Text mining and machine learning techniques for text classification, with application to the automatic categorization of websites

Gianpiero Bianchi[1], Renato Bruni[2], Francesco Scalfati[1], Fabiana Bianchi[1]

1) Istat, Direzione centrale per la metodologia e disegno dei processi statistici (DCME), Via Depretis 77, Roma, 00184 Italy
E-mail: {gianbia,scalfati,fabianchi}@istat.it

2) Università di Roma "Sapienza", Dip. di Ing. Informatica, Automatica e Gestionale (DIAG), Via Ariosto 25, Roma, 00185 Italy
E-mail: bruni@dis.uniroma1.it

Abstract

Website categorization has recently emerged as a very important task in several contexts. A huge amount of information is freely available through websites, and it could for instance be used to accomplish statistical surveys. However, the information of interest for the specific task under consideration has to be mined among that huge amount, and this turns out to be a difficult operation in practice. This work describes techniques that can be used to convert website categorization into a supervised classification problem. Each data record should summarize the content of an entire website. Records are obtained by using web scraping procedures, followed by a number of feature extraction and selection steps. When such records are completed, we apply state of the art classification techniques to categorize the websites according to the aspect of interest. Since in many applicative cases the labels available for the training set may be noisy, we also analyze the robustness of our procedure with respect to this issue. We present results obtained on real-world data for the detection of websites providing e-commerce facilities.

Keywords: Classification; Big Data; Machine Learning; Feature Selection; Text Mining

1 Introduction

Text Mining is the branch of Data Mining concerning the process of deriving high-quality information from texts. References can be found for instance in [10]. This area underwent considerable improvements in recent years, with a number of concurrent factors contributing to its progress, first of all the continuous expansion of the Internet and the demand for effective automatic search and manipulation strategies. Modern text mining techniques require the integration of natural language processing operations (see, e.g., [4]) with several advanced machine learning techniques (see, e.g., [12, 14]).

A very relevant problem in this field is the classification of text documents. This consists in using a set of text documents, each having a class label, to learn a *classifier*. The classifier is then used to automatically assign the class label to new unlabeled text documents. This task is required in a large variety of practical applications, in particular when the considered text documents are websites. In this case, the task is also called *Website categorization*.

The number of web pages available on the Internet has constantly been increasing in the last decades, and nowadays a huge amount of data is freely available through this channel. Indeed, the sheer size of the problem makes implausible a non-automatic intervention. On the other hand, the automatic extraction of statistical information from this source is extremely appealing, because it would produce large datasets with considerable savings, or it would provide a way to check or integrate datasets that are already available, increasing their quality.

However, website categorization turns out to be a very difficult task in practice, for several reasons. First, the classification problem has a very large dimension: the data records representing the websites generally have thousands of fields, and there are thousands of records. It is well known that such a dimensionality increase usually raises considerably the difficulty of the classification task. Secondly, the content of the generic website, consisting of thousands of pages in some cases, should be summarized by the corresponding data record. Hence, these records should be automatically generated. This is a complex process that requires the use of different Web Scraping and Text Mining phases. Web sites are of course not standardized, part of the information of a website is provided to the human users by means of the graphics rather than the text, etc. Thirdly, effective feature selection techniques are required, since these automatically produced data inevitably contain a very large amount of noise. This preponderant noise content should be reduced as much as possible to successfully apply the classification techniques. Therefore, to obtain a satisfactory performance in the classification phase, quite articulated procedures have to be developed. An extensive survey on text mining techniques is in [22], while the specific case of web pages is surveyed in [18]. Previous attempts in solving the same type of problem are described in [2, 3].

In this work, we propose an overall strategy to perform websites categorization. We apply this strategy to the specific problem of the detection of websites providing e-commerce facilities. In particular, we want to determine whether the generic website allows to buy, or at least to order, goods or services, or not. We analyze the performance and the robustness of our approach using real-world data.

Roughly speaking, we proceed as follows. Given a list of websites, for each of them, we take the text extracted from the website by means of automatic scraping. Also, we download images and screen-shots and we process them with optical character recognition (OCR) [24] to extract additional information provided by the graphics. We use this material to prepare very large text files. Subsequently, we perform several steps in order to identify and select only the relevant parts of the above information, and to exclude the noisy part as much as possible. This is done by using natural language processing techniques, such as tokenization, lemmatization and part-of-speech recognition [21], until we obtain, for each website, a number of relevant words, n-grams and additional features. After this, we apply term evaluation techniques to reduce the dimensions and obtain a set of standardized data records describing the above websites.

Finally, we classify the obtained records by using state-of-the-art classification algorithms [17]. In particular, we use Support Vector Machines [7], Random Forest [5], Logistic classifiers [11]. Each of these classifiers requires to set some algorithmic parameters, which greatly affect the result of the classification phase. We formulate the problem of the parameters' choice as follows. We chose the parameters which maximize the harmonic mean of precision and sensitivity of the classification produced. This type of accuracy measure is called F1-score [23].

