Parsimonious Naive Bayes

Marc Boullé
Orange Labs,
2 avenue Pierre Marzin, 22300 Lannion, France
http://perso.rd.francetelecom.fr/boulle
Email: marc.boulle@orange.com

***Abstract*—We describe our submission to the AAIA'14 Data Mining Competition, where the objective was to reach good predictive performance on text mining classification problems while using a small number of variables. Our submission was ranked 6^{th}, less than 1% behind the winner. We also present an empirical study on the trade-off between parsimony of the representation and accuracy, and show how good performance can be obtained quickly and efficiently.**

I. Introduction

The AAIA'14 Data Mining Competition[1] is related to a problem of text classification. A corpus of 50,000 texts coming from reports of the Polish State Fire Service is provided, with a representation consisting of 11,852 variables (mainly based on documents words). The objective is to classify the texts into three binary classes by the mean of an ensemble of ten Naive Bayes classifiers, using as few variables as possible. In this paper, we present the method we used at the AAIA'14 Data Mining Competition. It mainly exploits the results of a Selective Naive Bayes classifier (summarized in Section II), trained for each of the three challenge class variables. The best subset selections of variables are collected and grouped together to form our submission to the challenge (see Section III).

Interestingly, the challenge evaluation criterion combines the test performance and the number of selected variables in a single formula. Whereas feature selection [1] aims at improving interpretability, test accuracy and deployment time, most papers in the literature focus on test performance only. Still, a small number of selected variables is often a requirement in practical data mining studies. In Section III, the trade-off between test performance and number of selected variables is investigated, using the challenge dataset as a case study. Finally, Section IV summarizes the paper.

II. Selective Naive Bayes Classifier

We summarize the Selective Naive Bayes (SNB) classifier introduced in [2]. It extends the Naive Bayes classifier owing to an optimal estimation of the class conditional probabilities, a Bayesian variable selection and a Compression-based Model Averaging.

[1] http://challenge.mimuw.edu.pl/mod/page/view.php?id=565

A. Optimal discretization

The Naive Bayes (NB) classifier has proved to be very effective in many real data applications [3], [4]. It is based on the assumption that the variables are independent within each class, and solely relies on the estimation of univariate conditional probabilities. The evaluation of these probabilities for numerical variables has already been discussed in the literature [5], [6]. Experiments demonstrate that even a simple equal width discretization brings superior performance compared to the assumption using a Gaussian distribution per class. In the MODL approach [7], the discretization is turned into a model selection problem and solved in a Bayesian way. First, a space of discretization models is defined. The parameters of a specific discretization are the number of intervals, the bounds of the intervals and the class frequencies in each interval. Then, a prior distribution is proposed on this model space. This prior exploits the hierarchy of the parameters: the number of intervals is first chosen, then the bounds of the intervals and finally the class frequencies. The choice is uniform at each stage of the hierarchy. Finally, the multinomial distributions of the class values in each interval are assumed to be independent from each other. A Bayesian approach is applied to select the best discretization model, which is found by maximizing the probability $p(Model|Data)$ of the model given the data. Owing to the definition of the model space and its prior distribution, the Bayes formula is applicable to derive an exact analytical criterion to evaluate the posterior probability of a discretization model. Efficient search heuristics allow to find the most probable discretization given the data sample. Extensive comparative experiments report high performance.

The case of categorical variables is treated with the same approach in [8], using a family of conditional density estimators which partition the input values into groups of values.

B. Bayesian Approach for Variable Selection

The naive independence assumption can harm the performance when violated. In order to better deal with highly correlated variables, the Selective Naive Bayes approach [9] exploits a wrapper approach [10] to select the subset of variables which optimizes the classification accuracy. Although the Selective Naive Bayes approach performs quite well on datasets with a reasonable number of variables, it does not scale on very large datasets with hundreds of thousands of instances and thousands of variables, such as in marketing applications or text mining. The problem comes both from the

search algorithm, whose complexity is quadratic in the number of variables, and from the selection process which is prone to overfitting. In [2], the overfitting problem is tackled by relying on a Bayesian approach, where the best model is found by maximizing the probability of the model given the data. The parameters of a variable selection model are the number of selected variables and the subset of variables. A hierarchic prior is considered, by first choosing the number of selected variables and second choosing the subset of selected variables. The conditional likelihood of the models exploits the Naive Bayes assumption, which directly provides the conditional probability of each label. This allows an exact calculation of the posterior probability of the models. Efficient search heuristic with super-linear computation time are proposed, on the basis of greedy forward addition and backward elimination of variables. The classifier resulting from the best subset of variables is the MAP (maximum a posteriori) Naive Bayes, which we call MNB in the rest of the paper.

