International Conference on Control, Engineering & Information Technology (CEIT'13)
Proceedings Engineering & Technology - Vol.1, pp. 207-211, 2013
Copyright - IPCO

Conception of Real-Time Flood Forecasting System Approach Based on Anytime Techniques

El Mabrouk Marouane, Cherrat Loubna, Ezziyyani Mostafa, Essaaidi Mohammad

laSIT Physics, UAE
Tetuan, Morocco
elmabroukmarouane@gmail.com
cherrat81@yahoo.fr

laSIT Physics, UAE
Tetuan, Morocco
ezziyyani@gmail.com
essaaidi@ieee.com

***Abstract*— In a Decision Support System (DSS), a computer system must allow the decision maker to make the best possible decision, often before a given time. The objective of our project is to extend the notion of flood forecasting in an environment constrained by time, it means to consider the real-time aspects of a forecast DSS. To design such a system, we opted for the use of "Anytime techniques" that aimed at giving information systems the possibility to provide results which quality and accuracy evolve according to a given time. In this paper, we suggest ANY2FC (Anytime Flood Forecasting), a model of real-time flood forecasting based on Anytime techniques.**

***Keywords*— flood ; forecasting ; real-time ; decision tree ; machine Learning ; Anytime algorithm.**

I. Introduction

The growth of natural and technological hazards is a worldwide phenomenon that comes as a result of the industrialization and the increasing density of occupation of risk areas, prone to hazards or hazardous events. Risks, therefore, pose a global challenge for the future and they're one of the major problems that sustainable development may face.

In terms of natural hazards, Morocco, by its geographical location, is exposed to climatic, meteorological, geological and biological changes which may lead to major risks such as floods, flash floods, earthquakes and landslides etc.

Here comes the need for a kind of decision support in doing the real-time flood forecasting which may give the opportunity to decide before the normal end of treatment because the current algorithms don't provide the answer until the end of treatment. The former will ensure the quality of the decision accuracy of data, the speed of learning time and the execution and response. Therefore, in this paper, we propose a new algorithm that optimizes the execution time, the precision of the decision and response time.We based our approach on Anytime algorithms to produce partial decisions before the end of treatment trees.

II. State of Art of the Algorithms of Constructing the Decision Trees

The majority of algorithms dealing with decision trees construction follow the same principles; with exceptions concerning the ability of each to provide binary trees or n-ary, the processing capacity of input attributes in terms of density and quality of accuracy.

Some algorithms are characterized by the rapidity of providing answers on small samples such as CART, some are characterized by quick decisions regarding means samples such as C4.5 and ID3 while others are characterized by their rapidity of large samples as CHAID.

The problem of these algorithms is that their performance decreases when the size of the database processed and the number of classes increase.

In the next section, we present the basic decision algorithms make in literature.

A. Classification and Regression Trees (CART)

CART (Classification and Regression Trees), is characterized by the fact that it constructs binary trees, namely each internal node has exactly two outgoing edges whereas both ID3, C4.5 algorithms generate the decision trees with variable branches per node. CART is unique from other Hunt's based algorithm as it is also used for regression analysis with the help of regression trees. The regression analysis feature is used in forecasting a dependent variable (result) given a set of predictor variables over a given period of time. The CART decision tree is a binary recursive partitioning procedure capable of processing continuous and nominal attributes both as targets and predictors. In CART trees are grown, using genie index for splitting procedure, to a maximum size without the use of a stopping rule and then pruned back (essentially split by split) to the root via cost-complexity pruning. The CART mechanism is intended to produce not one, but a sequence of nested pruned trees, all of which are candidate optimal trees. The CART mechanism includes automatic (optional) class balancing, automatic missing value handling, and allows for cost-sensitive learning, dynamic feature construction, and probability tree estimation.

