Enhanced Anytime Algorithm for Induction of Oblivious Decision Trees

Mark Last[1], Albina Saveliev[1]

[1]Department of Information Systems Engineering, Ben-Gurion University of the Negev, POB 653, Beer-Sheva, 84105 Israel
{mlast, albinabu}@bgu.ac.il

Abstract. Real-time data mining of high-speed and non-stationary data streams has a large potential in such fields as efficient operation of machinery and vehicles, wireless sensor networks, urban traffic control, stock data analysis etc.. These domains are characterized by a great volume of noisy, uncertain data, and restricted amount of resources (mainly computational time). Anytime algorithms offer a tradeoff between solution quality and computation time, which has proved useful in applying artificial intelligence techniques to time-critical problems. In this paper we are presenting a new, enhanced version of an anytime algorithm for constructing a classification model called Information Network (IN). The algorithm improvement is aimed at reducing its computational cost while preserving the same level of model quality. The quality of the induced model is evaluated by its classification accuracy using the standard 10-fold cross validation. The improvement in the algorithm anytime performance is demonstrated on several benchmark data streams.

Keywords: anytime algorithms, classification, information theory, Information Network algorithm, classification accuracy, computation cost

1 Introduction

Systems that deal with continuous data streams are becoming increasingly important primarily due to the emergence of sensors and similar small-scale embedded computing devices that continuously produce large volumes of data they obtain from their environment. The complex nature of real-world, streaming data has increased the difficulties and challenges of data mining applications in terms of knowledge induction and decision making within the limited time scope.

Data generated by wireless sensor networks (WSN) is one of the important examples. WSN are now used in many application areas including environment and habitat monitoring, health care, home automation, and traffic control. Each sensor node of such network records as streams time-stamped observations, taken at varying time frequency. A typical observation includes measurements of various physical or environmental parameters such as temperature, sound, vibration, pressure, as well as sensor location. While real-time tracking of environmental conditions is extremely

important for handling a chemo/bio contamination, seismic detection etc., continuous transmission of *all* recorded observations by the meter-reading chips to the nearest hub node and, subsequently, to the central station may be infeasible due to the limited battery life of the chips and the local hubs. The intuitive solution is to use data-mining techniques to analyze and induce time-dependent models of observed behavior and transfer these models to the central station rather then the streamed data. At the same time, the high rate of data changes requires to generate the model rapidly within the allocated time frame.

The *anytime algorithms* give intelligent systems the capability to trade computational time for the quality of results. This capability is efficient for solving time-constrained problems such as decision making in dynamic environment, sensor interpretation, and planning [17]. The term *anytime algorithm* was introduced by Dean et al. in the mid-1980s in their work on time-dependent planning [4], [5]. Similar approaches termed *flexible computation* by Horvitz [10], [11] and *imprecise computation* by Liu et al. [15] are based on a general idea that many computational tasks are too complicated to be completed at real-time speeds, therefore it is important to build a system that can generate good approximate results in a much shorter time period.

According to Zilberstein [17], the desired properties of anytime algorithms include the following: *measurable solution quality*, which can be easily determined at run time, *monotonicity* (quality is a non-decreasing function of time), *consistency* of the quality w.r.t computation time and input quality, *diminishing returns* of the quality over time, *interruptibility* of the algorithm, and *preemptability* with minimal overhead.

In this paper, we propose a new, enhanced version of an anytime algorithm for inducing a classification model called Information Network (IN). The original algorithm was developed by Last et al. [13]. The model is a tree-like structure that represents relationship between input (predictive) features and target (classification) attributes. Unlike most other decision-tree models, the information network uses the same input attribute across all nodes of a given layer (level) and thus it can be considered an *oblivious decision-tree*. The method was shown theoretically and empirically to have the basic properties of interruptible anytime algorithms [12]. The enhanced method presented in this paper is aimed at improving the anytime performance of the IN algorithm by reducing its computational time while maintaining the same quality level of the induced model. The most time-intensive operation in network construction is choosing, at each iteration of the algorithm, an input attribute, which provides the maximum significant increase in *mutual information* relative to the previous layer. Therefore the idea is to filter out the least significant attributes, before the classifier construction, and afterwards to build a model using a reduced subset of candidate input attributes. We evaluate the performance of the algorithm on eleven benchmark datasets from various sources (see Section 4).