C. Compression-Based Model Averaging

Model averaging has been successfully exploited in bagging [11] using multiple classifiers trained from re-sampled datasets. In this approach, the averaged classifier uses a voting rule to classify new instances. Unlike this approach, where each classifier has the same weight, the Bayesian Model Averaging (BMA) approach [12] weights the classifiers according to their posterior probability. In the case of the Selective Naive Bayes classifier, an inspection of the optimized models reveals that their posterior distribution is so sharply peaked that averaging them according to the BMA approach almost reduces to the MAP model. In this situation, averaging is useless. In order to find a trade-off between equal weights as in bagging and extremely unbalanced weights as in the BMA approach, a logarithmic smoothing of the posterior distribution, called Compression-based Model Averaging (CMA), is introduced in [2]. The weighting scheme on the models reduces to a weighting scheme on the variables, and finally results in a single Naive Bayes classifier with weights per variable. Extensive experiments demonstrate that the resulting Compression-based Model Averaging scheme clearly outperforms the Bayesian Model Averaging scheme. In the rest of the paper, the classifier resulting from model averaging is called Selective Naive Bayes (SNB).

D. Training Time Complexity

The algorithm consists in three phase: data preprocessing using discretization or value grouping, variable selection and model averaging. The preprocessing phase is super-linear in time and requires $O(KN \log N)$ time, where K is the number of variables and N the number of instances. In the variable selection algorithm, the method alternates fast forward and backward variable selection steps based on randomized reorderings of the variables, and repeats the process several times in order to better explore the search space and reduce the variance caused by the dependence over the order of the variables. The number of repeats is fixed to $\log N + \log K$, so that the overall time complexity of this phase is $O(KN(\log K + \log N))$, which is comparable to that of the preprocessing phase. The model averaging algorithm consists in collecting all the models evaluated in the variable selection phase and averaging then according to a logarithmic smoothing of their posterior probability, with no overhead on the time complexity. Overall, the train algorithm has an $O(KN(\log K + \log N))$ time complexity and $O(KN)$ space complexity.

III. Challenge Submission

A. Preliminary experiments

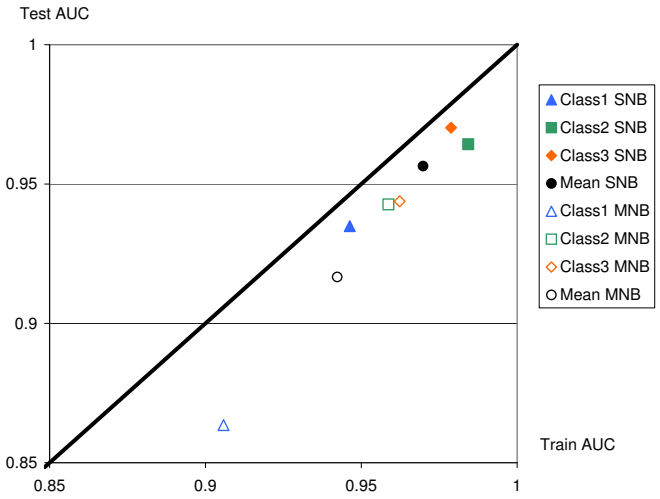

Fig. 1. Train versus test AUC for the MNB and SNB classifiers

The SNB classifier outputs several indicators for each variable:

- Level: evaluation of the predictive importance of the variable taken individually, based a normalized estimation of class conditional entropy. The level is between 0 (variable without predictive interest) and 1 (variable with optimal predictive importance).
- MAP: indicates that the variable belongs to best subset of variables, related to the MNB classifier.
- Weight: weight of the variable in the SNB classifier that exploits the model averaging method summarized in Section II.

We consider the standard NB classifier that exploits all the predictive variables; the non-informative variables with Level 0 are discarded. We also consider the 1NB classifier, which uses only one variable, the one with the highest Level. The MNB classifier is based on a subset of variables with fewer redundancy problems than the NB classifier. The SNB classifier exploits the same variables as the NB classifier, with weights per variable: it cannot be considered as a true Naive Bayes classifier.

As the challenge requires few variables, we focus on the MNB classifier which is very parsimonious compared to the SNB classifier, although it is both less accurate and less robust.

