Integrating Global and Local Application of Naive Bayes Classifier

Sotiris Kotsiantis
Department of Mathematics, University of Patras, Greece

Abstract: *Naive Bayes algorithm captures the assumption that every attribute is independent from the rest of the attributes, given the state of the class attribute. In this study, we attempted to increase the prediction accuracy of the simple Bayes model by integrating global and local application of Naive Bayes classifier. We performed a large-scale comparison with other attempts that have tried to improve the accuracy of the Naive Bayes algorithm as well as other state-of-the-art algorithms on 28 standard benchmark datasets and the proposed method gave better accuracy in most cases.*

Keywords: *Naive Bayes classifier, data mining, machine learning.*

Received July 30, 2011; accepted December 30, 2011; published online April 4, 2013

1. Introduction

Naive Bayes classifier is among the most popular learners used in the machine learning community [6]. The study of probabilistic classification is the approximation of a joint distribution with a product distribution. Probabilistic learners operate on data sets where each example x consists of attribute values $<a_1, a_2, ..., a_i>$ and the target function y can take on any value from a pre-defined finite set $V = (v_1, v_2, ..., v_j)$. Classifying unseen instances involves calculating the most probable target value v_{max} and is defined as: $v_{max} = \max_{v_j \in V} P(v_j \mid a_1, a_2, ..., a_i)$. Using Bayes theorem v_{max} can be rewritten as: $v_{max} = \max_{v_j \in V} P(a_1, a_2, ..., a_i \mid v_j) P(v_j)$. Bayes rule is used to estimate the conditional probability of a class label y, and then assumptions are made on the model, to decompose this probability into a product of conditional probabilities. Under the assumption that attribute values are conditionally independent given the target value, the formula used by the simple Bayes classifier is: $v_{max} = \max_{v_j \in V} P(v_j) \prod_i P(a_i \mid v_j)$, where V is the target output of the learner and $P(a_i|v_j)$ and $P(v_i)$ can be calculated based on their frequencies in the training set.

The assumption of independence is clearly almost always wrong. However, a large-scale comparison of naive Bayes classifier with state-of-the-art algorithms for decision tree induction and instance-based learning on standard benchmark datasets found that simple Bayesian classifier sometimes is superior to each of the other learning schemes even on datasets with substantial feature dependencies [8]. An explanation why simple Bayes method remains competitive, even though it provides very poor estimates of the true underlying probabilities can be found in [14].

However, there are comparisons where the accuracy of the naive Bayes algorithm is near the bottom as for instance in the experiment of [27]. Nevertheless, in that case too, this algorithm had the best accuracy per needed training time [9]. It predicts the class attribute in a very short time. Another advantage of simple Bayesian classifier is that during classification can easily handle data with missing values, whereas decision tree and neural network cannot.

In this study, we attempted to increase the prediction accuracy of the simple Bayes model by integrating global and local application of Naive Bayes (NB) classifier. During the classification of a test instance the model calculate the probabilities each class and if the probability of the most possible class is at least two times the probability of the next possible class then the decision is that of global NB model. However, if the global NB is not so sure e.g., the probability of the most possible class is less than two times the probability of the next possible class; the model finds the k nearest neighbors using a distance metric and train the local simple Bayes classifier using these k instances. Finally, in this case the model averages the probabilities of global NB with local NB classifier for the classification of the testing instance.

We performed a large-scale comparison with other attempts that have tried to improve the accuracy of the simple Bayes algorithm as well as other state-of-the-art algorithms on 28 standard benchmark datasets and the proposed method actually gives better accuracy in most cases using less time for training, too.

Description of some of the attempts that have been tried to improve the performance of simple Bayesian classifier is given in section 2. Section 3 discusses the proposed algorithm for improving the performance of simple Bayesian classifier. Experiment results in a number of data sets are presented in section 4, while brief summary with further research topics are given in section 5.

2. Previous Attempts for Improving the Performance of Simple Bayes Classifier

The application of simple Bayes formula is straightforward for the categorical attributes. For numerical attributes, one can model the component marginal distributions in a wide variety of ways. The simplest would be to adopt some parametric form e. g., marginal Gaussian estimators [8]. Yang and Webb [39] propose proportional discretization and fixed frequency discretization, two efficient unsupervised discretization methods that are able to effectively manage discretization bias and variance of naive-Bayes classifier.