The paper is organized as follows. Section 2 reviews the related works in the fields of anytime classification algorithms and resource-aware knowledge discovery in data streams. The enhanced anytime algorithm for induction of oblivious decision trees is described by us in Section 3. Experimental results are presented and discussed

in Section 4. Finally we conclude the paper and present the possible future research directions in Section 5.

2 Related work

2.1 Anytime Decision Tree Induction

Last et al. [12] introduced an interruptible anytime information-theoretic classification algorithm. Their method constructs a compact and accurate decision-tree model called Information Network. The algorithm has several objectives, such as: maximizing the mutual information between a set of predictive attributes and the target (classification) attribute, finding a minimal set of features involved in the induced model (hence, it can be also used as a feature selection method), and verifying the statistical significance of the discovered patterns.

Esmeir et al. [6] presented interruptible anytime algorithms for iterative improvement of decision trees. The motivation of their research is different from our goal of saving the computational resources. They explore the problem of how to produce better decision trees for hard-to-learn concepts when more time resources are available. Their framework consists of two anytime algorithms. The first one, called *Sequencing LSID3* converts the recent LSID3 contract algorithm to an interruptible version, which does not require the allocated time in advance and can be interrupted at any time. The second is *Interruptible Induction by Iterative Improvement* (IIDT) which repeatedly selects a sub tree whose reconstruction is estimated to yield the highest marginal utility and rebuilds it, exploiting extra time allocation.

2.2 Resource-aware Data Mining Techniques

Gaber et al. [7] presented a framework for resource-aware computing in data stream analysis. The streaming information is often generated, received or processed by computational devices such as wireless sensors. These devices are limited in terms of energy, memory, computational speed and communication bandwidth. The main goal of the research is to apply data mining techniques to continuous data streams within the scope of constrained device resources. This generic framework proposes Algorithm Granularity Settings (AGS). The idea is to periodically change algorithm settings from the input, output, and/or processing end points according to resource consumption pattern measurements performed over the last time period as well as a measure of resource criticality. In [7] this method is applied to a novel threshold-based micro-clustering algorithm, called RA-Cluster. The strategy of adapting the CPU demand is done using the *Randomized Assignment* approach. As the CPU load

increases, only a pre-specified fraction of the current micro-clusters is examined when making the micro-cluster assignment for a new data point.

Phung et al. [16] extended the previous work [7] for Wireless Sensor Networks. Their approach was applied to online clustering algorithm (ERA-Cluster), which uses the resource monitoring of the Sun SPOT sensor nodes from Sun Microsystem™ to adapt to resource availability. The CPU adaptation of [16] is also based on the Randomized Assignment approach.

2.3 Anytime Properties of the IN Algorithm

If the network quality is measured by its predictive accuracy, we can easily verify the algorithm conformity with the anytime properties defined by Zilberstein [17] using a line of arguments similar to [12]:

- *Measurable quality*. The predictive accuracy after each iteration of the algorithm can be estimated using 10-fold cross-validation or any other validation procedure.
- *Recognizable quality*. Due to the inherent compactness of IN models, counting the number of validation errors is a relatively fast procedure.
- *Monotonicity*. A new attribute is added by the algorithm to the set of input attributes only if it causes an increase in the mutual information. According to Fano's inequality [3],an increase in mutual information implies an expected decrease in the error rate.
- *Consistency*. The theoretical run time of the algorithm has been shown by us in [13] to be quadratic-logarithmic in the number of records and quadratic polynomial in the number of initial candidate input attributes.
- *Diminishing returns*. This property is very important for algorithm's practical usefulness: it means that after a small part of the running session, the results are expected to be sufficiently close to the results at the completion time. We could prove this property mathematically, if we could show that the mutual information is a concave function of the number of input attributes. Though the last proposition is not true in a general case, it is possible to conclude from Fano's inequality [3] that the mutual information is *bounded* by a function, which behaves this way. This conclusion is empirically confirmed by the results of Section 4.
- *Interruptibility*. The algorithm can be stopped at any time and provide the current list of selected attributes. Each iteration forms, what is called, a *contract anytime algorithm*, i.e. the corrections of predictive accuracy are available only after termination of an iteration.
- *Preemptability*. Since the algorithm maintains the training data, the list of selected input attributes, and the current structure of the information-theoretic network, it can be easily resumed after an interrupt. If the suspension is expected to be long, all relevant information may be stored on a hard disk.

3 Enhanced Algorithm for Anytime Induction of Oblivious Decision Trees

We aim at enhancing the Information Network algorithm by reducing the time needed to construct a classification model, while maintaining the same level of its predictive accuracy. At each iteration, the algorithm builds a new *hidden* layer by choosing an input attribute (either discrete, or continuous), which provides the maximum significant increase in mutual information relative to the previous layer. The computational complexity of evaluating a discrete attribute is the complexity of calculating its conditional mutual information $MI(A_i;T/z)$ **(1)**. The complexity of evaluating a continuous attribute consists of calculating its conditional mutual information MI (Th; T/S, z) for a given split **(2),** as well as discretizing it into a number of discrete intervals. Both these operations are performed in each hidden layer of information network for all candidates in that layer. Hence, to reduce the computational cost of the Information Network algorithm we propose the following "fast feature filtering" procedure to be applied before the network construction:

- Generate a random sample of training instances. The sample size is a pre-specified percentage of the training examples. Based on the experimental results described in Section 4, the recommended sample size can be as low as 5%.
- Compute the estimated mutual information for each candidate input attribute using the random sample of training instances. Due to the small sample size (5%), this calculation is expected to take much less time than the first iteration of the algorithm based on the entire training set. The mutual information calculated by the IN algorithm is shown in [14] to be a much more efficient feature selection method than two alternative feature selection algorithms (Relief and ABB).
- Filter out the least significant features, having the lowest values of estimated mutual information. The percentage of selected features is determined in advance. Based on the experimental results, described in Section 4, the recommended percentage is 30%, i.e., 70% of significant input attributes are removed from consideration by the algorithm. We call this approach *Fast Feature Filtering* (FFF).

The Information Network induction is performed subsequently on the subset of selected features using all training examples.

The pseudocode of the "fast feature filtering" procedure is given below:

Input: the set of n training instances; the set CI of m candidate input attributes (discrete and continuous); the target (classification) attribute T; the percentage of randomly selected training instances *sample_size*; the percentage of selected attributes from m candidate input attributes *significant_Set_size*.

Output: a set *I* of selected significant input attributes.

```
  I = ∅
  Create random sample of sample_size training instances.
  For each candidate input attribute A_i ∉ I do
      If A_i is discrete then
            Return the statistically significant
conditional mutual information cond_MI_i
between A_i and T.
      Else return the best threshold splits of A_i and the
statistically significant conditional
mutual information cond_MI_i between A_i and T.
      If cond_MI_i > 0, then
            Update the set I of selected input
attributes: I = I ∪ A_i*
  End do
  Sort the set I of selected input attributes according
to increasing its cond_MI_i
  For each i ← significantSet_size to |I|
      Exclude the less significant input attribute A_i from
the set I:
      I = I - A_i*
      i ← i + 1;
  End do
  Return a set I of selected significant input
attributes.
```

4 Experimental Results

According to [17], the performance profile (PP) of an anytime algorithm denotes the expected output quality as a function of the execution time *t*. Since there are many possible factors affecting the execution time, the performance profile, in many cases, has to be determined empirically and not analytically.

