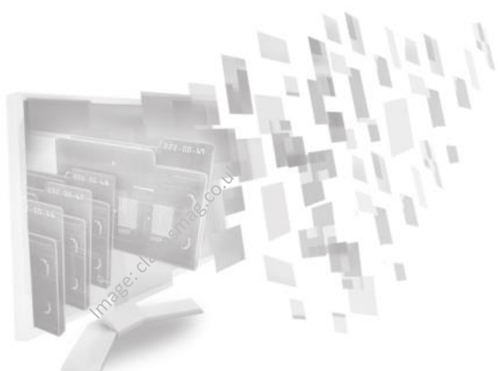
Ensemble Learning in Data Streams

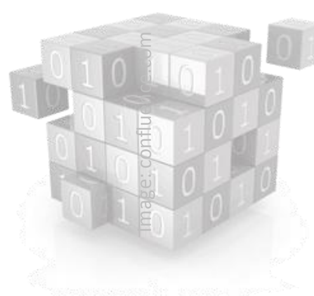
Master's thesis

Hossain **Mahmud**

Supervisor: Prof. Dr. Burkhard Rost

Advisors: Dr. Eirini Ntoutsi, Dr. Lothar Richter



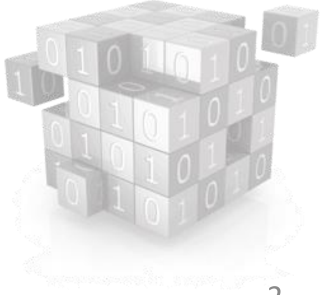


LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Data Streams

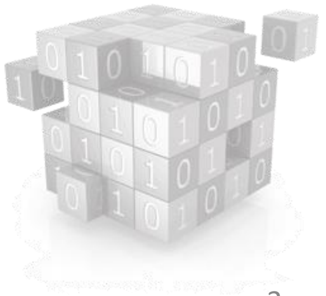
Stream data arrives continuously and rapidly, and if it is not processed immediately, then it is lost forever. Moreover, the arrival speed of data is so high that it is not feasible to store it all in active storage (i.e., in a conventional database), and then process it later.

Social networks, telecommunications, WWW, scientific experiments, e-commerce systems, etc.



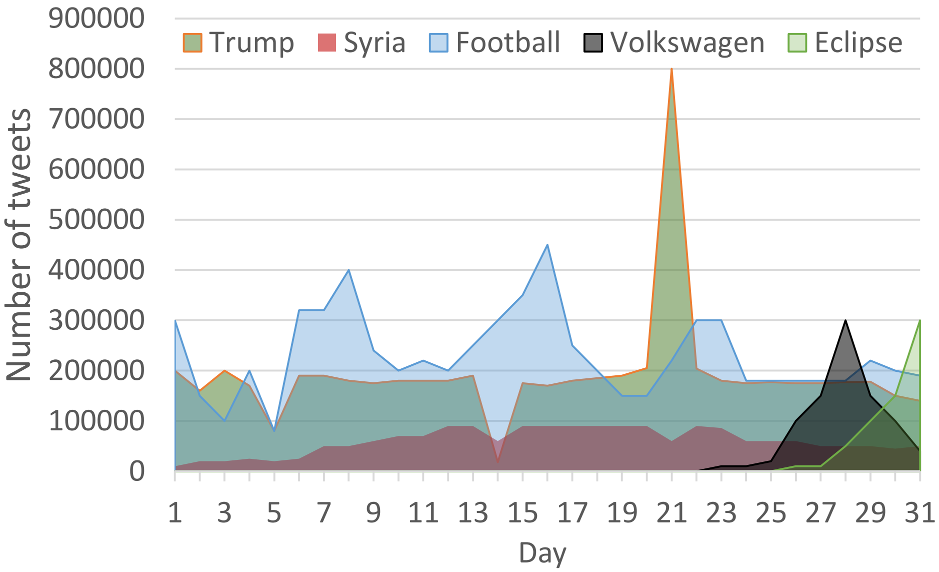
Introduction

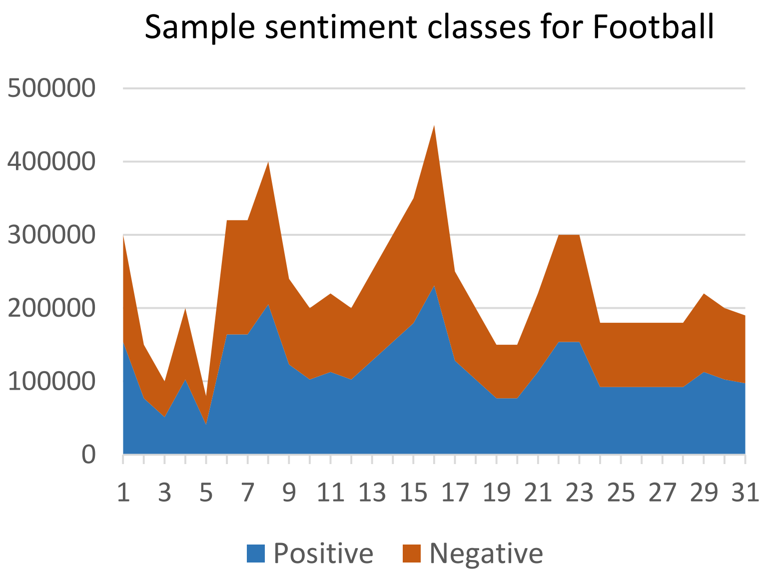
- Stream mining is different from batch mining
- Significant differences include –
  - The underlying data distribution may evolve over time
  - Data cannot be considered independent or identically distributed over time
  - Data is time and space dependent
- Model should be ready to predict anytime
- Model should access data only once (or a small number of times)



Motivation

- Number of tweets in Twitter for 5 different topics (Aug 28, 2015-Sept 28, 2015)
- Topics contribute at different rates in the final stream
- The target class could be balanced though



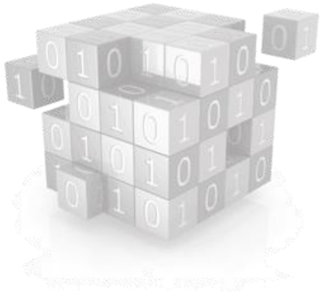


Source: topsy.com

Problem Statement

Online classification of data streams, which are decomposable into varying speed sub-streams, using decision tree ensemble.