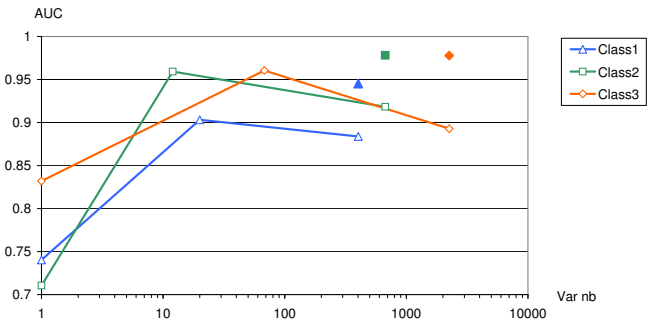

Fig. 2. AUC versus number of variables for the 1NB, MNB, NB and SNB classifiers

We trained the classifier [2] on a $70\% - 30\%$ split of the challenge dataset to evaluate its performance and robustness. We obtained a mean 96.9 train AUC and 95.6 test AUC for the SNB classifier, 94.0 train AUC and 91.6 test AUC for the MNB classifier, which confirms the usual behavior of both classifiers. This is illustrated in Figure 1, where the mean train and test AUC are presented as well as the detailed AUC per class.

We then trained the classifier using all the available data in order to reach better accuracy and robustness, with an expected decrease in accuracy on the challenge hidden dataset of about 1% for the SNB and 2% the MNB. In Figure 2, we report the AUC versus the number of variables for the 1NB, MNB, NB (empty shapes on the curves) and SNB (plain shapes) classifiers. As a reminder, the default AUC (with no variables) is 0.5. The results show that using one single variable, the 1NB classifier gets very good AUC, between 0.70 and 0.85. The MNB obtains 0.903 AUC with 20 variables for the first class, 0.959 AUC with 12 variables for the second class, 0.961 AUC with 68 variables for the third class. The NB classifier that keeps between 400 and 2000 informative variables out of the 11,852 input variables suffers from redundancy between the variables, which is harmful w.r.t. the independence assumption. All together, the NB uses far more variables than the MNB and gets a lower AUC. As expected, the SNB classifier obtains the higher accuracy using variable weights.

B. First submission

To get familiar with the challenge evaluation protocol, we grouped together the MAP variables of our three MNB classifiers (without removing the duplicates) and obtained a set of 100 variables. As the variable number is greater than that of each MNB, a better accuracy and robustness can be expected. The challenge rules states that the variable set is evaluated using and ensemble of ten Naive Bayes classifiers, each consisting of at least three variables. We chose to partition our set of 100 variables into ten random subsets of equal size, with the hope that the resulting ensemble classifier would behave similarly to a single Naive Bayes with 100 variables. This first submission was settled within a few hours after the download of the challenge data and got a score of 0.9468 on the leaderboard, second behind the leader (0.9476) on 2014-04-18.

[2] Available as a shareware at www.khiops.com

C. Additional experiments

As this first result was promising, we decided to proceed with further optimizations. The challenge evaluation criterion (score) is a mean AUC minus a penalty. The mean AUC should be above 0.5 (default performance). The penalty in the challenge is quadratic w.r.t the number of selected variables $|s|$ according to

$$penalty(s) = (\frac{|s| - 30}{1000})^2.$$

It is 0 with 30 variables and reaches 0.5 with 737 variables. With 100 variables, we got a penalty of 0.005 and therefore a leaderboard AUC of about 0.952, which is in line with our expectation. There might be room for some improvement, by optimizing directly the challenge evaluation criterion.

Algorithm 1 Bachward variable selection

Require: $X = (X_1, X_2, \dots X_K)$ {Set of input variables}
Ensure: S_{Best} {Best subset of variables}
$S = X, S_{Best} = X$ {Start with all the input variables}
{Backward selection}
while $|S| > 30$ **do**
  {Select best variable to remove}
  for $X_k \in S$ **do**
    if $(score(S - \{X_k\}) < score(S))$ **then**
      $X_{Remove} = X_k$
    end if
  end for
  {Update selection}
  $S = S - \{X_{Remove}\}$
  if $(score(S) < score(S_{Best}))$ **then**
    $S_{Best} = S$
  end if
end while

We then started from our subsets of MAP variables for each of the three classes, augmented with MAP variables resulting from the training of the three classes simultaneously ($AllClass = Concat(Class1, Class2, Class3)$). We obtained a starting set of 103 distinct variables. We then used a standard variable backward elimination algorithm based on a direct optimization of the challenge criterion on all the dataset. This variable selection method is summarized in Algorithm 1. At each step, it evaluates each variable elimination, then removes the variable that brings the best score. The algorithm returns the best subset of variables found during optimization.