Another problem with simple Bayes formula is the zero counts. Zero counts are obtained when a given class and attribute value never occur together in the training set, and is problematic because the resulting zero probabilities will wipe out the information in all the other probabilities when they are multiplied. A solution to this problem is to incorporate a small-sample correction into all probabilities, such as the Laplace correction [8]. If an attribute value does not occur given some class, its probability is set to 1/N, where N is the number of instances in the training set. An exception occurs when there is an equal probability for the class label value in the Naive Bayesian algorithm. Balamurugan *et al.* [1] suggest a solution with the help of a partial matching method.

The most well known attempt for improving the performance of the simple Bayes algorithm is the discretization of the continuous attributes into intervals, instead of using the default option to utilize the normal distribution to calculate probabilities. Numerous discretization methods have been examined such as the partition of the range of the features into k equal sized intervals and the partition of the range into intervals containing the same number of instances. A brief survey of these attempts and an empirical study comparing the performance of the most well-known discretization methods is given in [10, 28]. Yang and Webb [39] propose proportional discretization and fixed frequency discretization, two efficient unsupervised discretization methods that are able to effectively manage discretization bias and variance of naive-Bayes classifier. Hsu *et al.* [18] propose Extended Naive Bayes (ENB), which is capable for handling mixed data.

The performance of the naive Bayes Classifier on domains with redundant features can be also improved by removing redundant features [7]. Kohavi and John [24] used the best first backward selection method as search method for applying the wrapper approach. The ‘selective Bayesian classifier’, has been explored by numerous researchers such as [3]. Other researchers have explored the possibility of using a decision tree algorithm as a pre-processor to discover useful feature subsets for simple Bayes classifier [16, 32]. They showed that this combination has very good results with simple Bayes algorithm.

An iterative approach of simple Bayes is presented in [15]. The iterative Bayes begins with the distribution tables built by the simple Bayes and then the algorithm iteratively cycles through all the training examples using a hill-climbing technique. Experimental evaluation of iterative Bayes showed minor but consistent gain in accuracy in relation to simple Bayes [15]. However, the contingency tables are incrementally updated each time a training example is seen, which implies that the order of the examples could influence the final prediction.

Another attempt for improving the simple Bayes model was the Bayesian trees [40]. A Bayesian tree-learning algorithm builds a decision tree, and generates a local simple Bayesian classifier at each leaf. The tests, leading to a leaf, can alleviate feature inter-dependencies for the local simple Bayesian classifier. Zheng and Webb [40] also proposed the application of lazy learning techniques to Bayesian tree induction and presented the resulting lazy Bayesian rule-learning algorithm, called LBR. For each test instance, it builds a most appropriate rule with a local naive Bayesian classifier as its consequent. Kohavi [23] also, presents a model NBTree to combine a decision tree with naive Bayes. In an NBTree, a local Naive Bayes is deployed on each leaf of a traditional decision tree, and an example is classified using the local naive Bayes on the leaf into which it falls.

Hidden naive Bayes (HNB) creates a hidden parent for each attribute, which represents the influences from all other attributes [20]. The AODE classifier [37] is also considered an improvement on NB. Sahami [33] introduced the notion of k-dependence estimators, through which the probability of each attribute value is conditioned by the class and, at most, k other attributes. In order to maintain efficiency, AODE is restricted to exclusively use 1-dependence estimators (ODEs). Specifically, AODE makes use of SPODEs, as every attribute depends on the class and another shared attribute, designated as superparent. AODE weakens the attribute independence assumption by averaging all models from a restricted class of one-dependence classifiers. Motivated by their work, other authors [19] assigned different weights to these one-dependence classifiers in at attempt to obtain better results. This algorithm called Weightily Averaged One-Dependence Estimators, simply WAODE.

Shengtong and Langseth [34] focus on an alternative technique for learning the conditional probability tables from data. Instead of frequency counting (which leads to maximum likelihood parameters), they learn the (local) conditional probability tables under the guidance of the (global) NB model learnt thus far.

Calders and Verwer [5] present three approaches for making the naive Bayes classifier discrimination-free:

1. Modifying the probability of the decision being positive.
2. Training one model for every sensitive attribute value and balancing them.
3. Adding a latent variable to the Bayesian model that represents the unbiased label and optimizing the model parameters for likelihood using expectation maximization.

Jiang *et al.* [21] single out another two algorithms: Instance Weighted Naive Bayes (IWNB) and Combined Neighbourhood Naive Bayes (CNNB). In IWNB, each training instance is firstly weighted according to the similarity between it and the mode of the training instances, and then a NB classifier is built on the weighted training instances. In CNNB, multiple NB are firstly built on multiple neighbourhoods with different radius values for a test instance, and then their class probability estimates are averaged to estimate the class probability of the test instance.

