

Title

xxx¹, yyy¹, and zzz¹

¹Faculty of Mathematics and Information Science, Warsaw University of Technology
ul. Koszykowa 75, 00-662 Warsaw, Poland
{xxx, yyy, zzz}@mini.pw.edu.pl

Abstract. In the article we present

1 Introduction

Pattern recognition is a well known, classical machine learning problem. Its aim is to form a model (called classifier) that, after initial training, can assign correct class labels to provided patterns. The training process usually consists of feeding example data to classifier and providing answers in form of proper labels for each pattern in a training set. The ideal situation occurs when trained model is properly classifying patterns from outside of training dataset which means it gained ability to generalise information.

It is important to have in mind that patterns in their original form are often some sort of signal, for instance images or voice recordings. Due to the fact that the original patterns are often collected using some signal-acquiring devices, patterns that do not belong to any of the proper classes may be encountered. Such situation may happen, when the device that has been used to acquire data was automatically reset due to power outage and poor default calibration distorts the segmentation process. Another scenario is when we collect data in a noisy (out of lab) environment and apart from proper patterns there are a lot of unexpected residual ones. The main problem with such patterns, called garbage patterns, is that their characteristics cannot be easily predicted and therefore cannot be included in the model training process.

The motivation for our study is to provide algorithmic approaches used for distinguishing proper patterns (called native patterns) from garbage and erroneous ones (called foreign patterns) by using only classifiers for both rejection and classification. The design assumption is to provide methods based on native patterns only so that the approach could be truly versatile and be adapted to any pattern recognition problem in an uncertain environment, where foreign patterns may appear. The study focuses on providing novelty methods for classifier results interpretation that result in obtaining enhanced classifier with rejection capabilities. It should be emphasised that the novelty of the contribution presented in this paper lies not in the methods that are used, but in the proposed changes for already existing solutions.

The remainder of this paper is organized as follows. Section 2 presents the background knowledge on foreign patterns detection present in the literature. Section 3.1 presents the backbone algorithms, known in the literature, that were used to construct our models. Section 3.3 presents the proposed approach. Section 4.3 discusses a series of experiments. Section 5 concludes the paper and highlights future research directions.

2 Literature Review

AJ

3 Preliminaries

Modern studies often face the situation of contaminated datasets. This issue emerges from a substantial automation of data acquisition and processing, that results in poor data quality. The research we present in this paper has been motivated by the problem of purging datasets: accepting proper patterns and rejecting garbage ones. In this paper we concentrate on enhancing classifiers, such as SVM, Random Forests, KNN and regression-based solutions, with rejection mechanism. Our approach, based on those algorithms, is discussed in Section 3.3.

3.1 The Task of Classification with Rejection

The task of classification is to categorize unknown elements to their appropriate groups basing on their characteristics. Ideally, a category illuminates a relationship between the subjects and objects of knowledge. There are many mathematical models that can be used as classifiers in computer science studies, such as SVM, Random Forests, KNN, Regression model or Neural Networks. Their main disadvantage lies in their need to be trained prior to usage, which makes them unable to recognize elements from a completely new class, not used during training process. This unwanted behaviour can be especially troublesome in an unstable, noisy environment where patterns sent for classification can become corrupt, distorted or otherwise indistinguishable. In such situation a proper mechanism for rejecting garbage patterns could be used to provide better results.

3.2 Regression

Regression analysis is a statistical process for estimating the relationships between variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables. The estimation target is a function of the independent variables called the regression function. There are many different

regression models: linear regression, polynomial regression, logistic regression, Bayesian Ridge regression, etc. In this paper we will focus mainly on two regression variants: logistic regression and polynomial one (please note that linear regression is in fact polynomial regression using polynomial of degree 2).

Logistic regression relies on the assumption that independent variables are dichotomous (it means that their values are either 0 or 1), in other words they describe presence or absence of certain facts. In this model, the probabilities describing possible outcomes of a single trial are modelled using a logistic function. Model response is evaluated using the maximum likelihood method. Predicted values are probabilities and are therefore restricted to $[0,1]$ through the logistic distribution function.

Linear regression models relationship between a scalar dependent variable y and one or more explanatory variables denoted X . In linear regression, the relationships are modelled using linear predictor functions in form of [1] whose unknown model parameters are estimated from the data.

$$y = a_0 + a_1x_1 + a_2x_2 + \dots + a_kx_k + \epsilon_0 \quad (1)$$

where:

- y - dependent variable
- x_i - explanatory variable
- a_i - parameter variable
- ϵ - error term (ideally equals zero)

Polynomial regression can be viewed as a model that uses linear regression extended by constructing polynomial features from the coefficients. This approach maintains the generally fast performance of linear methods, while allowing them to fit a much wider range of data. The predictors resulting from the polynomial expansion of the "baseline" predictors are known as interaction features, and can be used in classification problem. Just like linear regression, polynomial regression models are usually fit using the method of least squares.

3.3 Rejection Mechanism for Classifiers

Calculations presented in this paper were conducted on a mixed set consisting of letters and digits. Because the aim of our work was to provide classifier-based solution for classifying digits and rejecting other symbols, we decided to compare three different approaches towards this problem. All of them based on the assumption that there is no information regarding outliers during classifier training.