B. Iterative Dichotomiser 3 (ID3)

ID3 (Iterative Dichotomized) algorithm is based on the Concept of Learning System (CLS) algorithm. CLS algorithm is the basic algorithm for decision tree learning. The tree growth phase of CLS is the matter of choosing attribute to test each node by the trainer. ID3 improves CLS by adding a heuristic for attribute selection. ID3 is based on Hunt's algorithm and is implemented serially. This algorithm recursively partitions the training dataset till the record sets belong to the class label using depth first greedy technique. In growth phase of the tree construction, this algorithm uses information gain, an entropy based measure, to select the best splitting attribute, and the attribute with the highest information gain is selected as the splitting one. ID3 doesn't give accurate result when there is too much noise or details in the training data set, thus an intensive pre-processing of data is carried out before building a decision tree model with ID3. One of the main drawbacks of ID3 is that the measure Gain used tends to favor attributes with a large number of distinct values. It accepts only categorical attributes in building a tree model from which algorithm generates various branches per node.

C. C4.5

C4.5 algorithm is an improved version of ID3, it uses Gain Ratio as a splitting criteria, instead of taking gain in ID3 algorithm for splitting criteria [14] in tree growth phase. Hence C4.5 is an evolution of ID3. This algorithm handles both continuous and discrete attributes- In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it. Like ID3 the data is sorted at every node of the tree in order to determine the best splitting attribute. The splitting ceases when the number of instances to be split is below a certain threshold. The main advantages of C4.5 is when building a decision tree, C4.5 can deal with datasets that have patterns with unknown attribute values. C4.5 can also deal with the case of attributes with continuous domains by discretization. This algorithm handles training data with attribute values by allowing attribute values to be marked as missing. Missing attribute values are simply not used in gain and entropy calculations. It has an enhanced method of tree pruning that reduces misclassification errors due to noise or too much detail in the training data set.

D. Scalable Parallelizable Induction of decision Tree algorithm (SPRINT)

SPRINT (Scalable Parallelizable Induction of decision Tree algorithm) was introduced by Shafer et al, 1996. It is a fast, scalable decision tree classifier. It is not based on Hunt's algorithm in constructing the decision tree, rather it partitions the training data set recursively using breadth-first greedy technique until each partition belong to the same leaf node or class (Anyanwu et al, 2009 and Shafer et al, 1996). It is an enhancement of SLIQ as it can be implemented in both serial and parallel pattern for good data placement andload balancing (Shafer et al, 1996). In this paper we will focus on the serial implementation of SPRINT. Like SLIQ it uses one time sort of the data items and it has no restriction on the input data size. Unlike SLIQ it uses two data structures: attribute list and histogram which is not memory resident making SPRINT suitable for large data set, thus it removes all the data memory restrictions on data (Shafer et al, 1996). It handles both continuous and categorical attributes.

III. EXPERIMENTAL RESULTS AND COMPARATIVE STUDIES

A. Study domain

To compare the algorithm's performance of construction of the decisions trees, we conducted five experiments, each time increased the number of records, the number of classes and the number of attributes in order to identify problems that may appears during execution. Here is the table that shows the experiences already realized.

TABLE I. STATLOG AREA

	No. of attributes	*No. of classes*	*No. of records*
E_1	24	2	270
E_2	11	4	690
E_3	8	2	753
E_4	6	2	1000
E_5	4	7	10500

B. Results

TABLE II. EXECUTION TIME TO BUILD MODEL

	ID3	*CART*	*C4.5*	*SPRINT*
E_1	0.12secs	0.61secs	0.05secs	0.03secs
E_2	0.31secs	1.64secs	0.23secs	0.15secs
E_3	0.47secs	2.17secs	0.35secs	0.26secs
E_4	1.08secs	11.41secs	0.52secs	0.41secs
E_5	2.08secs	38.31secs	1.17secs	0.89secs

TABLE III. CLASSIFICATION ACCURACY

	ID3	*CART*	*C4.5*	*SPRINT*
E_1	35.2%	56.67%	76.7%	80%
E_2	54.3%	65%	66.5%	67%
E_3	32.1 %	70 %	69.2 %	70 %
E_4	71.5 %	85.4 %	84.2 %	85.8 %
E_5	99.2 %	94 %	98.00 %	99.63 %

Table II shows that SPRINT classifiers have the fastest execution time among all the classifiers, regardless of the class size, number of attributes and records of the data sets volume. This is closely followed by C4.5. The table also showed that execution time for ID3 is faster than CAR. However, CART is preferred by researches and scientists as it handles both categorical and continuous attributes while ID3 does not.

