

# Implementing SPRINT in Weka

## IT608 Course Project

Kaushal Mittal  
(04329024)

Abhishek Seth  
(04329001)

Amar Agrawal  
(04329017)

April 20, 2005

## 1 Objective

- Existing decision tree implementation of weka fails for large datasets.
- To add a scalable implementation of decision trees in Weka.
- The main memory requirement is to be made efficient.
- Support for disk-resident data.

## 2 Design

The current implementation of Weka loads all the instance data into memory and makes multiple copies of the data over its execution. Inorder to make the decision tree implementation scalable we added support for disk resident data so that the instance data is never loaded fully into memory. Following classes are added in weka to support scalable decision tree.

- SInstances: This class is an extension of existing 'Instances' class. The data file is not loaded fully into memory at any time. Cache and disk files used to support incremental reads and writes.
- SprintClassifier: Design similar to Weka J48 classifier. Implements the SPRINT algorithm [SAM96] and uses Gini index as the split criteria.

- AttributeRecord : 3 tuple consisting of record-id , attribute value and classvalue.
- AttributeList: List of AttributeRecords stored for each attribute in the data file. Maintained in disk files and used incrementally for scalability. External merge sort used to sort attribute lists.
- OptimisedEvaluation : Need for a separate Evaluation class for SPRINT due to changes made.

There are other supporting classes in the SPRINT package that are not listed here.

### 3 Results Obtained

We tested the new Scalable Weka(SWeka) over various datasets. A summary of these tests is provided in table 1.

Dataset	Instances	Accuracy		Time(secs)	
		Weka	Sweka	Weka	Sweka
Glass	214	100%	91.67%	0.02	0.31
Adult	32561	89.42%	91.34%	52.83	292.55
Homework2	34871	66.91%	67.52%	333.19	1905.56

Table 1: Comparison for Weka and SWeka

Some important conclusions can be derived :

1. Memory profile for execution less than original Weka and remains constant over the execution of classifier.
2. Accuracy of Gini index comparable to Gain Ratio used by J48 for test cases used.
3. SWeka takes more time(x6) to build the model. Reason is the additional file I/O due to disk resident data.

### 3.1 Comparison between Gini index and Gain Ratio

The accuracy of our SPRINT implementation(GINI index) was greater than J48(Gain Ratio) over almost all the datasets used.

## 4 Current Status

Implementation for the SPRINT classifier completed. The classifier builds a binary classifier and results evaluated on supplied test file. Following extensions need to be made:

1. Support for Crossvalidation : Need to add fold generation logic to SInstance class
2. Integration with Weka gui : Modifications required to Weka to support new OptimisedEvaluation and SInstance class
3. Add support for other splitting criteria eg. gain ratio.

## 5 Members Contribution

Each project member has made an equal contribution to this project. We spent the initial time studying the Weka architecture. The design was discussed in the subsequent project meetings and work was divided roughly as below:

- Abhishek : SInstances, AttributeList.
- Amar: GiniSplitIndex, Evaluation.
- Kaushal: SprintClassifier, Pruneable Decision Tree.

We have spent around 80 man hours in this project.

## References

- [SAM96] John C. Shafer, Rakesh Agrawal, and Manish Mehta. Sprint: A scalable parallel classifier for data mining. In *VLDB*, pages 544–555, 1996.