We don’t know the original substreams, our stream consists of their fusion (e.g., Twitter stream)



Existing Decision Tree Based Approaches and their Ensemble

Assumption

- To find the best attribute for a split in a node, it would be sufficient to consider a certain fraction of the stream [1]
- Hoeffding bound provides a statistical guarantee [2]. Error bound to decide with *(1- δ)* certainty for *n* random variable with *R* being the range of the variables

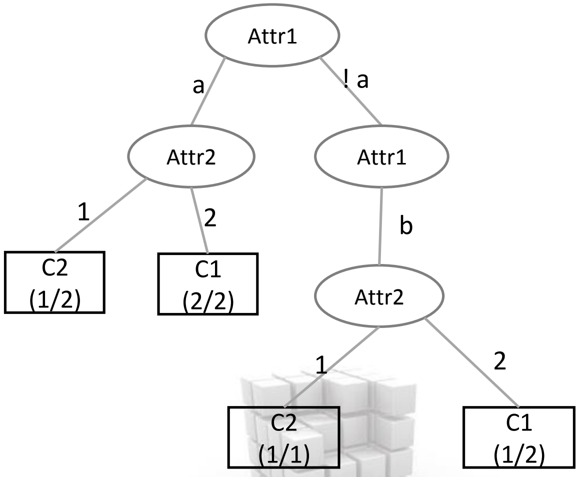
$$\epsilon \leq \sqrt{\frac{R^2 \ln(2/\delta)}{2n}}$$

- A decision taken after observing a certain amount of instances would remain the same after seeing an infinite number of instances with (1- δ) certainty

Hoeffding Tree (HT) [3]

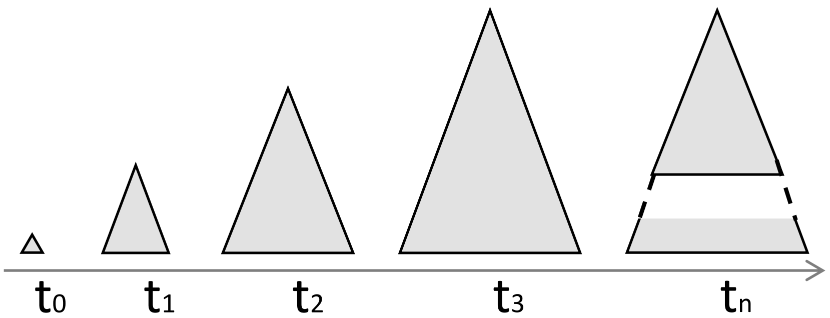
- Decide based on the data in hand
- Try to split if a leaf is impure
- Use information gain/ Gini index to obtain the best two split attributes
- Split if the best one performs better than the second best by at least a margin of the Hoeffding bound
- Keeps updating the tree with incoming data
- The tree only grows

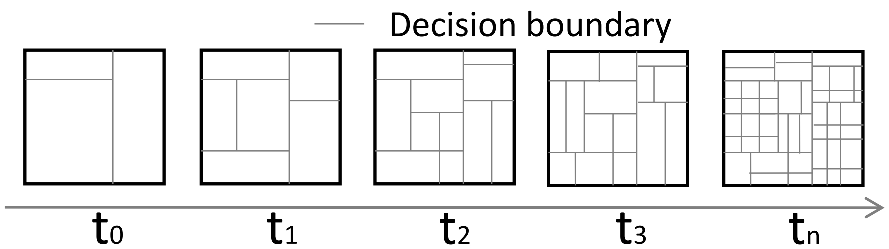
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1



Hoeffding Tree cont.

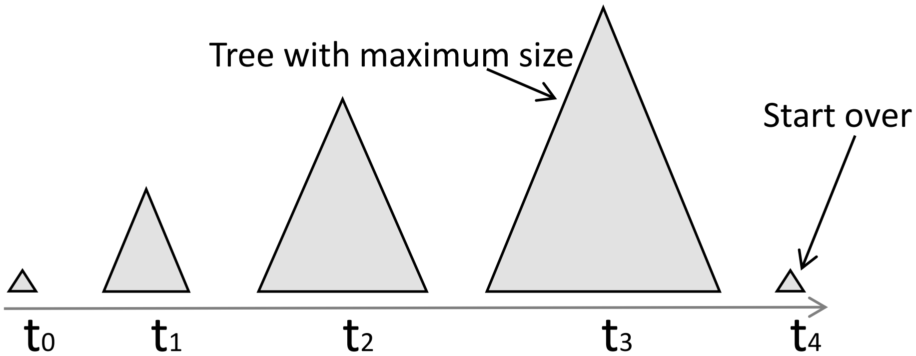
- No rule gets deleted
- With time
  - The tree becomes to complex → overfit
    - The tree produces redundant rules e.g., IF x < 7 AND x < 5 AND x <3 THEN ...
  - The historical data dominates the decisions→difficult to adapt

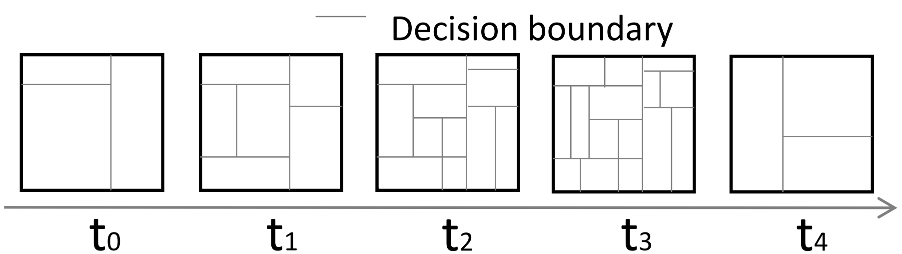




Adaptive Size HT (ASHT) [5]