Note that the classification paradigm is based on the availability of a set of labeled records, called training set, that constitute the source of information to learn a classifier. Therefore, to apply the described approach for website categorization, we need a set of websites for which we already have, or we can obtain, the class labels with respect to the considered categorization. In practical cases, such class labels may easily contain some errors, due to many reasons. For example, labels may have been assigned to websites by several human operators which follow slightly different labeling criteria, or which may be

mistaken on ambiguous cases; labels may have been assigned by automatic procedures which have difficulties on complex cases; labels may simply be outdated; etc. Hence, in practical cases, the training set physiologically contains a certain portion of misclassified records. As a consequence, website categorization techniques should have some degree of robustness with respect to the presence of noise in the class label. Hence, we also analyze the robustness of our categorization procedure, by perturbing at several intensity levels the class labels of the training set. We obtain very encouraging results, and we discuss on possible motivations of the observed robustness.

Therefore, the main contributions of this work is the presentation of a practically viable technique to perform automatic categorization of websites in order to attain satisfactory accuracy and robustness. The paper is organized as follows. Section 2 describes all the operations that have been used to convert the generic website into a data record of reasonable size summarizing that entire website. Section 3 provides a brief overview of the classification algorithms that have been used for our analysis. Section 4 reports the results of experiments by considering websites belonging to enterprises operating in Italy.

2 The Text Mining Phase

We begin with a list L of websites corresponding to commercial companies. For each $l_i \in L$, we use a web scraping procedure that reads and saves the content of the website, that is the text appearing in its pages. It starts from the homepage and continues with all the other pages reachable form it, up to a certain depth, that can be selected. The underlying idea is that the pages that are too nested are less relevant for the analysis, while they would mainly introduce a large amount of noise. On the other hand, besides the text appearing in the pages, we read some additional information: the attributes of HTML elements, the name of the image files, the keywords of the pages.

Moreover, we perform Optical Character Recognition (OCR) on all the types of images that appear in the pages processed by the scraping procedure, in order to read also the words provided as images. In fact, these words are often written with such a special emphasis because they are particularly relevant (consider for instance the case of logos, 'buy' and 'pay' commands, etc.). Similarly, we take screen-shots of the homepages and perform OCR on them, too. Indeed, many websites try to catch the eye on the key sentences of the homepage by adopting unusual writing techniques or symbols. OCR is achieved by using the Tesseract Open Source OCR Engine, initially developed by Hewlett Packard Research before 1995 and subsequently by Google Research since 2006 [24]. The regions of the images containing text have been located by using the Marvin image processing framework, developed in java [1].

The above operations produce, for each l_i, a text file d_i. The set of all such text files is D. These text files are very large, they may contain more than 1 million words. Unfortunately, for almost all non-trivial categorizations that may be required in practice, the overwhelming majority of this information is simply irrelevant. Therefore, we need to identify and select only the part of the above information that is relevant for the categorization required, and to exclude the remaining parts as much as possible.

To perform such a selection, we use the following steps. We initially clean the text by removing all non-alphabetic symbols (e.g. $\%, \&, =$, etc.), and by inserting white spaces to detach the words (tokenization). Then, we remove the stop-words (articles, prepositions, etc.), since their generic meaning has practically no relevance for the categorization task. Subsequently, we identify a training set S composed of 50% of the elements in D. The elements in S should have the class label for the categorization under analysis. When this information is not already available, it must be assigned by some ad hoc procedure or by manual intervention. Evidently, in practical cases, S may very easily contain misclassified elements. Hence, the class label is often noisy. Moreover, the dataset may be imbalanced, in the sense that the proportions of the records in the different classes are not comparable. For example, in the case of the detection of e-commerce, negative records, i.e., those not

providing e-commerce facilities, constitute about 80% of D.

After this, we extract from S two dictionaries: the set W of the uni-grams, i.e., the single words, appearing in those files, and the set Z of the n-grams appearing in the same files. N-grams are sequences of n adjacent words that are typically used together. An example is "credit card", which is a bi-gram, i.e., has $n = 2$.

For the set of uni-grams W, we perform lemmatization, that is, the inflectional ending of each word is removed in order to return the word to its basic lemma. This operation allows to group together the different inflected forms of a word (e.g., plurals of nouns, tenses of verbs, etc.) so they can be analyzed as a single item. Since we are working with websites of enterprises operating in Italy, this is done for Italian and English language. All words not belonging to these two languages are discarded. Clearly, the choice of the languages can be changed without affecting the fundamental aspects of our approach. Lemmatization is performed by using the software TreeTagger. This tool was initially described in [21] and it was subsequently developed by several contributors.

For the set of n-grams Z, we do not perform lemmatization, since substituting words with their basic lemmas may result in losing the identity of many n-grams, which are generally built with specific inflectional forms. On the other hand, for Z we do perform part-of-speech recognition (POS tagging), that is, each word is identified as a particular part of speech (e.g., a noun, a verb, etc.). This is done again in Italian and English language, and again with the software TreeTagger. All words not belonging to these two languages are discarded. Moreover, we discard all n-grams that cannot represent meaningful concepts. For example, in the case of bi-grams, we keep only the pairs that are syntactically well-composed. This means they must be composed by: noun and verb; noun and adjective; noun and adverb; noun and noun; verb and adjective; verb and adverb.