Lately in the area of ML the concept of combining classifiers is proposed as a new direction for the improvement of the performance of individual classifiers. Kim *et al.* [22] built an ensemble of simple Bayes classifiers using boosting procedure [13]. Other authors made use of boosting, with the difference that in each iteration of Adaboost, they used a discretization method and they removed redundant features using a filter feature selection method [25]. The same authors combined simple Bayesian method with Logitboost [26]. However, as it is well known, Logitboost requires a regression algorithm for base learner. For this reason, they slightly modify simple Bayesian classifier in order to run as a regression method. Another way that has been examined for generation of ensemble of simple Bayesian classifiers is by using different feature subsets randomly and taking a vote of the predictions of each classifier that uses different feature subset [35].

3. The Proposed Algorithm

The proposed model simple trains a Naive Bayes classifier during the train process. For this reason, the training time of the model is that of simple Naïve Bayes. During the classification of a test instance the model calculate the probabilities each class and if the probability of the most possible class is at least two times the probability of the next possible class then the decision is that of global NB model. However, if the global NB is not so sure e.g., the probability of the most possible class is less than two times the probability of the next possible class; the model finds the *k* nearest neighbors using the selected distance metric and train the local simple Bayes classifier using these *k* instances. Finally, in this case the model averages the probabilities of global NB with local NB classifier for the classification of the testing instance. It must be mentioned that local NB classifier is only used for a small number of test instances and for this reason classification time is not a problem for our model.

Combining instance-based learning with naive Bayes is motivated by improving naive Bayes through relaxing the conditional independence assumption using lazy learning. It is expected that there are no strong dependences within the *k* nearest neighbors of the test instance, although the attribute dependences might be strong in the whole data. Essentially, they are looking for a sub-space of the instance space in which the conditional independence assumption is true or almost true. Generally, the proposed ensemble is described by the pseudo-code in Figure 1.

```
Training:
Build Global Naive Classifier in all the training set e. g. calculate $P(c)$ for each class c, and all $P(f|c)$: The probability of each feature f in each class c.
Classification:
1. Obtain the test instance.
2. Calculate the probabilities of belonging the instance in each class of the dataset e. g. take $P(c)$ times the calculated probability of each feature in class c, i. e. each $P(f|c)$ based on the test instance.
3. If the probability of the most possible class is at least two times the probability of the next possible class then the decision is that of global NB model else
    a. Find the k (=50) nearest neighbors using the selected distance metric (manhattan in our implementation).
    b. Using as training instances the k instances train the local simple Bayes classifier.
    c. Aggregate the decisions of global NB with local NB classifier by averaging of the probabilities for the classification of the testing instance.
```

Figure 1. Integrating global and local application of naive Bayes classifier.

The proposed algorithm requires choosing the value of *K*. There are several ways to do this. A first, simple solution is to fix *K* a priori before the beginning of the learning process. However, the best K for a specific dataset is obviously not the best one for another dataset. A second, more time-consuming solution is therefore to determine this best *K* automatically through the minimization of a cost criterion. The idea is to apply a model selection process upon which the different hypothesis that can be built. One way to do that is to evaluate the mean error on a test set and thus keep as K the value for which the error is the least. In the current implementation we decided to use a fixed value for *K* (=50) in order to a) keep the training time low and b) since about this size of instances is appropriate for a simple algorithm, to build a precise model according to [12]. According to our experiments, the results are quite similar for *K* values between 30 and 60. *K* (=50) is most suitable for datasets with many features.

The proposed algorithm has as free parameters the distance metric. For two data points, $X = <x_1, x_2, x_3, ..., x_{n-1}>$ and $Y = <y_1, y_2, y_3, ..., y_{n-1}>$, the euclidean similarity function is defined as:

$$D_1(x, y) = \left(\sum_{i=1}^{m} |x_i - y_i|^2 \right)^{1/2} \quad (1)$$

Where as manhattan similarity function is defined as:

$$D_2(x, y) = \sum_{i=1}^{m} |x_i - y_i| \quad (2)$$

And chebychev similarity function is defined as:

$$D_3(x, y) = \max_{i=1}^{m} |x_i - y_i| \quad (3)$$

The manhattan metric is a little cheaper to compute than the Euclidean distance so it may be used if the speed of a particular application is important. As a result, we used the manhattan similarity function as distance metric in our implementation.