Table III shows that SPRINT classifier has the highest classification accuracy among all the classifiers, this is followed by C4.5. The class size, attribute number and record number do not affect the classification accuracy of SPRINT and C4.5 compared to other classifiers. The classification accuracy of the ID3 and CART classifiers depends to a large extent on the class size, attribute number and record number of the data sets.

As shown in Table III for a large data set, the classification accuracy of ID3 is better than that of CART as ID3 has a high accuracy for large data that have been pre-processed and loaded into the memory at the same time. But for small-medium data, the classification accuracy of CART is more accurate than that of ID3.

C. *Synthesis*

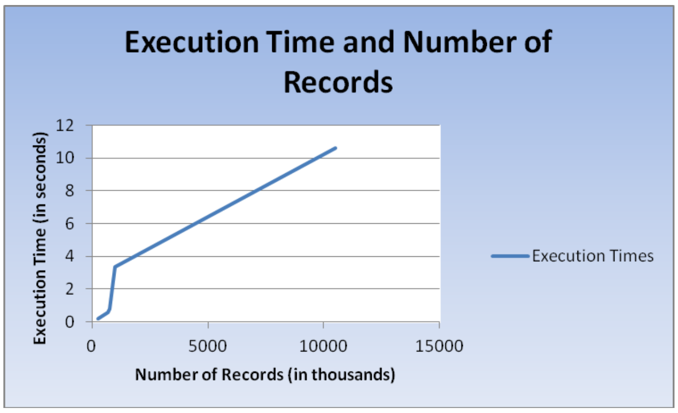

Fig. 1. Execution Time and Number of Records

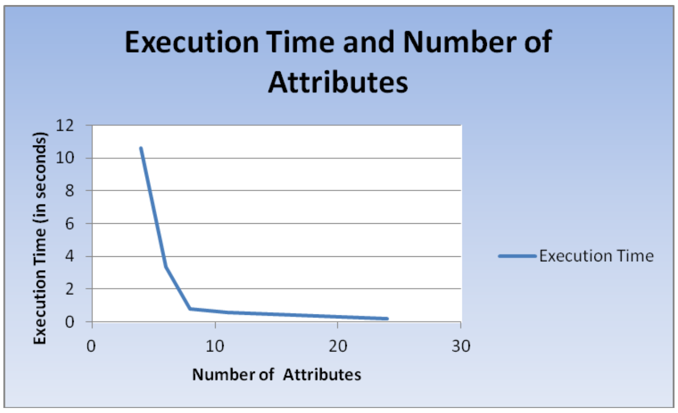

Fig. 2. Execution Time and Number of Attributes

Figure 1. shows that for all the classifiers, the execution time increases as the number of records increases. The steady portion of the graph is the effect of varying the number of attributes of the classifiers. Also Figure 2. shows that the execution time (time to build the model) of the classifiers decreases as the attributes of the classifiers increases and becomes steady at some point, due to the change in the number of records of the data sets and also class size change.

IV. ANY2FC (ANYTIME FLOOD FORECASTING) MODEL

The algorithms of construction of the decision trees existing in the literature requires the end of learning (the end of construction of the tree) to give a decision, which according to the experimental studies already conducted, as the number of records and classes increases, the execution time also increases, hence the necessity to stop the construction and give partial decisions which may be the final one, is essential. Anytime algorithms provide us this possibility.