Thus, we obtain at this stage the following two sets of terms:

1. Set W', whose component terms are single lemmas in Italian or English language.
2. Set Z', whose component terms are syntactically well-composed n-grams in Italian or English language.

Now, for each of the terms of W' and Z', we must compute a measure of its relevance for the categorization under analysis by means of a *Term Evaluation* (TE) function. There exist several possible TE functions, representing several metrics; for example Chi Square, Information Gain (also known as Mutual Information), Gain Ratio, etc. [8].

We have selected for our experiments the so-called Chi-square metric (χ^2), since it appears appropriate and it is one of the most frequently used in text categorization. Indeed, χ^2 statistics is often used in many fields to measure how the results of an observation differ from the results expected according to an initial hypothesis [16]. In our case, we make the hypothesis of dependence between the generic term $w \in W'$ and the class (positive or negative) of the generic file d containing w, and thus we measure the dependence of w from the class, with lower values corresponding to lower dependence. Since we have two classes, for each $w \in W'$, we compute a score $s(w) = \chi^2_+(w) + \chi^2_-(w)$, where χ^2_+ is called positive score and χ^2_- negative score. The positive score is defined as follows:

$$\chi^2_+ = \frac{p(p_{11}p_{22} - p_{12}p_{21})^2}{(p_{11} + p_{12})(p_{21} + p_{22})(p_{11} + p_{21})(p_{12} + p_{22})},$$

where p_{11} is the number of occurrences of w in positive files; p_{12} is the total number of occurrences of w; p_{21} is the number of all distinct words occurring in positive files; p_{22} is the total number of all distinct words; and $p = p_{11} + p_{12} + p_{21} + p_{22}$. The negative score is defined similarly, except that p_{11} becomes the number of occurrences of w in negative files and p_{21} the number of all distinct words occurring in negative files.

Similarly, for any n-gram $z \in Z'$, we compute a score $s(z) = \chi^2_+(z) + \chi^2_-(z)$, where χ^2_+ is the positive score and χ^2_- the negative score. These scores are again based on the described Chi-square metric, and the basic idea is now to measure the dependence between

the presence of the words constituting the generic $z \in Z'$ and the class of the generic file d containing z. Assuming z is a bi-gram, the positive score is defined as follows:

$$\chi^2_+ = \frac{q(q_{11}q_{22} - q_{12}q_{21})^2}{(q_{11}+q_{12})(q_{21}+q_{22})(q_{11}+q_{21})(q_{12}+q_{22})},$$

where q_{11} is the number of positive files containing all the words constituting z; q_{12} is the number of positive files containing only the first word of z; q_{21} is the number of positive files containing only the second word of z; q_{22} is the number of positive files not containing any of the words constituting z; and $q = q_{11} + q_{12} + q_{21} + q_{22}$. The negative score is defined similarly, except that all the above values are computed for negative files. In case z has $n \geq 3$, the above formula is consequently expanded. We extract n-grams using all the values of n from 2 to 5. In case an n-gram with n words is fully contained in a larger n-gram with $(n+1), \ldots, 5$ words, we remove the larger n-gram. This because we assume the presence of the shorter n-gram more significant than that of the larger one. We observe that, in our experiment, this practically leads to using mainly bi-grams. However, this reveals not to be a limitation, and it also provides computational advantages.

After this, all terms in W' and in Z' are sorted by decreasing score values, and we finally select among them the terms constituting the relevant information. We take from W' all the terms with a TE score larger than a threshold τ_w and up to a maximum of α_w terms. Similarly, we take from Z' all the terms with a TE score larger than τ_z and up to a maximum of α_z terms. Generally, we set these parameters in order to obtain a set T of about 1000 terms, where the uni-gram terms are about 800 and the n-gram terms are the remaining part. Now, for each $d_i \in D$, we project it on the set T, that is, we reduce d_i to a binary vector r_i of size $|T|$. The h-th element of r_i will be denoted by r_i^h and it is computed as follows:

$$r_i^h = \begin{cases} 1 & \text{if the } h\text{−th term in } T \text{ is present in } d_i \\ 0 & \text{otherwise.} \end{cases}$$

Vector r_i constitutes the data record summarizing d_i, as anticipated above. The set of all records is R. The class c_i of $r_i \in R$, when available, is

$$c_i = \begin{cases} 1 & \text{if the } i\text{−th website in } L \text{ offers e−commerce} \\ 0 & \text{otherwise.} \end{cases}$$

3 The Classification Phase

Many different classification approaches have been proposed in the literature, based on different data models and mathematical techniques. It is well known that there is not a single approach capable to outperform all the others on every instance of the classification problem. However, given a specific category of problems, it is empirically possible to identify which approaches generally provide the best performances for that category. For our category of problems, we have performed preliminary tests with several classifiers by means of scikit learn [17], that is a machine learning package currently included into scientific Phyton distributions. The best results in these preliminary tests have been obtained with:

- Support Vector Machines;
- Random Forests;
- Logistic classifiers.

Hence, we have selected these three classifiers for our full experiments. For each record r_i, the fields r_i^h corresponding to the terms constitute the input or independent variables; the class c_i constitutes the target or dependent variable.