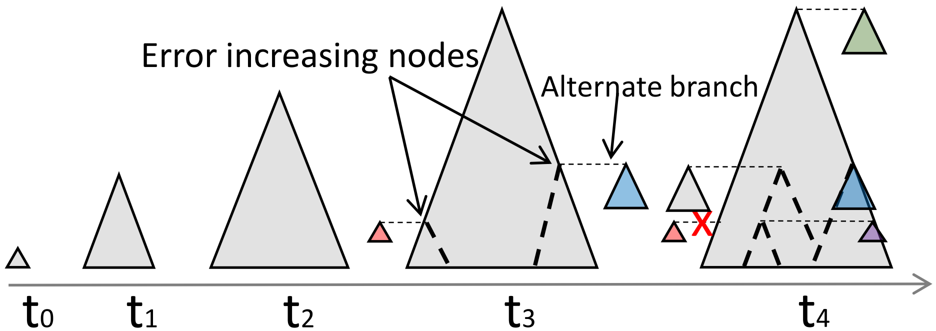
- Set bound on the maximum tree size
- Start over when limit is reached
- The tree forgets historical information but
  - Loses all information learned thus far

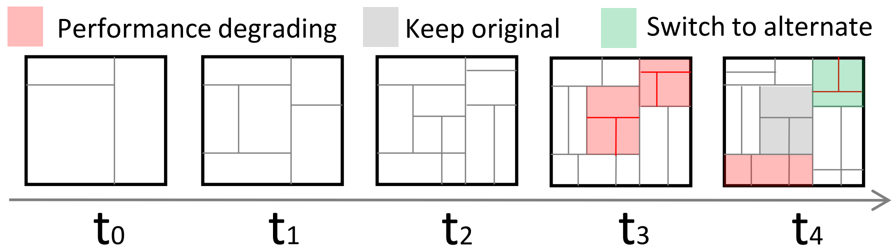




Adaptive HT (AdaHT) [4]

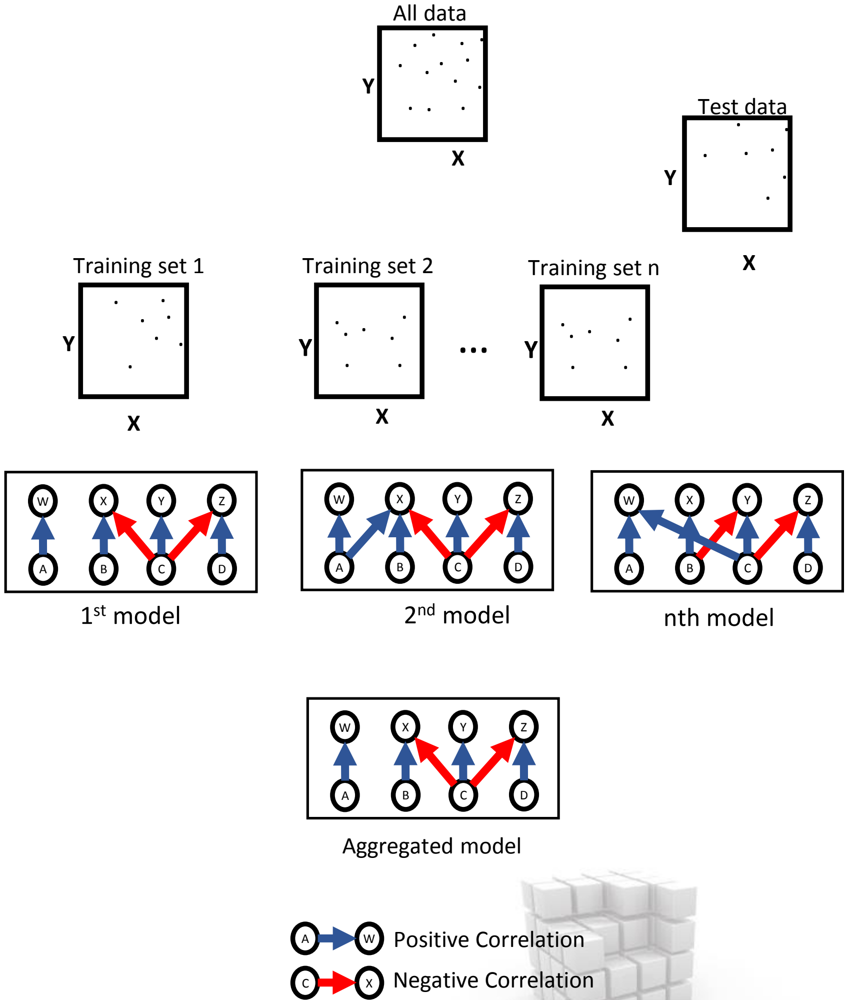
- Start maintaining an alternate sub-tree when a node starts performing worse than previous
- When the new sub-tree starts performing better, replace the original
- If original sub-tree keeps performing better, delete the alternate sub-tree