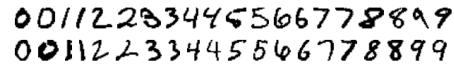
As the first way to deal with presented issue "one-versus-all" method was prepared. This approach requires creating a vector of classifiers constructed in a specific way. Each classifier has to be trained on a specially prepared training data, consisting of two sets: the first one (denoted as class_1) holding all training data entries for certain native class, and the second one (denoted as class_2) being the result of a subset sum operation performed on the rest of the classes except for the class used in class_1 set. Please note that both class_1 and class_2 sets should have the same size, so it is advisable to randomly choose elements when creating class_2. One classifier for each native class has to be present in a final vector. The actual classification with rejection is performed by presenting the unknown pattern to each of the classifiers from the vector. When the classifier recognizes element as a native one (belonging to class_1) then the pattern is treated as a recognized, and classified native element. In case of all classifiers rejecting such pattern (classifying it as element from class_2), it is treated as outlier and rejected. It is also worth noting that there is a possibility of more than one classifier recognizing the pattern as a native element. In such case randomly chosen class is assigned to this pattern.

The second approach uses "one-versus-one" method. Similarly to the previous one it requires preparing vector of classifiers, but this time it consists of $\binom{n}{2}$ classifiers, where n is the number of unique native classes. Each classifier is trained on data consisting of two sets: the first one (denoted as class_1) holding all training data entries for certain native class, and the second one (denoted as class_2) holding all training data entries for some other class (not the same as class_1). In the end there is one classifier for each pair of classes: 1 vs. 2, 1 vs. 3, ..., 1 vs. n , ..., $(n-1)$ vs. n . Classification with rejection mechanism is based on presenting unknown pattern to every classifier in the vector and remembering their answers (e.g. classifier constructed for 1 vs. n classes can classify pattern as belonging to class 1 or class n). In the end those answers can be presented as a n -wide array with each element being the number of times pattern was classified as belonging to certain class. The pattern is rejected when difference between two biggest values in the result array is smaller than two. In such case it is assumed that classifiers were uncertain as to which class should this unknown element belong to. Otherwise the pattern is classified as an element from the class which had the biggest value in the result array. [tutaj mona doda schemat prezentujcy opisan koncepcj?](#)

The last prepared and examined method, presented in this work, bases on the way of constructing classifiers vector used in the second approach ("one-versus-one"). The difference between those two methods lies in a rejection mechanism. In this method an unknown pattern is treated as a foreign element if its biggest value in the result array is lesser than $(n-1)$. What it actually means is that there must be a certain class that has always been chosen by a classifier from the vector. [tutaj mona doda schemat prezentujcy opisan koncepcj?](#)

3.4 Quality Evaluation

- *Correctly Classified* is the number of correctly classified patterns, i.e. native patterns classified as native ones with the correct class.
- *True Positives* is the number of native patterns classified as native (no matter, into which native class),
- *False Negatives* is the number of native patterns incorrectly classified as foreign,
- *False Positives* is the number of foreign patterns incorrectly classified as native,
- *True Negatives* is the number of foreign patterns correctly classified as foreign.
- *Strict Accuracy* is the absolute measure of the classifier's performance. It is the ratio of the number of all *correctly* classified patterns, i.e. native patterns classified to their respective classes and rejected foreign ones to the number of all patterns being processed.
- *Accuracy* is a characteristic derived from strict accuracy by ignoring the need to classify native patterns to their respective classes; in other words, it is sufficient to correctly identify whether a pattern is native or foreign one. This measure describes the ability to distinguish between native and foreign patterns.
- *Native Precision* is the ratio of the number of not rejected native patterns to the number of all not rejected patterns (i.e. all not rejected native and foreign ones). Native Precision evaluates the ability of the classifier to distinguish native patterns from foreign ones. The higher the value of this measure, the better ability to distinguish foreign elements from native ones. Native Precision does not evaluate how effective identification of native elements is.
- *Native Sensitivity* is the ratio of the number of not rejected native patterns to all native ones. This measure evaluates the ability of the classifier to identify native elements. The higher the value of Native Sensitivity, the more effective identification of native elements. Unlike the Native Precision, this measure does not evaluate the effectiveness of separation between native and foreign elements.
- *Strict Native Sensitivity* takes only correctly classified native patterns and does not consider native patterns, which are not rejected and assigned to incorrect classes, unlike *Native Sensitivity*, where all not rejected native patterns are taken into account.
- *Fine Accuracy* is the ratio of the number of native patterns classified to correct classes, i.e. assigned to their respective classes, to the number of all native patterns not rejected. This measure conveys how precise is correct classification of not rejected patterns.
- *Foreign Precision* corresponds to Native Precision.
- *Foreign Sensitivity* corresponds to Native Sensitivity.
- Precision and Sensitivity are complementary and there exists yet another characteristic that combines them: the *F-measure*. It is there to express the balance between precision and sensitivity since these two measures affect



00112233445566778899
00112233445566778899

Fig. 1. ...

each other. Increasing sensitivity can cause a drop in precision since, along with correctly classified elements, there might be more incorrectly classified,

4 Experiments

4.1 Presentation of Datasets

We assumed that handwritten digits (ten symbols from 0 to 9) were the native data set and handwritten letters (26 symbols from a to z) were the foreign data set. In other words, we focus on handwritten digits recognition task, and treat letters as garbage patterns here. The justification to assume such foreign dataset for testing purposes is that appearance of other real-world symbols, but not belonging to any proper class, is a common issue in a character recognition problem.