Support Vector Machines Support Vector Machines (SVMs) are supervised learning models that build a deterministic linear classifier. They are based on finding a separating hyperplane that maximizes the margin between the extreme training data of opposite classes. New examples are then mapped into that same space and predicted to belong to a class, on the basis of which side of the hyperplane they fall on. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, by implicitly mapping their inputs to a higher dimensional space, see also [7, 25]. This classifier requires to set some algorithmic parameters, in particular when using as kernel a Gaussian radial basis function. The main parameters in this case are the penalty parameter c of the error term and the kernel coefficient γ. They greatly affect the result of the classification, In particular, we chose the combination of values $(\bar{c}, \bar{\gamma})$ which maximizes the harmonic mean of precision and sensitivity of the classification produced, called F1-score [23]. By denoting with TP the number of true positive records produced, by TN that of true negatives, by FP that of false positives and by FN that of false negatives, this means:

$$(\bar{c}, \bar{\gamma}) = \arg\max_{(c,\gamma)} \frac{200\ TP}{2TP + FP + FN}. \tag{1}$$

We solve this minimization problem by using a grid search approach, see also [7] for details. The tolerance for the stopping criterion in SVM was set at the default value of 0.001; the maximum number of iterations at 1000.

Random forests Decision trees are a supervised learning model that maps observations about the input variables to conclusions about the target variable. The goal is to create a decision tree that predicts the value of the target variable based on combinations of the values of the input variables. Each internal node is associated with a decision concerning the value of an input variable that best splits the training set. Different algorithms can be used to determine the input variables associated with the internal nodes, see also [19]. This methodology is generally quite effective and computationally light, however it often exhibits a tendency to overfit the training set. This means that the model produced by the classifier may become unnecessarily complex in the attempt to excessively fit the peculiarities of the available data. To overcome similar problems, *Ensemble* techniques have been developed. Roughly speaking, those techniques generate many weak learners and combine their outputs in order to obtain a classification that is generally both accurate and robust.

In particular, Random Forest (RF) is an ensemble learning method that operates by generating a multitude of decision trees. The global output is obtained by computing the mode of the outputs of the individual trees. Additional details can be found in [13, 5] Random forests are generally more robust and can achieve better performances than the single decision trees. For this reason, we use such a version of the decision tree methodology in our experiments. The number of trees used in our experiments has been set at 500.

Logistic classifiers Logistic regression is a regression model where the target variable is categorical; hence, it can be used to perform a classification. This approach is called Logistic Classifier (LC). It measures the relationship between the target variable and one or more independent variables by estimating the probabilities using a logistic function, which is the cumulative logistic distribution, see also [11]. Logistic regression can be seen as a special case of the generalized linear model and thus analogous to linear regression. This approach is often used in practice because it is computationally light and it possesses a fair power of generalization.

We use the Python implementations of the above three classifiers that are available in the scikit-learn package through the functions: SVC(); RandomForestClassifier(); LogisticRegression().

4 Experimental Results

We apply the described procedure to a list L of 4,755 websites belonging to enterprises operating in Italy. Each item in L originally had a class value, although, as observed, this information may be noisy. By applying the techniques described in Section 2, we produce a set R of 4,755 records. Each $r_i \in R$ has 1000 fields, 800 obtained from unigrams and 200 from n-grams as described in Section 2, and the class label c_i. A record of this type is positive if the corresponding website offers e-commerce facilities, and it is negative otherwise.

Since only about 20% of the entires are positive, the dataset is very imbalanced. This increases the difficulty of the classification phase. Indeed, it is very easy to reach an 80% of classification accuracy by simply predicting all records as negative. However, this result would be completely useless from the practical point of view. In fact, obtaining the correct identification of the positive records constitutes the main goal in this type of problems, and this identification is particularly challenging.

To remove the noise in the class, the class labels have been interactively checked by human intervention. This operation led to reverse about 8% of positive records into negative ones, and about 3.33% of negative records into positive ones. The fact that the class noise is concentrated more on positive records has been frequently observed on our instances of the problem. After this, we obtain a set of records R' with correct class labels.

To perform the classification task, we select in R' a training set S of 2,377 records, that is 50% of the total dataset. The extraction have been randomly performed 3 times, and all performance results are averaged on the 3 trials. Given a training set S, we apply to it a random perturbation in the class label, in order to simulate the presence of the above described class noise. This is done at different levels of intensity and with the three following proportions in the distribution of the class errors:

1. *Observed error proportion.* We introduce errors in the class using an error proportion 2.4 times larger for positive records than for negative ones, as it was the error observed in our dataset.

2. *Balanced error proportion.* We introduce exactly the same number of class errors in reach class, independently of the size of each class.

3. *Uniform error proportion.* We introduce errors in the class using an error proportion that reflects the size of each class, so that the frequency of errors is the same all over the dataset. In our case, roughly 20% of the entries are positive, and the remaining 80% of the entries are negative. Therefore, the uniform error proportion has a number of errors on negative records that is about 4 times larger.