4. Comparisons and Results

For the purpose of our study, we used 28 well-known datasets from many domains from the UCI repository [11]. These data sets were hand selected so as to come from real-world problems and to vary in characteristics. In order to calculate the classifiers' accuracy, the whole training set was divided into ten mutually exclusive and equal-sized subsets and for each subset the classifier was trained on the union of all of the other subsets. Then, cross validation was run 10 times for each algorithm and the median value of the 10-cross validations was calculated.

In Tables 1, 2 and 3, we represent with "*v*" that the proposed algorithm looses from the specific algorithm. That is, the specific algorithm performed statistically better than the proposed algorithm according to corrected t-test with $p<0.05$ [29]. The simple t-test assumes the samples are independent. However, due to the way cross validation works, the samples are not independent. Ignoring this assumption generally gives very high type I errors (that is, the test saying there is a difference between the tested algorithms while in fact there is not). The corrected t-test uses a fudge factor to counter the dependence between samples which in practice results in acceptable type *I* errors [29]. Furthermore, in the following Tables "*" indicates that the proposed algorithm performed statistically better than the specific classifier according to corrected t-test with $p<0.05$. In all the other cases, there is no significant statistical difference between the results (Draws). In the last row of the following Tables one can see the aggregated results in the form (*a/b/c*). In this notation "*a*" means that the proposed algorithm is significantly less accurate than the compared algorithm in *a* out of 28 data sets, "*c*" means that the proposed algorithm is significantly more accurate than the compared algorithm in *c* out of 28 data sets, while in the remaining cases (*b*), there is no significant statistical difference between the results.

In Tables 1 and 2, one can see the comparisons of the proposed algorithm with the other attempts that have tried to improve the classification accuracy of the simple Bayes algorithm. Eight well-known algorithms were used for the comparison: discretize simple Bayes [17], NB with kernel estimation [17], locally weighted naive Bayes [12], lazy bayesian rule-learning algorithm [40], discretize NB [17], averaged one-dependence estimator [37], weightily averaged one-dependence estimator [19], hidden naive Bayes algorithm [20].

Thus, the proposed algorithm is significantly more accurate than simple Bayes (NB) in 8 out of the 28 data sets, while it has significantly higher error rates than simple Bayes in none data set. The average error rate of our algorithm is also about 23% less than that of simple Bayes algorithm. In addition, the proposed algorithm has significantly lower error rates in 6 out of the 28 data sets than the simple discretize version of simple Bayes, while it is significantly less accurate in none data set. Furthermore, the proposed algorithm is significantly more accurate than NB with kernel estimation in 7 out of the 28 data while, the proposed algorithm has significantly higher error rate in none data set.

Moreover, the proposed algorithm is significantly more accurate than LBR algorithm in 6 out of the 28 data sets and even though, the proposed algorithm has significantly higher error rates in other 2 data sets it uses much less time for training and classification. The proposed algorithm is significantly more accurate than locally weighted naive Bayes (Local NB) algorithm in 4 out of the 28 data sets, while it is significantly less accurate in1data sets. Furthermore, the proposed algorithm is significantly more accurate than WAODE in 2 out of the 28 data sets using less time for training, too. In none data set, the proposed algorithm has significantly higher error rate. Moreover, the proposed algorithm is significantly more accurate than HNB algorithm in 3 out of the 28 data sets and the proposed algorithm has significantly higher error rates in other 1 data set. Finally, the proposed algorithm is significantly more accurate than AODE algorithm in 3 out of the 28 data sets, while it is significantly less accurate in none data set.

In addition, a representative algorithm for each of the other sophisticated machine learning techniques was tested Table 3. SMO algorithm was the representative of the support vector machines [30] and *K2* as a representative of bayesian networks [38]. We also used the 3-NN algorithm as a representative of kNN [38] and C4.5 as a representative of decision trees [31]. NBTree [23] also used as combination method of a decision tree with naive Bayes.

Table 1. Comparing the proposed algorithm with other attempts to improve the simple Bayes.

