Physiological Data Modeling Contest

João Gama JGAMA@LIACC.UP.PT
LIACC - University of Porto
Rua Campo Alegre 823, 4150 Porto, Portugal

Pedro Rodrigues PRODRIGUES@DCC.FC.UP.PT
LIACC - University of Porto
Rua Campo Alegre 823, 4150 Porto, Portugal

1. Introduction

The physiological data is characterized by *large amounts of data, sequential data, issues of sensor fusion, and a rich domain complete with noise, hidden variables, and significant effects of context.* There is a continuous flow of data, and the problem is to built a predictive model from the data stream. Several authors refer desirable properties for mining data streams: incremental algorithms, able to process examples in constant time and memory, performing a single scan over the training data, maintaining classifiers at any time, and dealing with concept drift.

Our recent work focus on induction of decision trees from data streams. The possibility to evaluate the system in a real problem is the main motivation to participate in the Workshop.

The paper is organized as follows. The next section presents a short overview of the system. The Section 3 presents evaluation results of the system in this problem. The last section resumes the main aspects of the applicability of the algorithm in this problem.

2. Ultra-Fast Forest Trees - UFFT

The UFFT is an algorithm for supervised classification learning, that generates a forest of binary trees. The algorithm is incremental, processing each example in constant time, works on-line, and uses the Hoeffding bound to decide when to install a splitting test in a leaf leading to a decision node. UFFT is designed for continuous data. It uses analytical techniques to choose the splitting criteria, and the information gain to estimate the merit of each possible splitting-test. For multi-class problems, the algorithm builds a binary tree for each possible pair of classes leading to a forest-of-trees. During the training phase the algorithm maintains a short term memory. Given a data stream, a limited number of the most recent examples are maintained in a data structure that supports constant time insertion and deletion. When a test is installed, a leaf is transformed into a decision node with two descendant leaves. The sufficient statistics of the leaf are initialized with the examples in the short term memory that will fall at that leaf. The UFFT has shown good results with several problems and large and medium datasets.

In the following sections we provide detailed information about the most relevant aspects of the system.

The Splitting Criteria The UFFT starts with a single leaf. When a splitting test is installed at a leaf, the leaf becomes a decision node, and two descendant leaves are generated. The splitting test has two possible outcomes each conducting to a different leaf. The value *True* is associated with one branch and the value *False*, with the other. The splitting tests are over a numerical attribute and are of the form $attribute_i \leq value_j$. We use the analytical method for split point selection presented in [6]. We choose, for all numerical attributes, the most promising $value_j$. The only sufficient statistics required are the mean and variance per class of each numerical attribute. This is a major advantage over other approaches, as the exhaustive method used in C4.5 [7] because all the necessary statistics are computed on the fly. This is a desirable property on the treatment of huge data streams because it guarantees constant time processing each example.

The analytical method uses a modified form of quadratic discriminant analysis to include different variances on the two classes[1]. This analysis assumes that the distribution of the values of an attribute

[1]The reader should note that in UFFT any n-class problem is decomposed into $n(n-1)/2$ two-class problems.

follows a normal distribution for both classes. Let $\phi(\bar{x}, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\bar{x})^2}{2\sigma^2}\right)$ be the normal density function, where $\bar{x}$ and σ^2 are the sample mean and variance of the class. The class mean and variance for the normal density function are estimated from the sample set of examples at the node. The quadratic discriminant splits the X-axis into three intervals $(-\infty, d_1)$, (d_1, d_2), (d_2, ∞) where d_1 and d_2 are the possible roots of the equation $p(-)\phi\{(\bar{x}_-, \sigma_-)\} = p(+)\phi\{(\bar{x}_+, \sigma_+)\}$ where $p(i)$ denotes the estimated probability than an example belongs to class i. We pretend a binary split, so we use the root closer to the sample means of both classes. Let d be that root. The splitting test candidate for each numeric attribute i will be of the form $Att_i \leq d_i$. To choose the best splitting test from the candidate list we use an heuristic method. We use the information gain to choose, from all the splitting point candidates, the best splitting test. To compute the information gain we need to construct a contingency table with the distribution per class of the number of examples lesser and greater than d_i. Once the merit of each splitting has been evaluated, we have to decide on the expansion of the tree. This problem is discussed in the next section.

