 2013. Currently, he is working in National Information Processing Institute - Laboratory of Intelligent Information Systems, Warsaw, Poland. His research interests are in the areas of applied

mathematics, data/ text mining techniques and information extraction.

Jarosław Protasiewicz (PhD) is an assistant professor at the National Information Processing Institute and the head of the Laboratory of Intelligent Systems. He received a Ph.D. in computer science at the Systems Research Institute of the Polish Academy of Sciences. His areas of interest include agile project management, software design and development, big data, machine learning, and bio-inspired algorithms.