To study the performance profile of the enhanced method for induction of oblivious decision trees, we have applied it to eleven real-world datasets, including five datasets (Housing, Image Segmentation, Spambase, Waveform, Adult) from the UCI Machine Learning Repository [1], five Traffic Direction datasets provided by the Traffic Control Center of Jerusalem, and the Intrusion Detection database originally used for the Third International Knowledge Discovery and Data Mining Tools Competition (current available from the UCI KDD Archive [8]).The characteristics of each dataset are shown in Table 1. The size of the datasets varies between 506 and 10,000 cases. The total number of candidate input attributes is from 11 up to 57, including nominal and continuous features. It should be noted that the Traffic Direction, Intrusion Detection and Adult datasets have actually more than 10,000 instances, but due to the memory constraints we have confined ourselves to this amount of training examples.

We have measured the quality of the induced model by the standard 10-fold cross validation procedure. To evaluate the attribute filtering method we have experimented with three different sample sizes of 5%, 10% and 20% accordingly.

Using each sample of the training set, we have calculated the *mutual information* for all candidate input attributes and selected 20%, 30%, 40%, 50%, 60% and 70% of the most significant features. With each subset of selected significant attributes, we have built 10 *Information Networks,* using the ten-fold cross validation procedure. This experiment has been repeated eighteen times for each dataset, using six different amounts of selected attributes and three different samples of the training set. The results of each experiment, which are the averages of 10 cross-validation models, are compared to the results of the original method (not using fast feature selection). After each iteration of the algorithm, we have computed the accuracy of the current model and the time needed to induce the new hidden layer of that model. These parameters are compared with the same parameters of the original algorithm, which induces a classification model from all candidates, without filtering out less significant attributes.

Based on the results of experiments we can say that on average, only three *hidden* layers are built in all 10 models over 11 datasets. We have found also, that after the third iteration the cross-validation accuracy of most models stops to increase significantly (see the "simplicity first" approach proposed in [9]). Measuring the run time and the predictive accuracy of the enhanced algorithm over three different sample sizes (5%, 10%, 20%), we have found that the 5%-sample preserves the same performance level as the larger samples. Considering these facts we have presented in Figure 2 the performance profile of only three-layered networks induced from various sets of significant attributes selected by a 5% random sample. To simplify the comparison of the results of the novel approach with the original one, as well, for better illustration, we have normalized the execution time of each experiment with respect to the execution time of the original algorithm. For the run time equal to zero, the average accuracy over 11 datasets is computed by means of the majority rule.

Several important observations can be made from Figure 2. First, we can see, that the average performance profiles are *concave* functions of time. After the first iteration of the algorithm, the accuracy of the model is sufficiently close (85%) to the accuracy at completion time. It proves the very important anytime property of the algorithm: diminishing returns (see subsection 2.3). Second, we can observe that execution time of the enhanced approach varies between 20-50% of run time using the original method, where the lowest computational time of 20% refers to induction of the model from 30% of selected significant attributes and the highest time of 50% refers to construction of the model from 70% subset accordingly. Finally, we note that with a 20% subset of selected features, the induced model has only two layers in eight datasets out of eleven (Housing, Adult, five Traffic Direction datasets, and Intrusion Detection). Hence, we exclude the 20% subset of significant attributes from our study, and compute the average performance for a three-layered network, regarding this network as a minimal model in all 11 datasets.

The run time of the enhanced method with the 5% sample starts with 93.8 msec. for the Traffic-Direction2 datasets and goes up to 87,895 msec. for the Spambase dataset, which has 4,601 records and 57 continuous attributes. Due to space limitations, Figure 3 shows the performance profiles of five datasets only.