From Leaf to Decision Node To expand the tree, a test $attribute_i \leq d_i$ is installed in a leaf, and the leaf becomes a decision node with two new descendant leaves. To expand a leaf two conditions must be satisfied. The first one requires the information gain of the selected splitting test to be positive. That is, there is a gain in expanding the leaf against not expanding. The second condition, it must exist statistical support in favor of the best splitting test which is asserted using the Hoeffding bound as in VFDT [1].

Functional Leaves To classify an unlabeled example, the example traverses the tree from the root to a leaf. It follows the path established, at each decision node, by the splitting test at the appropriate attribute-value. The leaf reached classifies the example. The classification method is a *naive-Bayes* classifier. The use of the naive-Bayes classifiers at the tree leaves does not enter any overhead in the training phase. At each leaf we maintain sufficient statistics to compute the information gain. These are the necessary statistics to compute the conditional probabilities of $P(x_i|Class)$ assuming that the attribute values follow, for each class, a normal distribution. Let l be the number of attributes, and $\phi(\bar{x}, \sigma)$ denotes the standard normal density function for the values of attribute i that belong to a given class. Assuming that the attributes are independent given the class, the Bayes rule will classify an example in the class that maximizes the *a posteriori* conditional probability, given by: $P(C^i|\vec{x}) \propto \log(Pr(C^i)) + \sum_{k=1}^{l} \log(\phi(\bar{x}_k^i, s_k^i))$. There is a simple motivation for this option. UFFT only changes a leaf to a decision node when there is a sufficient number of examples to support the change. Usually hundreds or even thousands of examples are required. To classify a test example, the majority class strategy only use the information about class distributions and does not look for the attribute-values. It uses only a small part of the available information, a crude approximation to the distribution of the examples. On the other hand, naive-Bayes takes into account not only the prior distribution of the classes, but also the conditional probabilities of the attribute-values given the class. In this way, there is a much better exploitation of the information available at each leaf [3].

Forest of Trees The splitting criteria only applies to two class problems. Most of real-world problems are multi-class. In the original paper [6] and for a batch-learning scenario, this problem was solved using, at each decision node, a 2-means cluster algorithm to group the classes into two super-classes. Obviously, the cluster method can not be applied in the context of learning from data streams. We propose another methodology based on round-robin classification [2]. The round-robin classification technique decomposes a multi-class problem into k binary problems, that is, each pair of classes defines a two-classes problem. In [2] the author shows the advantages of this method to solve n-class problems. The UFFT algorithm builds a binary tree for each possible pair of classes. For example, in a three class problem (A,B, and C) the algorithm grows a forest of binary trees, one for each pair: A-B, B-C, and A-C. In the general case of n classes, the algorithm grows a forest of $\frac{n(n-1)}{2}$ binary trees. When a new example is received during the tree growing phase each tree will receive the example if the class attached to it is one of the two classes in the tree label. Each example is used to train several trees and neither tree will get all examples. The short term memory is common to all trees in the forest. When a leaf in a particular tree becomes a decision node, only the examples corresponding to this tree are used to initialize the new leaves.

Fusion of Classifiers When doing classification of a test example, the algorithm sends the example to all trees in the forest. The example will traverse the tree from root to leaf and the classification is registered.

Each tree in the forest make a prediction. This prediction takes the form of a probability class distribution.

Taking into account the classes that each tree discriminates, these probabilities are aggregated using the *sum rule* [5]. The most probable class is used to classify the example. Note that some examples will be forced to be classified erroneously by some of the binary base classifiers, because each classifier must label all examples as belonging to one of the two classes it was trained on.

Functional Inner-nodes When evaluating the splitting-criteria the merit of the best attributes could be closed enough that the difference in gains does not satisfy the Hoeffding bound. This aspect has been pointed out in [1]. In VFDT, the authors propose the use of a user defined constant, τ, that can decide towards a split (given that $\epsilon < \tau$), even when the Hoeffding bound is not satisfied. In UFFT when there is a tie in the evaluation of the merit of tests based on single attributes, the system starts trying more complex splitting tests [4].

The sufficient statistics for the splitting-criteria can be directly used to construct a naive-Bayes classifier. The idea of functional inner nodes is to install splitting-tests based on the predictions of the naive-Bayes classifier build at that node.