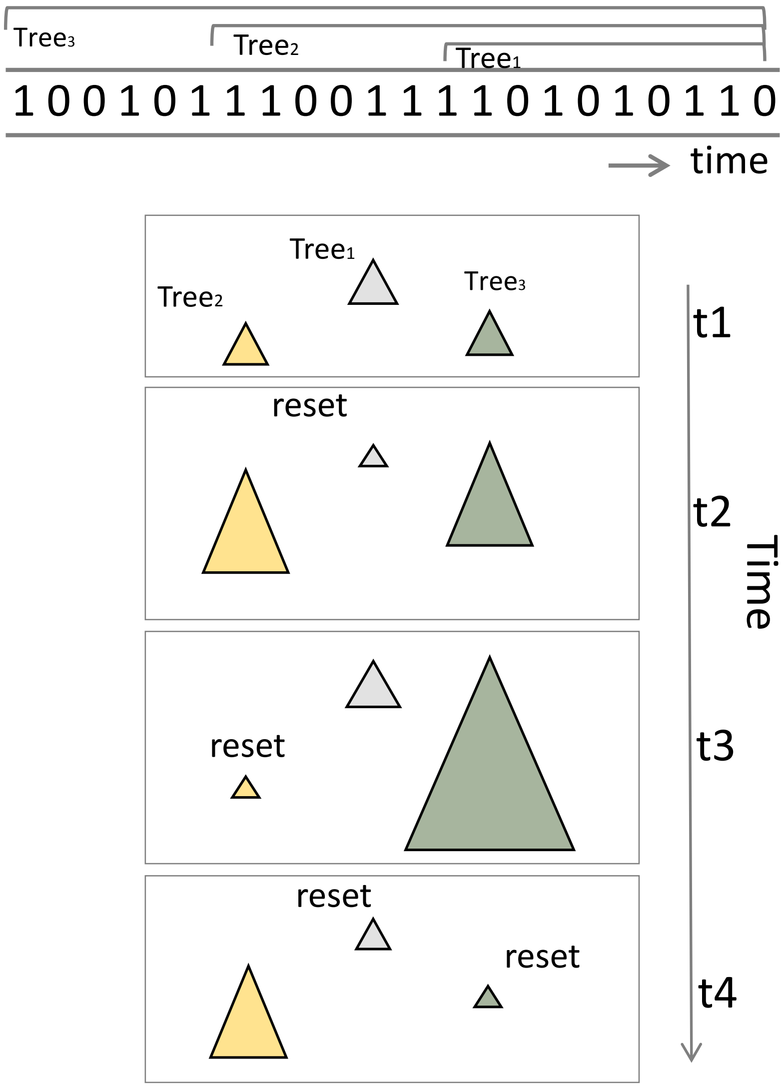
Bagging Ensemble [6,7]

- Generate n training sets by random sampling of the original training set
- Learn a model for each training set
- Use majority voting for classifying test data



Bagging with ASHT (BagASHT) [5]

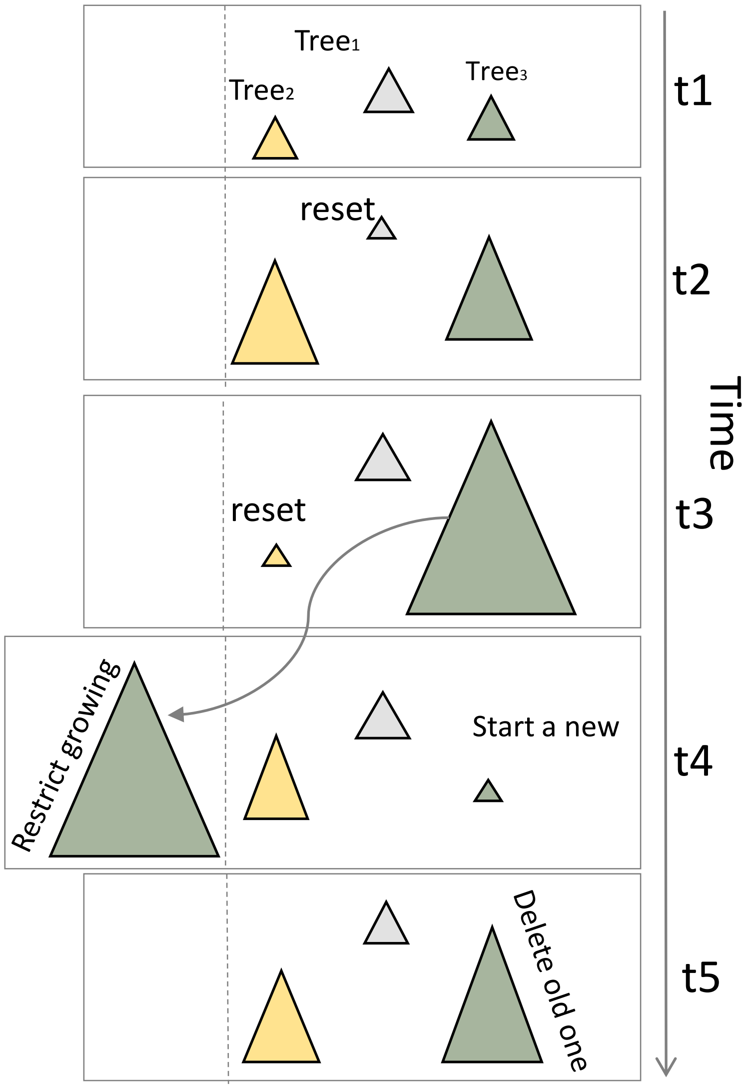
- Bagging with Adaptive Sized Hoeffding Tree (BagASHT) [5]
- Maintain an ensemble of HTs of different size limit
- Allows building models for different time-frames
  - Smaller trees react faster to change, larger trees slower
- Every time a larger tree resets, the ensemble loses significant information about “longer living” concepts



A New Approach

Carry-over Bagging (BagSRHT)

- Carry-over Bagging with Size Restricted Hoeffding Tree (BagSRHT)
- Once the limit is reached
  - Move the tree to the extra list
  - Restrict it from growing
  - Start maintaining another tree
  - Take votes from both
- Delete oldest from the extra list
  - When the new one is large enough
  - Or more trees reached their limit
- Use alternate sub-tree method (AdaHT) instead of HT