Dataset	IGLNB	NB		NB with Kernel Estimation		Discretize NB	
Audiology	74.04	72.64		72.64		72.64	
Autos	77.79	57.41	*	61.33	*	65.17	*
Balance-Scale	90.69	90.53		91.44		71.56	*
Breast-Cancer	72.05	72.7		72.7		72.7	
Wisconsin-Breast-Cancer	96.94	96.07		97.51		97.2	
Horse-Colic	82.2	78.7	*	79.24	*	79.54	*
German_Credit	76.29	75.16		74.38		75.04	
Pima_Diabetes	75.36	75.75		74.96		75.26	
Glass	73.62	49.45	*	51.09	*	71.94	
Haberman	75.09	75.06		74.77		71.57	
Cleveland-14-Heart-Diseas	83.61	83.34		84.17		83.47	
Hungarian-14-Heart-Diseas	84.36	83.95		84.97		84.2	
Heart-Statlog	83.15	83.59		84.07		82.56	
Hepatitis	86.04	83.81		85.16		84.34	
Ionosphere	91.31	82.17	*	91.8		89.4	
Iris	95.8	95.53		96.2		93.33	
Labor	94.23	93.57		93.4		88.57	
Lymphography	83.76	83.13		83.15		85.1	
Monk1	81.1	73.38	*	73.38	*	73.38	*
Monk2	61.06	56.83		56.83		56.83	
Monk3	93.2	93.45		93.45		93.45	
Primary-Tumor	47.23	49.71		49.71		49.71	
Sonar	85.7	67.71	*	72.4	*	76.71	*
Students	85.8	85.7		85.7		85.7	
Titanic	78.33	77.85	*	77.85		77.85	
Vehicle	68.87	44.68	*	60.9	*	61.06	*
Wine	98.88	97.46		97.34		98.71	
Zoo	96.73	94.97		95.17		93.21	
Average	81.90	77.65		79.13		78.93	
a/b/c		[0/20/8]		[0/21/7]		[0/22/6]	

Table 2. Comparing the proposed algorithm with other attempts to improve the simple Bayes.

Dataset	IGLNB	Local NB		AODE		WAODE		HNB		LBR	
Audiology	74.04	78.85		71.66	*	77.05		73.94		72.20	*
Autos	77.79	77.94		74.75		81.72		82		73.80	*
Balance-Scale	90.69	89.91		69.96	*	70.06	*	69.67	*	72.17	*
Breast-Cancer	72.05	72.86		72.73		71.97		70.23		72.35	
Wisconsin-Breast-Cancer	96.94	96.25		97.05		97.2		96.37		97.21	
Horse-Colic	82.2	82.28		82.25		81.66		81.69		82.33	
German_Credit	76.29	75.08		75.83		75.72		75.65		86.1	v
Pima_Diabetes	75.36	70.38	*	75.7		75.61		74.57		75.38	
Glass	73.62	72.2		74.53		73.26		73.77		72.32	
Haberman	75.09	70.59	*	71.57		71.57		71.57		71.57	
Cleveland-14-Heart-Diseas	83.61	81.39		82.87		82.64		81.95		83.54	
Hungarian-14-Heart-Diseas	84.36	82.3		84.67		85.28		84.87		84.54	
Heart-Statlog	83.15	79.33	*	82.7		82.07		82.74		82.59	
Hepatitis	86.04	86.27		84.06		83.87		84.64		84.91	
Ionosphere	91.31	82.91	*	91.09		92.4		91.48		90	
Iris	95.8	95.53		93.07		92.93		92.07	*	93.2	
Labor	94.23	93.3		91.07		90.7		90.67		87.50	*
Lymphography	83.76	83.82		86.86		88.22		85.57		85.45	
Monk1	81.1	84.42		82.32		75.46		97.66	v	94.91	v
Monk2	61.06	61.24		59.62		57.96		60.65		60.40	
Monk3	93.2	91.65		93.21		93.53		93.14		93.45	
Primary-Tumor	47.23	44.52		47.87		47.94		47.85		48.85	
Sonar	85.7	88.15		77.05	*	77.24	*	76.13	*	76.04	*
Students	85.8	83.47		85.96		85.3		86.58		85.38	
Titanic	78.33	79.05		78.21		78.88		79.04		78.31	
Vehicle	68.87	75.01	v	70.32		70.89		70.62		69.43	
Wine	98.88	98.93		98.21		97.98		97.59		98.71	
Zoo	96.73	97.21		94.66		98.1		97.11		93.21	*
Average	81.90	81.24		80.35		80.61		81.06		80.93	
a/b/c		[1/23/4]		[0/25/3]		[0/26/2]		[1/24/3]		[2/20/6]	

The proposed algorithm is also significantly more precise than C4.5 algorithm in 11 out of the 28 data sets, whilst it has significantly higher error rates in none data set. In addition, the proposed algorithm is significantly more accurate than SMO algorithm in 5 out of the 28 data sets, whereas it has significantly higher error rates in 2 data sets. The proposed algorithm is significantly more precise than 3NN algorithm in 8 out of the 28 data sets, while it has significantly higher error rates in none data set. The proposed algorithm is significantly more precise than NBTree algorithm in 4 out of the 28 data sets, while it

Table 3. Comparing the proposed algorithm with other state-of-the-art classifiers.