A. *Anytime algorithms*

Anytime algorithm is an algorithm that can find a valid solution to a problem even if it's interrupted at any time before it ends. The algorithm is expected to find better solutions as long as it keeps running. Most algorithms run to completion: they provide a single answer after performing some fixed amount of computation.

However, in some cases, the user may wish to terminate the algorithm prior to completion. The amount of the computation required may be substantial, for example, and computational resources might need to be reallocated. Most algorithms either run to completion or they provide no useful solution information. Anytime algorithms, however, are able to return a partial answer, according to the quality which depends on the amount of computation they were able to perform. The answer generated by anytime algorithms is an approximation of the correct answer.

B. *New proposed Model ANY2FC*

According to the synthesis already presented on the experimental study, the execution time of the algorithms increases when the number of records is larger and the number of attributes is smaller.

Projecting that's already seen on our application domain which is flooding, according to the large number of records with a small number of attributes in this scope, you can't use any of the existing algorithms in the literature, so you should have an optimal solution that takes into consideration these two constraints. Here comes the need to suggest a new algorithm which is based on the latter. In the following section, we present the ANY2FC Model for real-time forecasting of floods.

ANY2FC (Anytime Flood Forecasting) is a model based on the Anytime's concept to interrupt the treatment before its normal end in order to provide a decision about the impact of flooding. The quality of the decision depends on the quality of the precised shaft portion built in order to save execution time and response to lead to the notion of real-time forecasting.

This model will be based on a parameter called the cost of importance of a factor. Factors are the parameters leading to the incidence of flooding. We distinguish between two types of factors, the dominating factors on a region and the secondary factors.

C. Cost of importance

The cost of importance C is a parameter which can stop the growth of the tree to be able to decide. To calculate this parameter we proceed to the following equation:
$f(C) = \alpha X + \beta$ with $X \in]0, +\infty[$, $X = 1/\text{Value(Factor)}$ and $(\alpha, \beta) \in \{0,1\}$

D. ANY2FC Model

The model is triggered when the cost of importance of the factors start to decrease. If the dominant factors are null then there will be no flood. The system will not be triggered if there is less dominant factor. Here is the algorithm of the model.

ANY2FC Procedure

```
Procedure ANY2FC ()
   Array Factors_List;
   Integer Factor_Type ;
   If ( Number_Of_Important_ Factors () > 1 ) Then
   If ( It's the dominant factor ) Then Factor_Type ← 1;
   Else Factor_Type ← 0;
      If (Decrease_Cost_Factor(Factor_Type, C) = True )
Then
          Machine Learning;
      Else
          No Flood;
      EndIf
   EndIf
 EndProcedure
```

Fig. 3. ANY2FC Procedure

This function consists of counting the number of factors that have costs of importances in order to trigger the model.

Number_Of_Important_ Factors Function

```
Function Number_Of_Important_ Factors ()
   Number ← 0;
   C ← Null ;
   Dominant_Factor ← Null ;
   Decrease_Cost ← Null ;
   Foreach Factor
      If ( It's the dominant factor on the region ) Then
         C ← $\alpha X + \beta$ with $\alpha \leftarrow 1, \beta \leftarrow 1$
         Sleep (20 Seconds) ;
         Dominant_Factor ← 1;
         Decrease_Cost← Decrease_Cost_Factor
(Dominant_Factor, C);
         If ( Decrease_Cost = True ) Then
            Factors_List [Number] ← Factor ;
            Number ← Number + 1 ;
         EndIf
      Else
         C ← $\alpha X + \beta$ with $\alpha \leftarrow 1, \beta \leftarrow 0$
         Sleep (20 Seconds) ;
         Dominant_Factor ← 0;
         Decrease_Cost← Decrease_Cost_Factor
(Dominant_Factor, C);
         If ( Decrease_Cost = True ) Then
            Factors_List [Number] ← Factor ;
            Number ← Number + 1 ;
         EndIf
      EndIf
   EndFor
   Return Number;
 EndFunction
```

Fig. 4. Number_Of_Important_ Factors Function

This function is used to test the decreasing costs of importances of factors.