It is very important to note that foreign patterns do not participate in the model building phase. The entire scheme is trained on native patterns set only and the foreign set is used for rejection quality evaluation only. Samples of processed patterns are displayed in Figure 1.

The native dataset consisted of 10,000 handwritten digits with approximately 1,000 observations in each class taken from publicly available MNIST database. The MNIST dataset is publicly available online under [2]. We split each class in proportion ca. 7:3 and as a result we got two sets. The first one included 6,996 patterns and was used for training. The second set, the test set, contained 3,004 patterns. The dataset of foreign patterns contained 26,383 handwritten Latin letters, ca. 1,000 letters in each class. This dataset was created by 16 students, writing about 70 copies of each letter.

All patterns were normalized and feature vectors comprising of 106 numerical features were created. Examples of features are: maximum/position of maximum values of projections, histograms of projections, transitions, offsets; raw moments, central moments, Euler numbers etc. We considered features standardization but the training data is sufficiently consistent (there are no outliers), so we normalized those features to bring linearly all values into the range [0,1]. The best first search for the optimal feature subset has been performed using FSelector R package, [3] and then analysis of variance was employed to select good features. The final feature vector contained 24 elements.

4.2 Experimental Settings

Solutions presented in this paper have been implemented in Python programming language, using scientific libraries [4, 5] and their implementations for SVM, Random Forests, KNN, logistic and polynomial regression. Several tests have been performed in order to find best suited method parameters for all classifiers by using Grid Search [5] algorithm.

4.3 Rejection Quality

The results of our study is presented in Table 1. As it can be seen Polynomial Regression model achieved better accuracy in classifying native patterns and was better at rejecting the foreign ones. It turned out that the third approach (Method 3) yielded the best results whereas the second one performed the worst. Method 2 was the one that "destroyed" classification capabilities of classifiers and had tendency to reject native patterns. On the other hand Method 1 accepted most of the foreign elements. Those results become more obvious if one looks at them from the methods' construction point of view.

The first approach rejected input only after n failed tests which makes rejection less probable the more tests are performed (more native classes means more tests). Provided pattern was treated as native and labelled as an element from the first class that passed its corresponding test. Because tests were done in order (from class 1 to class n) we observed increased number of classified elements being treated as members from the first few classes.

The second method required certain class to be obviously "stronger" than the others in terms of classification. All patterns that had similarities between two or more different classes were rejected which resulted in poor classification capabilities. This result is not surprising given how often even people can be unsure what kind of handwritten symbol they see. Although it's not too hard for them to distinguish between a letter and a digit, correct classification of certain digits can be sometimes troublesome. Method 2 automatically rejected all symbols that were similar to two or more different classes.

The third method provided some sort of equilibrium between classification and rejection by modifying method 2. Elements were rejected only when there was no strong evidence that they belong to certain class. This evidence was in fact the number of times pattern was assigned certain class' label. If there was no class with required number of labels the element was identified as a foreign one. This approach can be compared to real-life situation where person classifies patterns by intuition and rejects them only when is really unsure.

5 Conclusion

AJ Proposed ...
In future ...

Acknowledgment

The research is partially supported by the

References

1. Hempstalk, K., Frank, E., Witten, I., *One-class classification by combining density and class probability estimation*, Machine Learning and Knowl. Disc. in Databases, pp. 505-519, 2008.
2. LeCun, Y., Cortes, C., and Burges, C., *The MNIST database of handwritten digits*, in: <http://yann.lecun.com/exdb/mnist>.
3. Romanski, P., Kotthoff, L., *Package FSelector*, <http://cran.r-project.org/web/packages/FSelector/FSelector.pdf>
4. <http://www.numpy.org/>
5. <http://scikit-learn.org/stable/>

Table 1. Results for classification with rejection when using only classifiers with three different approaches.

	Method 1		Method 2		Method 3	
Regression type	Polynomial	Logistic	Polynomial	Logistic	Polynomial	Logistic
Data Set	Native Patterns, Train Set					
Strict Accuracy	0.137	0.186	0.716	0.726	0.888	0.236
Fine Accuracy	0.645	0.731	1	0.897	1	0.926
Strict Native Sens.	0.645	0.720	0.610	0.030	1	0.921
Data Set	Native Patterns, Test Set					
Strict Accuracy	0.064	0.114	0.719	0.820	0.853	0.143
Fine Accuracy	0.602	0.738	0.969	0.882	0.959	0.925
Strict Native Sens.	0.601	0.727	0.494	0.030	0.811	0.920