Dataset	IGLNB	3NN		SMO		C4.5		K2		NBTree		Adaboost NB	
Audiology	74.04	67.97	*	80.77	v	77.26		76.79		76.82		79.26	v
Autos	77.79	67.23	*	71.34	*	81.77		67.26	*	77.87		57.12	*
Balance-Scale	90.69	86.74	*	87.57	*	77.82	*	71.56	*	75.97	*	91.68	
Breast-Cancer	72.05	73.13		69.52		74.28		72.59		70.99		68.68	
Wisconsin-Breast-Cancer	96.94	96.6		96.75		95.01	*	97.2		96.38		95.55	
Horse-Colic	82.2	80.95		82.66		85.16		80.98		81.71		77.62	
German_Credit	76.29	72.21	*	75.09		71.25	*	74.97		74.27		75.14	
Pima_Diabetes	75.36	73.86		76.8		74.49		75.25		75.24		75.86	
Glass	73.62	70.02		57.36	*	67.63		71.56		69.90		49.63	*
Haberman	75.09	69.77		73.4		71.05		71.57		71.94		73.91	
Cleveland-14-Heart-Diseas	83.61	81.82		83.86		76.94	*	83.34		80.43		82.97	
Hungarian-14-Heart-Diseas	84.36	82.33		82.74		80.22	*	84.57		82.26		84.81	
Heart-Statlog	83.15	79.11	*	83.89		78.15	*	82.56		79.26	*	82.59	
Hepatitis	86.04	80.85	*	85.77		79.22	*	84.18		80.93	*	84.62	
Ionosphere	91.31	86.02		88.07		89.74		89.54		89.15		91.06	
Iris	95.8	95.2		96.27		94.73		93.2		93.80		94.8	
Labor	94.23	92.83		92.97		78.6	*	90.6		92.27		89.6	
Lymphography	83.76	81.74		86.48		75.84		85.64		80.80		81.27	
Monk1	81.1	78.97		79.58		80.61		73.46	*	91.78	v	72.68	*
Monk2	61.06	54.74	*	58.7		57.75		56.78		63.72		56.83	
Monk3	93.2	86.72	*	93.45		92.95		93.45		92.94		90.9	
Primary-Tumor	47.23	44.98		47.09		41.39	*	47.11		47.50		49.71	
Sonar	85.7	83.76		76.6	*	73.61	*	76.71	*	77.07	*	80.77	
Students	85.8	82.29		86.72		86.75		85.76		84.62		85.18	
Titanic	78.33	78.9		77.6	*	78.55		77.86	*	78.00		77.86	*
Vehicle	68.87	70.21		74.08	v	72.28		61.05	*	70.99		44.68	*
Wine	98.88	95.85		98.76		93.2	*	98.65		96.62		96.18	*
Zoo	96.73	92.61		96.05		92.61		94.37		94.44		97.23	
Average	81.90	78.84		80.71		78.53		79.23		80.27		78.15	
a/b/c		[0/20/8]		[2/21/5]		[0/17/11]		[0/22/6]		[1/23/4]		[1/21/6]	

has significantly higher error rates in one data set. In addition, the proposed algorithm is significantly more accurate than K2 algorithm in 6 out of the 28 data sets, whereas it has significantly higher error rates in none data set. Subsequently, we compare the performance of the proposed algorithm with the accuracy of the simple Bayes algorithm after applying the boosting procedures (with 10 classifiers) [2]. The proposed algorithm is significantly more accurate than single boosting simple Bayes (AdaBoostNB) algorithm (using 10 classifiers) in 6 out of the 28 data sets, while in 1 data set, the proposed algorithm has significantly higher error rates.

In brief, we managed to improve the performance of the simple Bayes Classifier obtaining better accuracy than other well known methods that have tried to improve the performance of the simple Bayes algorithm. The proposed algorithm also gave better accuracy than other sophisticated state-of-the-art algorithms on most of the 28 standard benchmark datasets.

5. Conclusions

Ideally, we would like to be able to identify or design the single best learning algorithm to be used in all situations. However, this is not possible. The simple Bayes classifier has much broader applicability than previously thought. Besides its high classification accuracy, it also has advantages in terms of simplicity, learning speed, classification speed and storage space. Turhan and Bener [36] analyze the assumptions of Naive Bayes using public software defect data from NASA. Their analysis shows that independence assumption is not harmful for software defect data with PCA pre-processing.