Therefore, we obtain 8 x 3 = 24 versions of the training set, denoted by S_h^k. Index $h = 1, \ldots, 8$ represents the perturbation level, from 0% (not perturbed at all) to 21% (a strong perturbation); index $k = 1, \ldots, 3$ represents the perturbation model: *observed, balanced, uniform*. The actual number of training records whose class have been changed to obtain each S_h^k is reported in Table 1 below. Note that the total number of positive records in S is only 432; hence some of the highest perturbation levels reverse the class of more than half of the positive training records. This means that the information provided to the classifier to be able to predict the positive class has been strongly altered.

After this, we perform the training phase of the three classifiers (RF, SVM, LC) described in Section 3 for each of the above 24 training sets S_h^k. The objective in the training phase is the maximization of F1, as shown in equation (1). This is obtained for RF and LC in less than 5 minutes of computation for each training set S_h^k. SVM classifier, on the other hand, requires a more elaborate training phase. We perform a grid search to find the best parameters, requiring about 20 minutes for each training set S_h^k and using 3-fold cross-validation. Note that the use of such an approach is standard for training SVM in practical applications. Though it cannot theoretically guarantee to determine exactly the

optimal parameters, it is generally regarded as the more reasonable compromise between time and performance.

Table 1: Number of perturbed training records in each perturbation scheme.

	Observed		Balanced		Uniform	
Total Perturbation	pos.	neg.	pos.	neg.	pos.	neg.
3%	50	21	36	36	13	58
6%	101	42	71	71	26	117
9%	151	63	107	107	39	175
12%	201	84	143	143	51	234
15%	252	105	178	178	62	292
18%	302	126	214	214	77	351
21%	352	147	250	250	90	409

When the training phase is completed, we use the learned classifiers to predict the class for all the records in the test sets $T = R' - S$. Subsequently, by knowing the real class of each $r_i \in T$, we compare it to the predicted class so we can compute the confusion matrix corresponding to each classifier and each training set S_h^k. The elements of each confusion matrix (true positives TP, false negatives FN, true negatives TN, false positives FP) are then used to evaluate the following performance measures:

- Accuracy a, defined as the percentage of correct predictions over all predictions:
$$a = \frac{100(TP + TN)}{TP + FN + TN + FP}.$$
- Precision p, also called the positive predictive value, defined as the percentage of true positive records in all positive predictions: $p = \frac{100\ TP}{TP + FP}$.
- Sensitivity s, also called the true positive rate, defined as the percentage of correct positive predictions in all real positive records: $s = \frac{100\ TP}{TP + FN}$.
- F1-score, which is the harmonic mean of the above described measures of precision and sensitivity:
$$F_1 = \frac{200\ TP}{2TP + FP + FN}.$$

Note that, for the detection of e-commerce, the latter one appears the most relevant performance measure, since it fully evaluates the correct identification of the positive records, that is the most important and difficult task. Therefore, in our experiments, besides the basic measure of accuracy, we consider the F1-score.

Table 2 compares the results of the three described classifiers using the correct training set S_1^1 and all the training sets perturbed with the observed error proportion $S_2^1 \dots S_8^1$. Table 3 compares the same information but using the training sets perturbed with the balanced error proportion $S_2^2 \dots S_8^2$. Finally, Table 4 compares the same information but using the training sets perturbed with the uniform error proportion $S_2^3 \dots S_8^3$.
By analyzing the above results, we observe what follows.

1. The classification performance obviously degrades by increasing the perturbation level. However, this degradation is not so marked as it could be expected. Indeed, up to a perturbation level of 15%, the degradation is less than proportional to the perturbation introduced. Also, in some cases, one step of increase in the perturbation level does not worsen the classification performance. In other words, the whole

Table 2: Results obtained when perturbing with the observed proportion.

	RF		SVM		LC	
Perturbation level	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
0%	90.45 %	73.51 %	88.81 %	70.31 %	87.76 %	65.86 %
3%	90.66 %	73.25 %	87.94 %	68.80 %	87.35 %	63.90 %
6%	90.15 %	70.97 %	86.63 %	64.68 %	87.30 %	62.37 %
9%	90.2 %	70.39 %	84.63 %	59.68 %	86.96 %	60.46 %
12%	89.91 %	68.50 %	82.93 %	56.21 %	87.59 %	60.32 %
15%	86.92 %	57.92 %	76.79 %	53.77 %	86.50 %	56.44 %
18%	84.57 %	48.96 %	71.86 %	49.15 %	85.87 %	52.94 %
21%	84.74 %	47.47 %	70.67 %	47.31 %	85.79 %	51.01 %

Table 3: Results obtained using balanced perturbation.