Suppose that we observe a leaf where the differences in gain between the two best attributes does not satisfies the Hoeffding bound. Since the first tie, when a new training example fall at this leaf, it will be classified using the naive-Bayes derived from the sufficient statistics. Those predictions are used to populate the 2×2 contingency table. In the next evaluation step we evaluate also, in addiction to the evaluation of all the original attributes, the information gain of the contingency table obtained by the naive-Bayes predictions. This evaluation corresponds to consider a new attribute: the naive-Bayes predictions. If this implicit attribute is the best attribute in terms of information gain, and the difference with respect to the second best satisfies the Hoeffding bound, then the leaf becomes a decision node with two outcomes: the naive-Bayes predictions.

Naive Bayes classifiers use all attributes to make predictions. This aspect could be negative in the presence of irrelevant attributes. In UFFT we only consider splitting-tests based on naive-Bayes classifiers after the first tie. This aspect can be used to select the most informative attributes to use with naive-Bayes.

3. The Physiological Data Modeling Problem

For all the three problems we have used the following attributes: characteristic[1..2], and sensor[1..9].

The tasks:

1. The first task is to return a column of predictions for the test set for the gender for every sessionID.
2. The second task is to correctly identify when a person is participating in context 1, signified by annotations with the value 3004.
3. The third and final task is to correctly identify when a person is participating in context 2, signified by annotations with the value 5102.

3.1. Task 1

To evaluate our algorithm we sample from the labelled set training sets of size 500000. The generated classifier was evaluated in the complete set of labelled examples and in the remainder 80264 examples. Please note that UFFT process once each example, reducing overfitting problems. Even the error on the training set is a good approximation of the generalization error.

```
Training Time 39 seconds,
Error Rate: 0.004784 (2776  580264)
        0       1
0       556335  2776
1       0       21153
```

3.2. Task 2 and 3

We present the evaluation only for Task 2, similar results were obtained for Task 3. In these tasks, we have considered several alternatives for the target attribute.

1. 2-class problem: Positive examples of context versus Negative examples (including the could be)
2. 3 classes problem: Positive, Negative, and Could_Be

In both alternatives there was a skew class distribution, mostly in task 2. Although low error rate, the Positive class was very difficult to predict:

```
Training Time 234 seconds
Error Rate: 0.003526 (283 / 80265)
        0       1
0       79982   228
1       55      0
```

This aspect motivated us to consider misclassification costs. We have used the following cost matrix:

```
  N    P
N 0    10
P 0.1  0
```

Predictions were post-processed in order to guarantee sequences of Positive predictions.

The following table resumes the sequences obtained in the training data:

```
Task 2:
             Nr.seq Mean_size Min_Size Max_Size
Prediction    2992       12.7        3      154
Observed        75         57        6      177

Task 3:
             Nr.seq Mean_size Min_Size Max_Size
Prediction      795    30.3          3      516
Observed         37   301.4         65      592
```

The confusion matrix is

```
Task 2:
  N        P
N 539898  2317
P  35953  2096

Task 3:
    N      P
N  53425  2472
P  13708 10659
```

Another interesting aspect for this task is learning time. The training time to built a model using all the training data was around 4 minutes.

4. Conclusions

There are several advantages of using a system like UFFT in this type of problems. The algorithm is incremental, works online, takes misclassification costs into account, maintains accurate classifiers at any time, and can detect drift (a feature not used in this task). It generate a single (in that case where only two classes are involved) and interpretable model in a form of a decision tree. Results on the training data seems promising, but the important are the test set results.

Acknowledgments

This work was developed in the context of the project ALES (POSI/SRI/39770/2001).

References

[1] P. Domingos and G. Hulten. Mining high-speed data streams. In Ismail Parsa, Raghu Ramakrishnan, and Sal Stolfo, editors, *Proceedings of the Six International Conference on Knowledge Discovery and Data Mining*, pages 71–80. ACM Press, 2000.

[2] J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002.

[3] J. Gama, R. Rocha, and P. Medas. Accurate decision trees for mining high-speed data streams. In P.Domingos and C. Faloutsos, editors, *Procs. of the 9th ACM SigKDD Int. Conference in Knowledge Discovery and Data Mining*. ACM Press, 2003.

[4] João Gama. Functional trees. *Machine Learning*, (to appear), 2004.

[5] J. Kittler. Combining classifiers: A theoretical framework. *Pattern analysis and Applications, Vol. 1, No. 1*, 1998.

[6] W.-Y. Loh and Y.-S. Shih. Split selection methods for classification trees. *Statistica Sinica*, 1997.

[7] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., 1993.