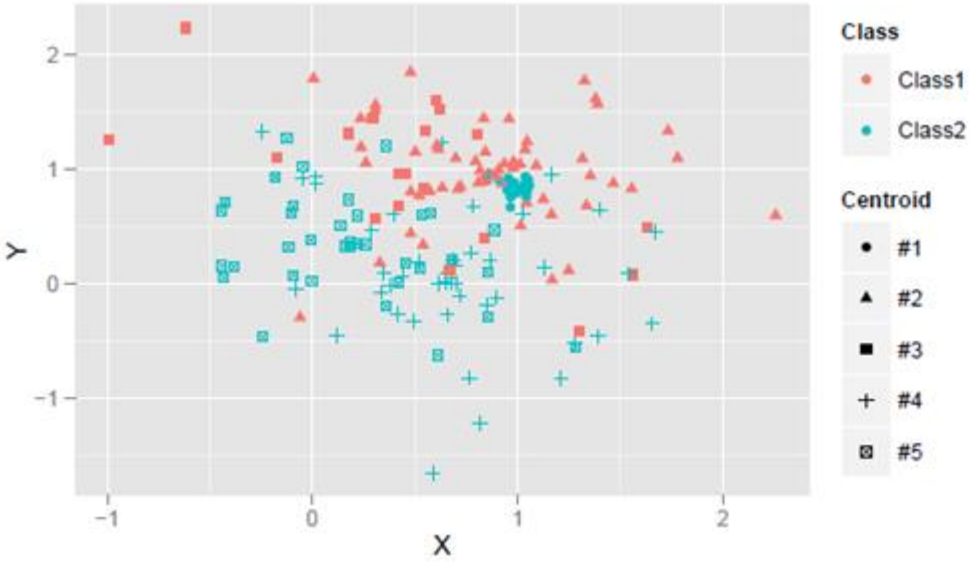


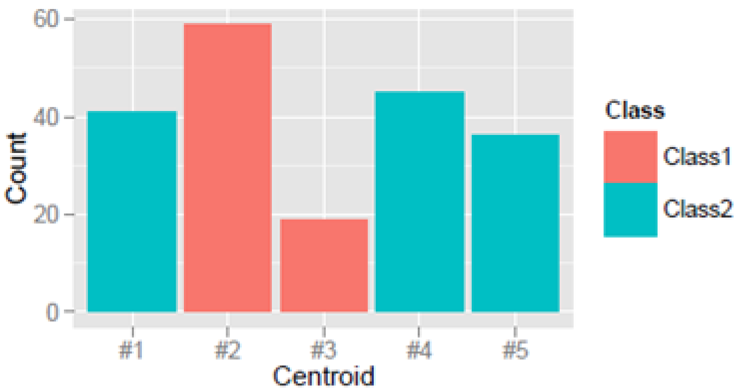
Data Sets

- Most existing generators use a randomized approach
- Random Radial Basis Function (RandRBF) generator
  - Not possible to generate varying speed data set
- Modified the generation scheme to achieve desired properties in the data set
  - Varying Speed RBF (VSRBF) generator
  - Slow but consistent sub-streams
  - Fast and short lived sub-streams, etc.

Random RBF Data Set

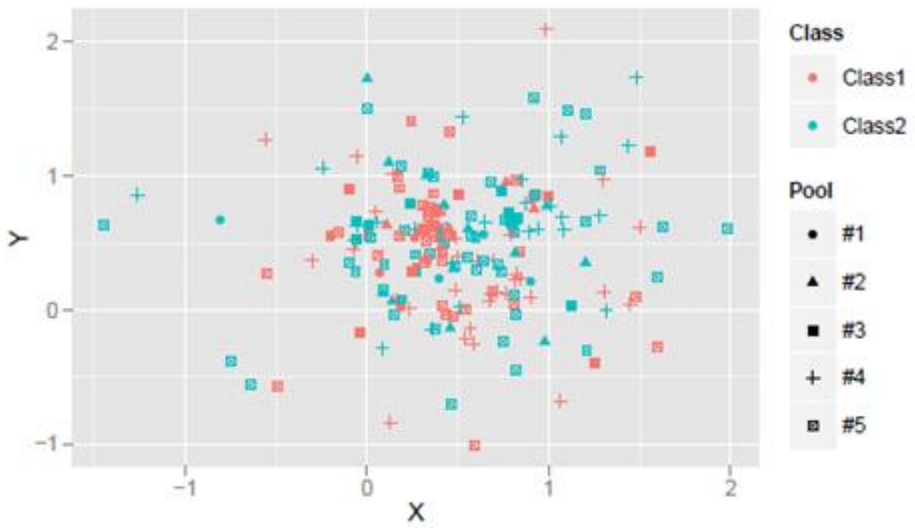
- Centroids contribute to the final stream depending on their weights
- All centroids are active all the time