	RF		SVM		LC	
Perturbation level	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
0%	90.45 %	73.51 %	88.81 %	70.31 %	87.76 %	65.86 %
3%	90.19 %	72.33 %	88.18 %	68.95 %	87.34 %	64.71 %
6%	89.86 %	71.75 %	85.75 %	65.83 %	86.63 %	63.70 %
9%	89.49 %	70.79 %	86.54 %	64.04 %	86.03 %	62.10 %
12%	88.77 %	68.77 %	85.67 %	57.64 %	86.03 %	60.43 %
15%	85.79 %	60.42 %	80.12 %	55.04 %	85.37 %	60.27 %
18%	84.82 %	57.88 %	78.20 %	48.89 %	86.96 %	56.70 %
21%	80.91 %	47.96 %	75.15 %	47.08 %	84.18 %	55.97 %

procedure appears to possess a certain degree of robustness with respect to the presence of class errors in the training set. We hypothesize that this robustness is due to the use of the automatic classification algorithms. Indeed, it is well known that the capability of a classifier to generalize what is learned from the training set is strictly correlated with its ability to search for a "simple" model of the data, according to the so-called Occam's razor principle, see also [9]. In our experiments, by reversing the class of some training records, we are actually providing a limited amount of wrong information, carried by the records that have been perturbed, mixed with the correct information carried by the unperturbed records. The simplification ability of the automatic classifiers allows to override this amount of wrong information, at least until it remains a minoritary part of the whole information provided.

2. Data perturbed with the uniform model produce the best F1 performance, while those perturbed with the observed model produce the worst F1 performance. This holds because, for each given perturbation level, the observed model is the one that corrupts the largest amount of positive records, so their correct detection becomes more difficult. On the contrary, the uniform model is the one that corrupts the smallest amount of positive records, so the effect is the opposite of the former case.

3. Despite the intrinsic robustness observed in the classification approach, we have to note that there exists a kind of threshold effect. Indeed, when the perturbation level goes beyond 12 %, the performance degrades more sharply, since the amount of wrong information becomes consistent and it starts to cause a sort of "avalanche

Table 4: Results obtained using uniform perturbation.

	RF		SVM		LC	
Perturbation level	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
0%	90.45 %	73.11 %	88.81 %	70.31 %	87.76 %	65.86 %
3%	89.66 %	72.67 %	86.24 %	68.20 %	85.70 %	63.98 %
6%	89.02 %	72.38 %	85.83 %	65.44 %	85.37 %	63.04 %
9%	87.13 %	69.09 %	76.32 %	62.57 %	84.61 %	63.83 %
12%	86.99 %	68.71 %	73.43 %	58.69 %	83.05 %	62.07 %
15%	83.60 %	64.09 %	73.30 %	57.07 %	82.42 %	60.87 %
18%	80.07 %	57.99 %	69.28 %	49.69 %	79.48 %	57.97 %
21%	79.39 %	56.33 %	68.61 %	49.69 %	78.85 %	57.50 %

effect". This especially holds for the dataset perturbed with the observed model, for the reasons specified in the previous observation.

4. Random Forest classifier (RF) generally provides the best performances in our experiments. However, the results of the other two classifiers (SVM, LC) are not considerably worse, and in any case the trend is very similar. Hence, the overall classification results can be considered as to be quite aligned. As known, in practical applications, any generic dataset has its inherent "level of difficulty", in the sense that in many cases the classification accuracy cannot be pushed up to 100%, no matter which classifier and parameters' combination is used. In other words, if we operate a bad choice for classifier/parameters, we may be able to worsen the results at will, but, on the other extreme, there exist a sort of upper limit in the classification performance that is obtainable on a dataset. Coming back to our case, the fact that the classification results are quite aligned allows us to deem that, after the careful parameters' optimization and the learning phase, the performance of the classification produced by our classifiers is not far from to the upper limit in the performance obtainable by a generic automatic classification strategy on the considered dataset.

To provide further insight on the robustness of the procedure, we also report the graphs of the decrease of the performance obtained when increasing the perturbation level. In particular, Fig. 1 reports the analysis of the accuracy of RF classifier; Fig. 2 reports the analysis of the F1 score of the same classifier; Fig. 3 reports the analysis of the accuracy of SVM classifier; Fig. 4 reports the analysis of the F1 score of the same classifier; Fig. 5 reports the analysis of the accuracy of LC; Fig. 6 reports the analysis of the F1 score of the same classifier. These figures allow to fully observe the evolution of the degradation in the classification performance, confirming the observations reported above.

The computational times and the memory usage of the whole procedures are quite reasonable, considering also the large size of the problem. In particular, using a PC with i7 processor and 16GB RAM, we experienced what follows. Given the set of text files D, the text mining operations which produce the set of records R and the whole classification phase, running all the three classifiers in sequence, require about 50 minutes in total. On the other hand, the generation of the set D from the initial list of website L by means of web scraping and OCR procedures is much more time consuming, requiring several hours; however this part is intended to be performed offline, and it can also be completely parallelized on several machines.

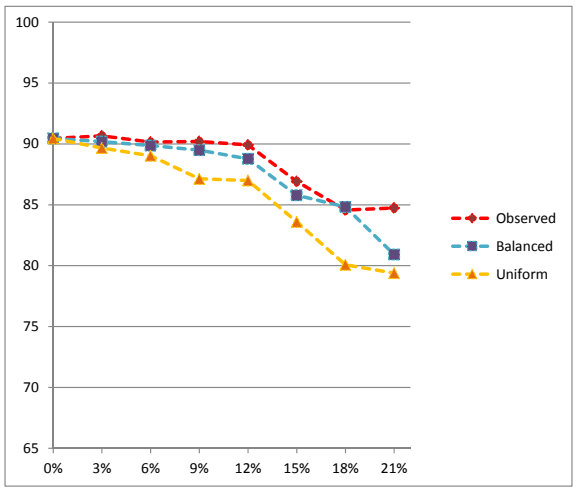

Figure 1: Accuracy of Random Forest classifier for different perturbations of the training set.