We managed to improve the prediction accuracy of the simple Bayes model by integrating global and local application of Naive Bayes classifier. We performed a large-scale comparison with other attempts that have tried to improve the accuracy of the simple Bayes algorithm as well as other state-of-the-art algorithms and ensembles on 28 standard benchmark datasets and we obtain better accuracy in most cases.

Local weighted learning algorithms must often decide what examples to store for use during generalization in order to avoid extreme storage and time complexity [38]. By removing a set of examples from a dataset the response time for classification decisions will decrease, as fewer examples are examined when a query example is presented. This objective is primary when we are working with large dataset and have limited storage. In a following work we will focus on the problem of reducing the size of the stored set of instances while trying to maintain or even improve generalization accuracy by avoiding noise and overfitting. Numerous instance selection methods that can be combined with the proposed technique can be found in [4]. Features can be also selected by keeping those for which gain ratio [7] exceeds a fixed threshold e. In order to have a robust selection, we can set e to 0. 02 of the gain ratio filter, in an attempt to discard only features with negligible impact on predictions. However, such a low threshold can discard many features.

References

[1] Balamurugan A., Rajaram R., Pramala S., Rajalakshmi S., Jeyendran C., Dinesh S., and Prakash J., "NB+: An improved Naïve Bayesian Algorithm," *Knowledge-Based Systems*, vol. 24, no. 5, pp. 563-569, 2011.

[2] Bauer E. and Kohavi R., "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine Learning*, vol. 36, no. 1-2, pp. 105-139, 1999.

[3] Boullé M., "Regularization and Averaging of the Selective Naïve Bayes Classifier," *in Proceedings of IEEE International Joint Conference on Neural Networks*, Vancouver, Canada, pp. 1680-1688, 2006.

[4] Brighton H. and Mellish C., "Advances in Instance Selection for Instance-Based Learning Algorithms," *Data Mining and Knowledge Discovery*, vol. 6, no. 2, pp. 153-172, 2002.

[5] Calders T. and Verwer S., "Three Naive Bayes Approaches for Discrimination-Free Classification," *Data Mining and Knowledge Discovery*, vol. 21, no. 2, pp. 277-292, 2010.

[6] Catal C., Sevim U., and Diri B., "Practical Development of an Eclipse-Based Software Fault Prediction Tool Using Naive Bayes Algorithm," *Expert System Application*, vol. 38, no. 3, pp. 2347-2353, 2011.

[7] Chen J., Huang H., Tian S., and Qu Y., "Feature Selection for Text Classification with Naïve Bayes," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5432-5435, 2009.

[8] Domingos P. and Pazzani M., "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss," *Machine Learning*, vol. 29, no. 2-3, pp. 103-130, 1997.

[9] El-Halees A., "Filtering Spam E-Mail from Mixed Arabic and English Messages: A Comparison of Machine Learning Techniques," *the International Arab Journal of Information Technology*, vol. 6, no. 1, pp. 52-59, 2009.

[10] Flores M., Gámez J., Martínez A., and Puerta J., "Analyzing the Impact of the Discretization Method When Comparing Bayesian Classifiers," *in Proceedings of the 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, Lecture Notes in Computer Science*, Spain, vol. 6096, pp. 570-579, 2010.

[11] Frank A. and Asuncion A., "UCI Machine Learning Repository," available at: http://archive.ics.uci.edu/ml, last visited 2010.

[12] Frank E., Hall M., and Pfahringer B., "Locally Weighted Naive Bayes," *in Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, Mexico, pp. 249-256, 2003.

[13] Freund Y. and Schapire R., "Experiments with a New Boosting Algorithm," *in Proceedings of International Conference on Machine Learning*, pp. 148-156, 1996.

[14] Friedman J., "On Bias, Variance, 0/1-Loss and Curse-of-Dimensionality," *Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 55-77, 1997.

[15] Gama J., "Iterative Bayes," *Intelligent Data Analysis*, vol. 4, no. 6, pp. 463-473, 2000.

[16] Hall M., "A Decision Tree-Based Attribute Weighting Filter for Naive Bayes," *Knowledge-Based Systems*, vol. 20, no. 2, pp. 120-126, 2007.

[17] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., and Witten I., "The WEKA Data Mining Software: An Update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10-18, 2009.

[18] Hsu C., Huang Y., and Chang K., "Extended Naive Bayes Classifier for Mixed Data," *Expert Systems with Applications*, vol. 35, no. 3, pp. 1080-1083, 2008.