In Figure 3, we report the AUC per class and the mean AUC obtained along the optimization path, from 103 variables down to 30 variables. The very few first optimization steps eliminate redundant variables, and improve the mean AUC from 0.953 to 0.957 with 95 variables. We then have a long plateau until getting 65 variables, and finally a slow decrease in

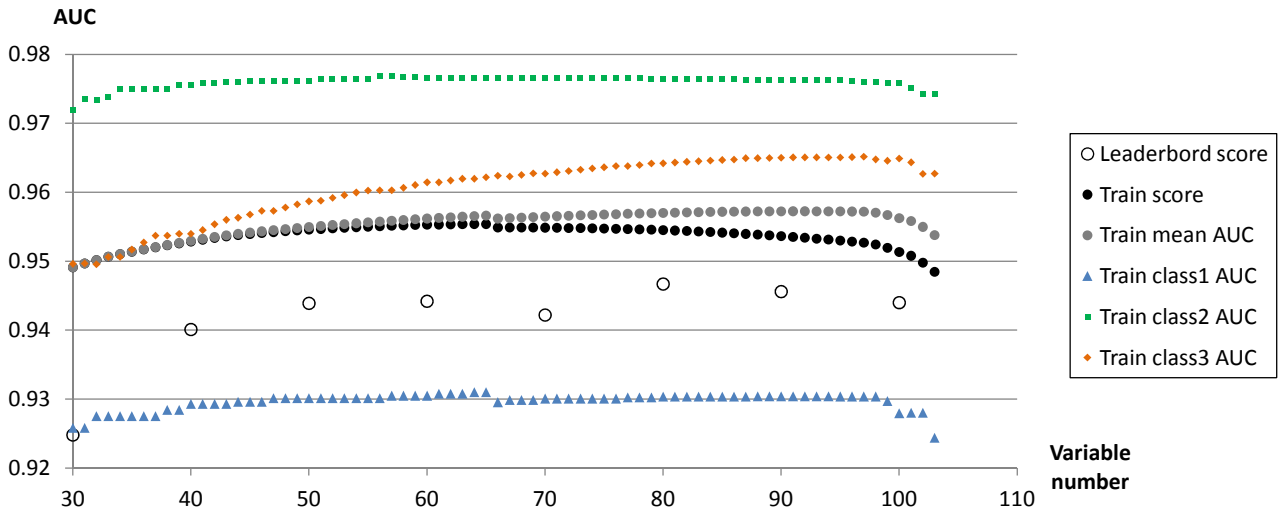

Fig. 3. Sensibility analysis: AUC versus number of selected variables

AUC, with a final AUC of 0.949 with 30 variables. The shape of the AUC curves is similar to that of Figure 2, which shows three AUC results with selections from one single variable to thousands of variables. The AUC increases quickly with very few variables (in Figure 2, one single variable is sufficient to goes from an AUC of 0.5 to around 0.75). Then, after a few tens of variables, a plateau is reached, and finally, for large numbers of variables, the performance decreases significantly down to the performance of the NB classifier (0.89 on average in Figure 2). In this bi-criterion problem, the beginning of the curve presented in Figure 3 can be interpreted as a Pareto curve, where each point corresponds to an optimal AUC given a max number of selected variables. In real data mining projects, this kind of curve might be helpful to find the best trade-off between accuracy and number of selected variables, according to the requirement and constraints of the project.

In the challenge, the score includes a penalty to choose the best trade-off. The challenge score is reported with black circles in Figure 3. The best score shown in Figure 3 gets a 0.5% improvement (up to a train score of 0.955 with 65 variables). This improvement is rather small and might be prone to overfitting, with an expected increased variance as the number of variables decreases. We got a score of 0.9452 on the leaderboard with this optimized solution. We tried other random partitions of the same variables into ten subsets (for the ten Naive Bayes ensemble classifier) and obtained score variations of about 0.3%. We also submitted a series of variable sets of increasing size along our optimization path, from 30 variables up to 100 variables by steps of ten. The resulting leaderboard scores (reported using white circles in Figure 3) shows that the improvements obtained during the train optimization vanish withing the variance of the results on the challenge leaderboard dataset. Furthermore, within a same set of selected variables, different random partitions in ten subsets produce a variance of the results of similar magnitude.