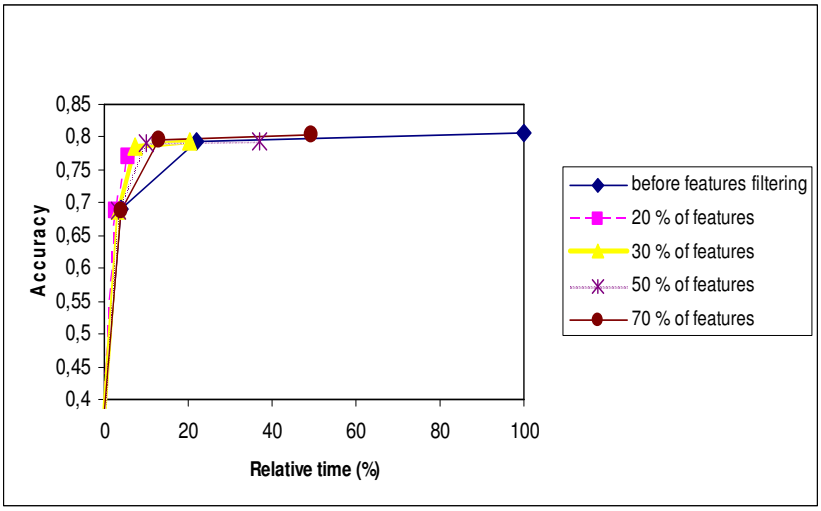

Figure2. Average performance profile of the enhanced anytime algorithm over eleven datasets, sample size 5%.

Our research is primarily aimed at reducing the computational time of the IN algorithm while keeping the same quality level of the classification model. To study how the sample size affects the accuracy and the execution time of constructing the Information Network, the average value of these parameters have been calculated for each sample size (see Table 2)

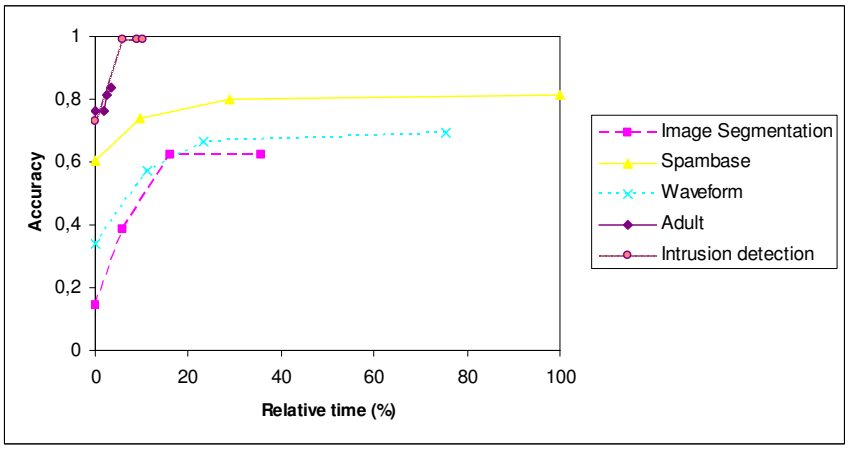

Figure3. Performance profiles of the enhanced anytime algorithm for five datasets, sample size 5%

Table 1. The characteristics of eleven benchmark datasets

Dataset	Data size	Classes	Continuous	Nominal	Total Attributes
Housing	506	3	12	1	13
Image Segment.	2,100	7	19	0	19
Spambase	4,601	2	57	0	57
Waveform	5,000	3	21	0	21
Adult	10,000	2	6	8	14
Traffic-Direction1	10,000	4	6	5	11
Traffic-Direction2	10,000	4	6	5	11
Traffic-Direction3	10,000	4	6	5	11
Traffic-Direction4	10,000	4	6	5	11
Traffic-Direction5	10,000	4	6	5	11
Intrusion Detect.	10,000	4	14	2	16

Table2. Average accuracy, execution time and standard deviation of three-layered model over eleven datasets and various percentages of selected significant attributes