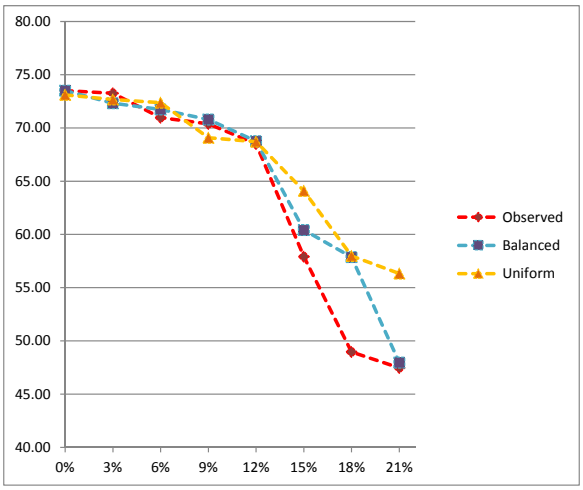

Figure 2: F1 score of Random Forest classifier for different perturbations of the training set.

5 Conclusions

Website categorization has recently emerged as a very important task in several contexts, because it allows the automatic individuation of some feature of interest by solving a classification problem. Determining whether an enterprise website offers e-commerce facilities or not is a particularly interesting case of this problem. However, to use a classification algorithm, it is necessary to convert each website into a record describing the website in a compact and tractable manner. These records should contain only the relevant portion of the information of the websites, and a careful selection of the information inserted in the records is a key element for obtaining satisfactory performances in the classification. On the other hand, we found that the use of classification algorithms provides also a certain degree of robustness with respect to the presence of errors in the class labels of the training set. This feature is very useful in practice, because in similar cases the class label physiologically contain some errors. Our experiments show that the overall procedure presented in this work constitute a practically viable technique that is able to perform automatic categorization of websites with a satisfactory degree of accuracy and robustness.

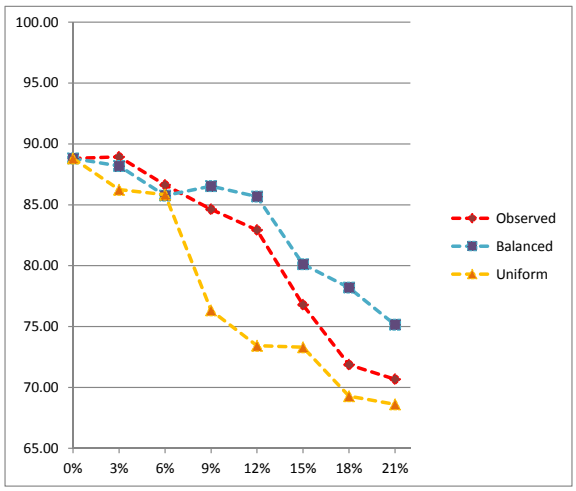

Figure 3: Accuracy of Support Vector Machines classifier for different perturbations of the training set.

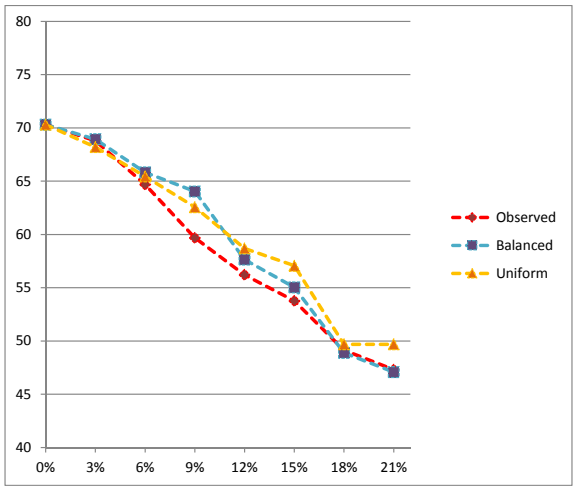

Figure 4: F1 score of Support Vector Machines classifier for different perturbations of the training set.

References

[1] G.A. Archanjo, F. Andrijauskas, D. Munoz: "Marvin–A Tool for Image Processing Algorithm Development", Technical Posters Proceedings of XXI Brazilian Symposium of Computer Graphics and Image Processing, 2008.

[2] G.Barcaroli, G.Bianchi, R.Bruni, A.Nurra, S.Salamone, M.Scarnò, Machine learning and statistical inference: the case of Istat survey on ICT, Proceeding of 48th scientific meeting of the Italian Statistical Society SIS 2016, Salerno, Italy, 2016. Editors: M. Pratesi and C. Pena ISBN: 9788861970618

[3] D. Blazquez, J. Domenech, J. Gil, A. Pont, Automatic detection of e-commerce availability from web data, Proceedings of First International Conference on Advanced Research Methods and Analytics (CARMA2016), València, Spain, 2016.

[4] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python. OReilly Media, 2009.