[19] Jiang L. and Zhang H., "Weightily Averaged One-Dependence Estimators," *in Proceedings of 9th Biennial Pacific Rim International Conference on Artificial Intelligence*, Berlin, Germany, pp. 970-974, 2006.

[20] Jiang L., Zhang H., and Cai Z., "A Novel Bayes Model: Hidden Naive Bayes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 10, pp. 1361-1371, 2009.

[21] Jiang L., Cai Z., and Wang D., "Improving Naive Bayes for Classification," *the International Journal of Computers and Applications*, vol. 32, no. 3, pp. 328-332, 2010.

[22] Kim H., Kim J., and Ra Y., "Boosting Naïve Bayes Text Classification Using Uncertainty-Based Selective Sampling," *Neurocomputing*, vol. 67, no. 1, pp. 403-410, 2005.

[23] Kohavi R., "Scaling up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid," *in Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 202-207, 1996.

[24] Kohavi R. and John G., "Wrappers for Feature Subset Selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273-324, 1997.

[25] Kotsiantis S. and Pintelas P., "Increasing the Classification Accuracy of Simple Bayesian Classifier," *in Proceedings of the 11th International Conference Artificial Intelligence: Methodology, Systems, and Applications Lecture Notes in Computer Science*, Bulgaria, vol. 3192, pp. 198-207, 2004.

[26] Kotsiantis S. and Pintelas P., "Logitboost of Simple Bayesian Classifier," *Informatica*, vol. 29, no. 1, pp. 53-60, 2005.

[27] Michie D., Spiegelhalter D., and Taylor C., *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, USA, 1994.

[28] Mizianty M., Kurgan L., and Ogiela M., "Comparative Analysis of the Impact of Discretization on the Classification with Naive Bayes and Semi-Naïve Bayes Classifiers," *in Proceedings of the 7th International Conference on Machine Learning and Applications*, San Diego, USA, pp. 823-828, 2008.

[29] Nadeau C. and Bengio Y., "Inference for the Generalization Error," *Machine Learning*, vol. 52, no. 3, pp. 239-281, 2003.

[30] Platt J., "Using Sparseness and Analytic QP to Speed Training of Support Vector Machines," *in Proceedings of Advances in Neural Information Processing Systems* 11, USA, pp. 557-563, 1999.

[31] Quinlan J., "C4.5: Programs for Machine Learning. Morgan Kaufmann," *Machine Learning*, vol. 16, no. 3, pp. 235-240, 1993.

[32] Ratanamahatana C. and Gunopulos D., "Feature Selection for the Naive Bayesian Classifier Using Decision Trees," *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 475-487, 2003.

[33] Sahami M., "Learning Limited Dependence Bayesian Classifiers," *in Proceedings of the 2nd International Conference on Knowledge Discovery in Databases*, pp. 335-338, 1996.

[34] Shengtong Z. and Langseth H., "Local-Global-Learning of Naive Bayesian Classifier," *in Proceedings of the 4th International Conference on Innovative Computing, Information and Control*, Kaohsiung, Taiwan, pp. 278-281, 2009.

[35] Tsymbal A., Puuronen S., and Patterson D., "Feature Selection for Ensembles of Simple Bayesian Classifiers," *in Proceedings of the 13th International Symposium Foundations of Intelligent Systems*, Lyon, France, pp. 592-600, 2002.

[36] Turhan B. and Bener A., "Analysis of Naive Bayes' Assumptions on Software Fault Data: An Empirical Study," *Data & Knowledge Engineering*, vol. 68, no. 2, pp. 278-290, 2009.

[37] Webb G., Boughton J., and Wang Z., "Not So Naive Bayes: Aggregating One-Dependence Estimators," *Machine Learning*, vol. 58, no. 1, pp. 5-24, 2005.

[38] Witten I., Frank E., and Hall M., *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, USA, 2011.

[39] Yang Y. and Webb G., "Discretization for Naive-Bayes Learning: Managing Discretization Bias and Variance," *Machine Learning*, vol. 74, no. 1, pp. 39-74, 2009.

[40] Zheng Z. and Webb G., "Lazy Learning of Bayesian Rules," *Machine Learning*, vol. 41, no. 1, pp. 53-84, 2000.

Sotiris Kotsiantis is a lecturer in the University of Patras, Greece. He received a diploma in mathematics, a Master and a PhD degree in computer science from the University of Patras, Greece. His research interests are in the field of data mining, machine learning and ontologies. He has more than 100 publications to his credit in international journals and conferences. He has h-index: 17 and g-index: 39 according to google scholar.