Sample Size (%)	Average attributes filtering time (sec.)	Average accuracy	Average execution time (sec)	STDEV of mean accur.	STDEV of mean time	Slope ($*10^{-4}$)
5	1,8	0.79	31,7	0.013	6	2.5
10	1,8	0.80	32,1	0.013	6	2.49
20	2	0.79	32,2	0.014	6	2.45

As one can see from Table 2, the sample size affects the induction time of the classifier and does not affect its accuracy. To evaluate the trade-off between these characteristics we calculate their ratio called the *Slope* using the following equation:

$$\text{SLOPE} = \frac{\Delta Q(t)}{\Delta t} \qquad \textbf{(3)}$$

Where,

$\Delta Q(t)$ = the difference between the accuracy of the complete (three-layered) model and the initial (majority rule) accuracy;

Δt = the execution time of inducing a complete (three-layered) model

According to the value of *Slope* we can suggest that the 5% sample size is slightly more preferable than the 10% and 20% sample sizes.

Another question is which percentage of selected significant attributes is preferable for optimizing the accuracy-time relationship. To answer this question, we

are summarizing in Table 3, the average accuracy and execution time, for each subset of significant attributes, comparing these parameters to the results of the original method, without the fast feature filtering (FFF), where the average accuracy is 0.806 and execution time is 88,115 msec.

The decrease in accuracy and execution time (see Table 3, columns 2 and 3) is computed relative to the 100 % set of candidate attributes. As we can see, the maximal reduction of time (79.9%) is reached with the 30% set. It is important to note that, the decrease in accuracy vs. the original method (see Table 3, column 5) has not been found statistically significant as for various sample sizes, as for various percentages of selected attributes. To find the optimal percentage of significant features we have calculated the *Slope* for each subset of selected attributes. According to the *Slope* value we can say that the 30% percentage of significant features is optimal for accuracy-time optimization task.

Finally, we can conclude, based on analysis of the experimental results obtained for eleven datasets that best trade-off between the accuracy of the three-layered Information Network and computational time needed for its construction is achieved on a 30% subset of significant attributes selected by a 5% random sample. In this case, the execution time is reduced by almost 80%.

Table 3. Average accuracy, execution time and standard deviation of three-layered model, over eleven datasets and various sample sizes

Percent signif. attrib.	Aver. accur. after FFF	Aver. time (sec.) after FFF	Slope ($*10^{-4}$)	Decrease accur. after FFF (%)	Decrease time, after FFF (%)	STDEV of mean accuracy after FFF	STDEV of mean time, after FFF
30	0.788	17,6	4.38	4	80	0.018	5
40	0.792	25,7	3.08	2	71	0.018	6
50	0.793	33,2	2.39	2	62	0.018	8
60	0.801	39,3	2.04	1	55	0.015	9
70	0.804	44,0	1.83	0.3	50	0.015	11

One of the important benefits of the proposed FFF approach is that it allows capturing the tradeoff between the solution quality and the time saved and/or complexity of classification represented by the number of the most significant input attributes. The anytime interruptability of the algorithm allows stopping it after each iteration to provide an approximate solution that is close to the complete result. This can be crucial for real-time classification algorithms working with a large number of input attributes and/or with timing constraints.

5 Conclusions

In this paper, we have proposed a new, "Fast Feature Filtering" version of an anytime algorithm for constructing a classification model called Information Network (IN). We have studied and improved the important anytime property *diminishing returns* of the algorithm. The new method enables to reduce significantly its computation cost while preserving the same level of model quality. This goal is achieved by means of monitoring the relationship between the random sample size of training examples and the percentage of most significant input attributes selected by this sample. The proposed algorithm is evaluated on eleven benchmark datasets available from different sources. The quality of the induced model is measured by its classification accuracy using the standard 10-fold cross validation. The performance profiles of the new version have been shown to be concave functions of time. Based on the experimental results, the optimal tradeoff between accuracy of a three-layered Information network and execution time needed for its construction is achieved with a 30% subset of significant attributes selected using a 5% random sample. In this case, the accuracy rate is very close to the accuracy of the original algorithm, whereas the execution time is reduced by almost 80%. Topics for future research include predicting the expected quality for a given execution time (and vice versa), and integrating the enhanced version of the algorithm with real-time learning systems such as IOLIN [2].