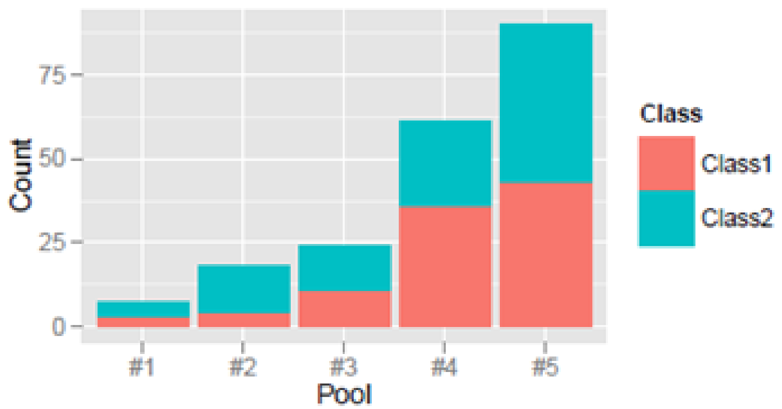




Varying Speed RBF Generator

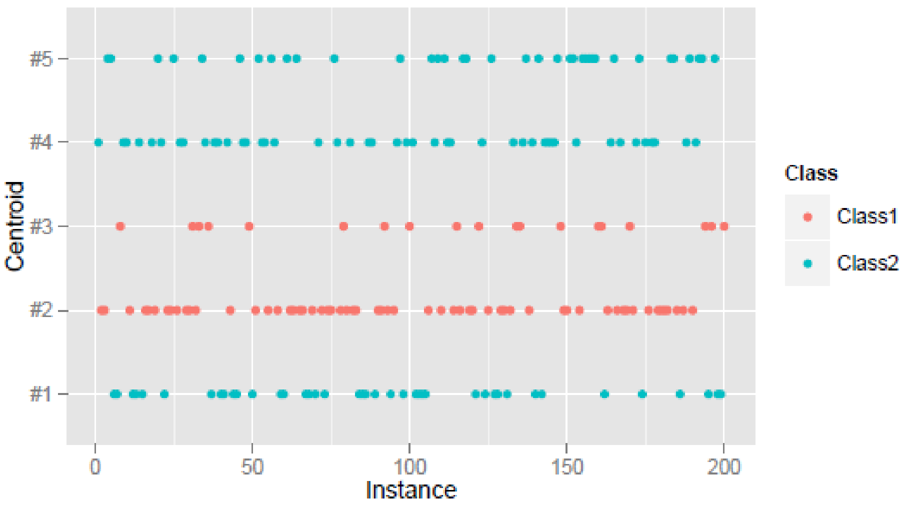
- Replace the concept of centroids with the concept of pools
- Each pool contains a number of centroid and has different activation and contribution rate
  - Slow pools, fast pools, average speed pools, etc.



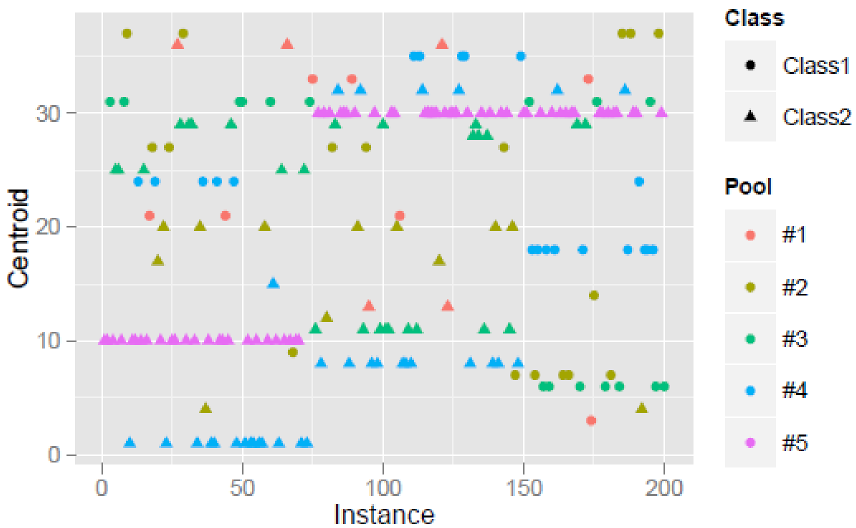


Comparison

- All the concepts (centroids) are active all the time for the random RBF generator
- The activation period changes in VSRBF generation scheme



Random RBF Generator



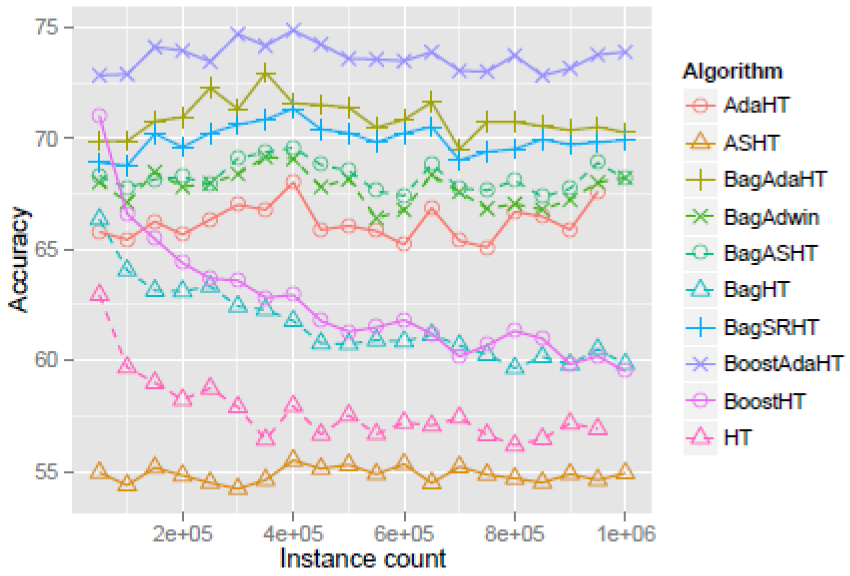
VSRBF Generator

Evaluation

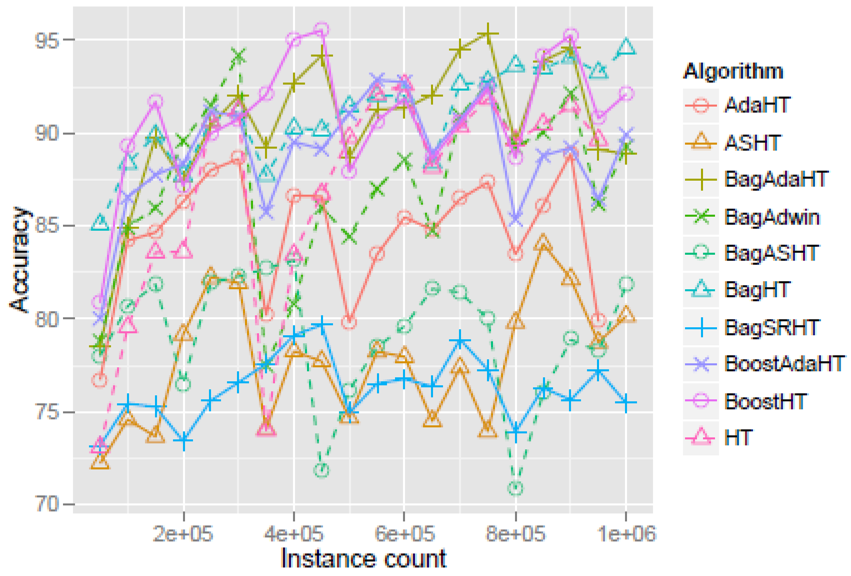
- Data generation
  - Random RBF generator
  - Varied Speed RBF generator
- Prequential evaluation
  - Use every instance to first test and then train the model
- Binary class problem
  - 10 attributes
  - 1 million instances