Decrease_Cost_Factor Function

```
Function Decrease_Cost_Factor (Dominant_Factor, C1)
   C2 ← Null ;
   Result ← False ;
   If (Dominant_Factor = 1 ) Then
      C2 ← $\alpha X + \beta$ with $\alpha \leftarrow 1, \beta \leftarrow 1$
   Else
      C2 ← $\alpha X + \beta$ with $\alpha \leftarrow 1, \beta \leftarrow 0$
   EndIf
   If ( C2 < C1 ) Then
      Result ← True ;
   EndIf
   Return Result;
EndFunction
```

Fig. 5. Decrease_Cost_Factor Function

V. Conclusion & perspectives

In this paper, we have presented the ANY2FC model of real-time flood forecasting that allows reducing the execution time of making the decision when we have big Databases. In addition, we have presented the Cost of importance parameter C and how to calculate it.

In terms of perspectives, it will be studied how to identify the dominant parameters that will be negotiated with the forecasting system to be processed in order to give the final decision. The studies will be based on the principle of computational processing offline and online.

VI. References

[1] A. Srivastava, E. Han, V. Kumar, V. Singh, 1998, "Parallel Formulations of Decision-Tree Classification Algorithms". International Conference on Parallel Processing (ICPP'98). pp 237.

[2] Andrew Colin, 1996, “Building Decision Trees with the ID3 Algorithm", Dr. Dobbs Journal, June 1996.

[3] Breiman, Leo, Friedman, J. H. , Olshen, R. A. , & Stone, C. J, 1984, “Classification and regression trees, Monterey”, CA: Wadsworth & Brooks/Cole Advanced Books & Software, ISBN 978-0412048418.

[4] J.C. Shafer, R. Agrawal, M. Mehta, "SPRINT: A Scalable Parallel Classifier for Data Mining",Proc.of the 22th Int’l Conference on Very Large Databases, Mumbai (Bombay), India, Sept. 1996.

[5] J.R. Quinlin. Induction of décision trees. Machine Learning, vol. 1, pages 81 - 106, 1989.

[6] J. R. Quinlan, 1996, “Improved use of continuous attributes in C4.5” , Journal of Artificial Intelligence Research, Vol. 4, pp. 77-90.

[7] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufman, 1993.

[8] J.R. Quinlan, Discovering rules by induction from large collections of examples, D. Michie ed., Expert Systems in the Microelectronic age, pp. 168-201, 1979.

[9] Li_Tianrui, 2007, “Construction of Decision Trees based Entropy and Rough Sets under Tolerance Relation”, ISKE-2007 Proceedings part of series: Advances in Intelligent Systems Research. ISBN: 978-90-78677-04 8.

[10] M. Mehta, R. Agrawal and J. Rissanen, “SLIQ: A Fast Scalable Classifier for Data Mining”,Proc. ofthe Fifth Int’l Conference on Extending Database Technology, Avignon, France, March 1996.

[11] Ricco RAKOTOMALALA. Arbres de Décision. Revue MODULAD, 2005 N 33, pages 163 - 187.

[12] Shlomo Zilberstein, “Using Anytime Algorithmsin Intelligent Systems”. FALL, 1996 N 73, pages 73 - 83.

[13] Xin dong Wu,Vipin Kumar ,J.Ross Quinlan, Joydeep Ghosh,Qiang Yang, Hiroshi Motoda, 2007 , ”Top 10 algorithms in data mining“ , Spinger- Verlag London Limited ,pages 20 -28.

[14] Mehta, M., Agrawal, R., and Rissanen, J. (1995). MDL-based decision tree pruning. International conference on knowledge discovery in databases and data mining (KDD-95) Montreal, Canada