[5] L. Breiman. Random Forests. Machine Learning. 45 (1): 532 (2001).

[6] R. Bruni, G. Bianchi, Effective Classification using Binarization and Statistical Analysis, IEEE Transactions on Knowledge and Data Engineering 27(9), 2349-2361 (2015).

[7] C.-C. Chang and C.-J. Lin, Training ν-support vector classifiers: Theory and algorithms, Neural Computation, 13(9), 2119-2147 (2001).

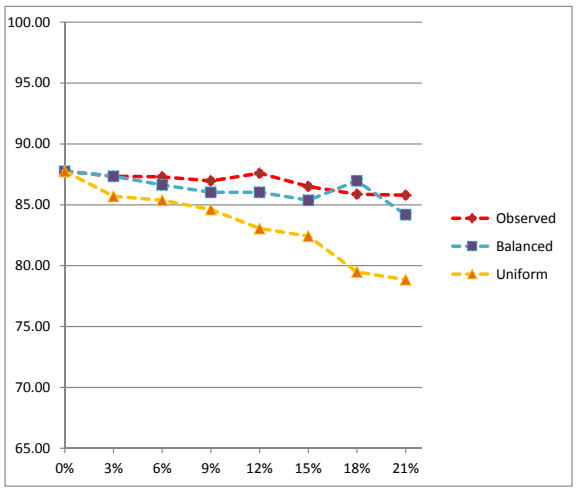

Figure 5: Accuracy of Logistic classifier for different perturbations of the training set.

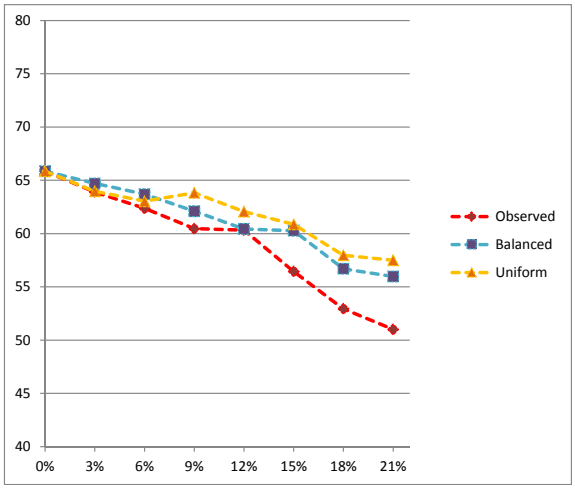

Figure 6: F1 score of Logistic classifier for different perturbations of the training set.

[8] Debole F., Sebastiani F.: Supervised Term Weighting for Automated Text Categorization. In: Sirmakessis S. (eds) Text Mining and its Applications. Studies in Fuzziness and Soft Computing, vol 138. Springer, Berlin, Heidelberg 2004

[9] Domingos, P. The role of Occams razor in knowledge discovery. Data Mining and Knowledge Discovery 3, 409425, 1999.

[10] R. Feldman, J. Sanger, The Text Mining Handbook. Cambridge University Press, 2006.

[11] D.A. Freedman, Statistical Models: Theory and Practice. Cambridge University Press, 2009.

[12] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer-Verlag, 2002.

[13] T.K. Ho: The Random Subspace Method for Constructing Decision Forests. IEEE Transactions on Pattern Analysis and Machine Intelligence. 20 (8): 832844 (1998).

[14] W. Klosgen, J.M. Zytkow (eds), Handbook of Data Mining and Knowledge Discovery, Oxford University Press, 2002.

[15] Lam W., Ruiz M., Srinivasan P.: Automatic text categorization and its application to text retrieval. IEEE Transactions on Knowledge and Data Engineering Vol 11(6), 1999, 865-879.

[16] Pearson, K. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. Philosophical Magazine 50(302) 157-175, 1900 DOI: 10.1080/14786440009463897

[17] F. Pedregosa et al., Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, 2825-2830 (2011).

[18] Qi, X., Davison, B.D. Web page classification: Features and algorithms. ACM Computing Surveys Vol 41(2), article 12, 2009.

[19] J.R Quinlan, Induction of Decision Trees. Machine Learning, 1, 81-106 (1986).

[20] R. Rehurek, P. Sojka, Software Framework for Topic Modelling with Large Corpora, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, Malta, 2010.

[21] H. Schmid, Improvements in Part-of-Speech Tagging with an Application to German. Proceedings of the ACL SIGDAT-Workshop. Dublin, Ireland, 1995.

[22] Sebastiani F. Machine Learning in Automated Text Categorization ACM Computing Surveys Vol 34(1), 1-47, 2002.

[23] Sokolova M., Japkowicz N., Szpakowicz S. (2006) Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. In: Sattar A., Kang B. (eds) AI 2006: Advances in Artificial Intelligence. Lecture Notes in Computer Science, vol 4304. Springer, Berlin, Heidelberg

[24] R. Smith, An Overview of the Tesseract OCR Engine. in Proc. of the Ninth International Conference on Document Analysis and Recognition, 629-633, 2007 ISBN:0-7695-2822-8 IEEE Computer Society, Washington, USA, 2007.

[25] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, second edition, 1995.