BagSRHT vs BagASHT

- Performance is prone to tree reset
  - VSRBF performance suffers significantly more than Random RBF
- BagSRHT is likely to be more stable than BagASHT



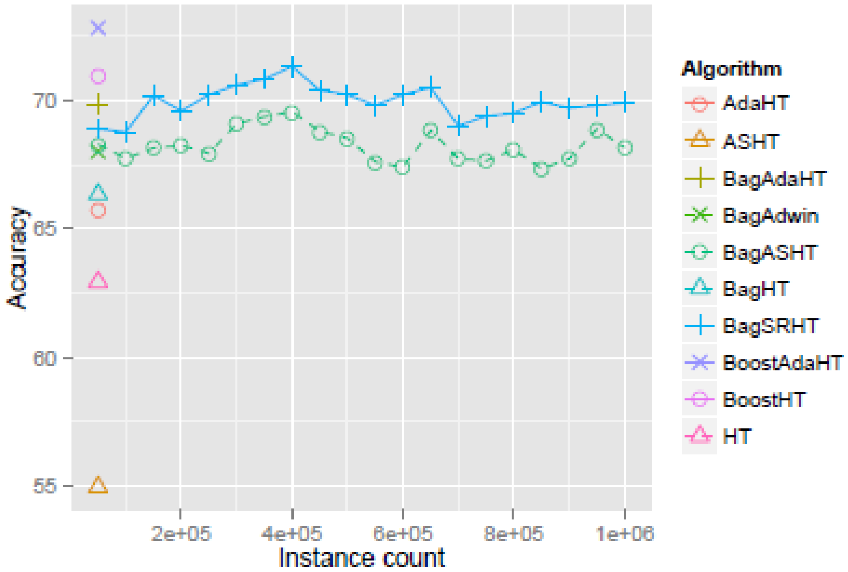
Random RBF



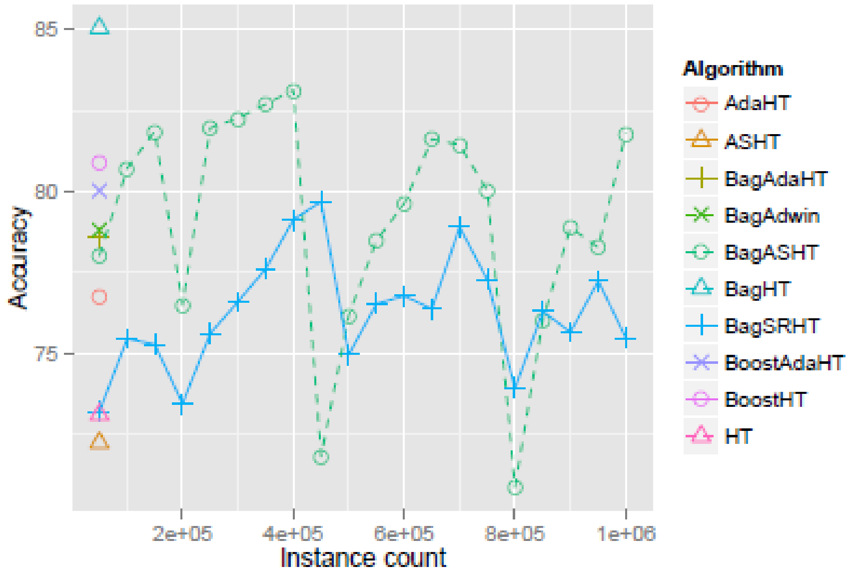
VSRBF

BagSRHT vs BagASHT

- Performance is prone to tree reset
  - VSRBF performance suffers significantly more than Random RBF
- BagSRHT is likely to be more stable than BagASHT