References

1. Blake, C.L., Merz, C.J. UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.
2. Cohen, L., Avrahami, G., Last, M, Kandel, A., Kipersztok, O. "Real-Time Data Mining of Non-Stationary Data Streams from Sensor Networks", Information Fusion Journal, Special Issue on Information Fusion in Distributed Sensor Networks, in press. doi:10.1016/j.inffus.2005.05.005.
3. Cover, T.M., Thomas J.A., Elements of Information Theory, Second edition, Wiley, 2006
4. Dean, T.L., 1987. Intractability and Time-Dependent Planning. In *Reasoning About Actions and Plans*, Georgeff M.P., Lansky A. L., Eds. Morgan Kaufmann Publishers, San Francisco, California, 1986, pp. 245-266.
5. Dean, T.L, Boddy, M., An Analysis of Time-Dependent Planning, In *Proceedings of the American Association for Artificial Intelligence Conference (AAAI-88)* (Cambridge, Massachusetts, 1988), AAAI, MIT Press, pp. 49-54.
6. Esmeir, S., Markovitch, S., Interruptible Anytime Algorithm for Iterative Improvement of Decision Trees, In Proceedings of The 1st Workshop on Utility-Based Data Mining (UBDM-2005), held with The 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2005), pp 78-85
7. Gaber, M.M., Yu, P.S., A Framework for Resource-aware Knowledge Discovery in Data Streams: A Holistic Approach with Its Application to Clustering, in Proceedings of ACM, Symposium on Applied Computing, (SAC 2006), ACM Press, pp 649-656.
8. Hettich, S., Bay, S.D. (1999). The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science.
9. Holte, R.C., Very simple classification rules perform well on most commonly used datasets. Machine Learning, 11(1), pp 63-91, Apr. 1993.

10. Horvitz, E.J., Reasoning about Beliefs and Actions under Computational Resource Constraints, Proc. of the 1987 Workshop on Uncertainty in AI, Seattle, Washington, 1987, pp 429-444.
11. Horvitz, E.J., Suermondt, H.J., Cooper G.F., Bounded Conditioning: Flexible Inference for Decision under Scarce Resources. Proc. of the 1989 Workshop on Uncertainty in Artificial Intelligence, 182–193. New York: North-Holland, 1989.
12. Last, M., Kandel, A., Maimon, O., Eberbach, E., Anytime Algorithm for Feature Selection, Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing (RSCTC'2000), pp. 532-539, Springer-Verlag, 2001.
13. Last, M., Maimon, O., A Compact and Accurate Model for Classification, IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 2, pp. 203-215, February 2004.
14. Last, M., Kandel, A., Maimon, O., Information-theoretic algorithm for feature selection, Pattern Recognition Letters 22(2001) pp. 799-811.
15. Liu, J.W.S., Lin, K.J., Shih, W.K., Yu, A.C., Chung, J.Y., Zhao, W., Algorithms for Scheduling Imprecise Computations, Computer, vol.24 no.5, pp.58-68, May 1991.
16. Phung, N.D., Gaber, M. M., Röhm, U., Resource-aware Online Data Mining in Wireless Sensor Networks, Proceedings of the 2007 IEEE Symposium on *Symposium on Computational Intelligence and Data Mining (CIDM 2007)*, pp 139-146.
17. Zilberstein, S., Using Anytime Algorithms in Intelligent Systems, AI Magazine, vol. 17, no. 3, pp. 73-83, 1996