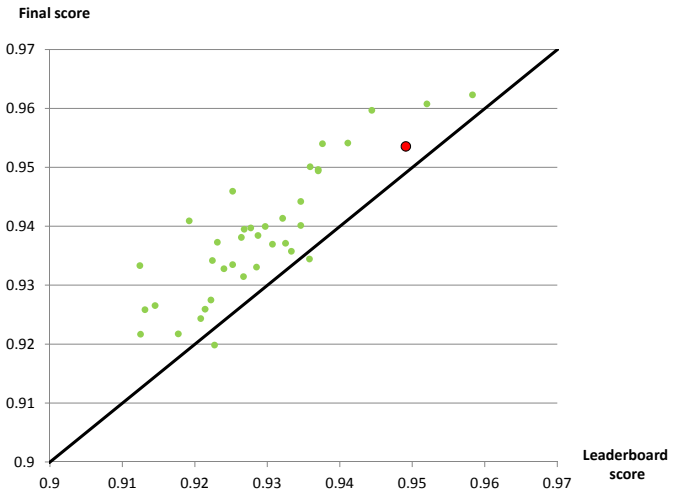

Fig. 4. Final challenge results

D. Final submission

The sensibility analysis of performance versus number of selected variables in Section III-C shows that the initial submission (see Section III-B) is competitive w.r.t. the objective of the challenge. Given the expected small and unreliable improvement in challenge score, the potential risk of using too few variables (larger expected variance) and the ignorance regarding the behavior of the ensemble classifier used in the challenge, we finally came back to the first submission. We removed the duplicate variables (keeping 97 variables), and obtained a challenge leaderboard score of 0.9491.

E. Challenge results

The last day of the challenge, our submission was ranked 3rd on the leaderboard. Usually, in data mining challenges, the participants tend to overfit the leaderboard score with many submissions, and we anticipated to get lower scores in the final results. Figure 4 shows the leaderboard versus final scores of all participants that obtained a score beyond that of the organizer's baseline; our score is represented by the red circle. Our final score (0.9536) was improved by 0.5% compared to the leaderboard score, which was a good surprise. Overall in this challenge, the final scores where improved on average by 1%. Participants ranked 4^{th} to 6^{th} on the leaderboard dataset got a 1.5% improvement of their final score and we finally got ranked 6^{th} in the final evaluation, 0.9% behind the winner.

IV. Conclusion

In most data mining projects, specific business requirements and constraints must be fulfilled. Several criterions must be taken into account, such as the time spend for the project, the training time, the deployment time, the intepretability of the models, the predictive accuracy. The AAIA'14 Data Mining Competition was an interesting challenge that focused on predictive accuracy versus number of selected variables. We have shown that using the Selective Naive Bayes classifier allows to quickly and efficiently obtain a competitive solution. We have also presented a sensitivity analysis between the two challenge criterions, that presents all possible trade-offs along a Pareto curve. This kind of analysis might be helpful to fulfill requirements in real world data mining projects.

References

[1] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[2] M. Boullé, "Compression-based averaging of selective naive Bayes classifiers," *Journal of Machine Learning Research*, vol. 8, pp. 1659–1685, 2007.

[3] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers," in *10th National Conference on Artificial Intelligence*. AAAI Press, 1992, pp. 223–228.

[4] D. Hand and K. Yu, "Idiot bayes ? not so stupid after all?" *International Statistical Review*, vol. 69, no. 3, pp. 385–399, 2001.

[5] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Proceedings of the 12th International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 1995, pp. 194–202.

[6] H. Liu, F. Hussain, C. Tan, and M. Dash, "Discretization: An enabling technique," *Data Mining and Knowledge Discovery*, vol. 4, no. 6, pp. 393–423, 2002.

[7] M. Boullé, "MODL: a Bayes optimal discretization method for continuous attributes," *Machine Learning*, vol. 65, no. 1, pp. 131–165, 2006.

[8] ——, "A Bayes optimal approach for partitioning the values of categorical attributes," *Journal of Machine Learning Research*, vol. 6, pp. 1431–1452, 2005.

[9] P. Langley and S. Sage, "Induction of selective Bayesian classifiers," in *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1994, pp. 399–406.

[10] R. Kohavi and G. John, "Wrappers for feature selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[11] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[12] J. Hoeting, D. Madigan, A. Raftery, and C. Volinsky, "Bayesian model averaging: A tutorial," *Statistical Science*, vol. 14, no. 4, pp. 382–417, 1999.