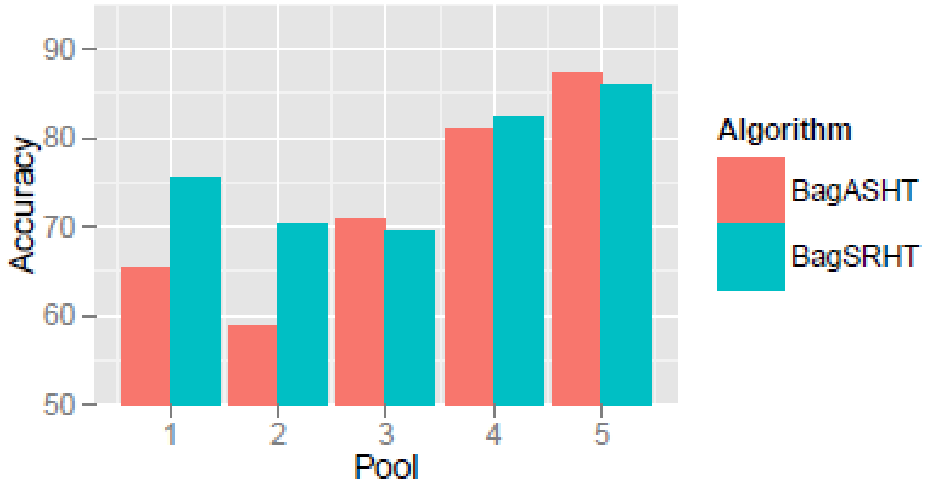
Random RBF



VSRBF

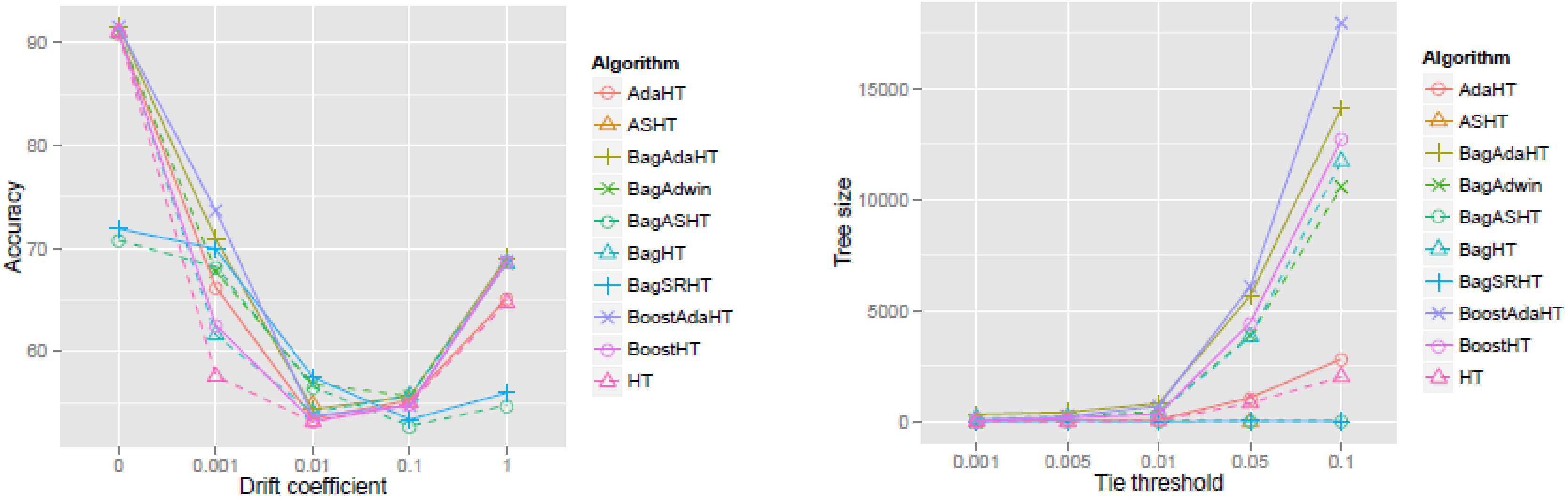
BagSRHT vs BagASHT

- Bag SRHT performs better for slower sub-streams
- However, increases misclassification for faster sub-streams



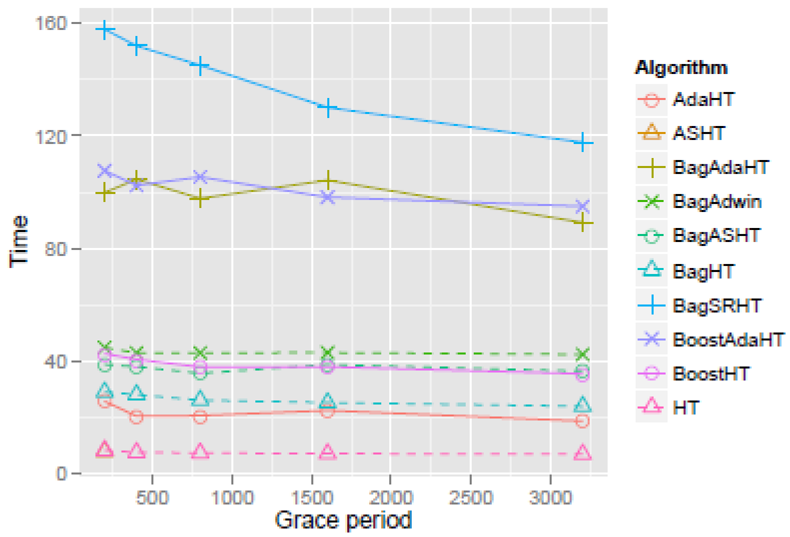
Effect of Parameters

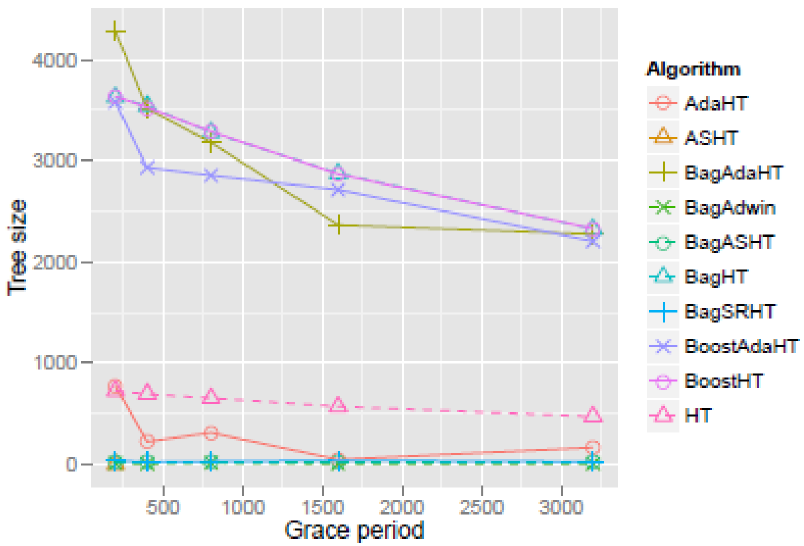
- Without any drift, all methods perform the same
- With small drift ADWIN and boost variants perform best
- A tie threshold may be used to break ties between two equally good attributes
  - Larger tie threshold causes massive trees



Effect of Parameters

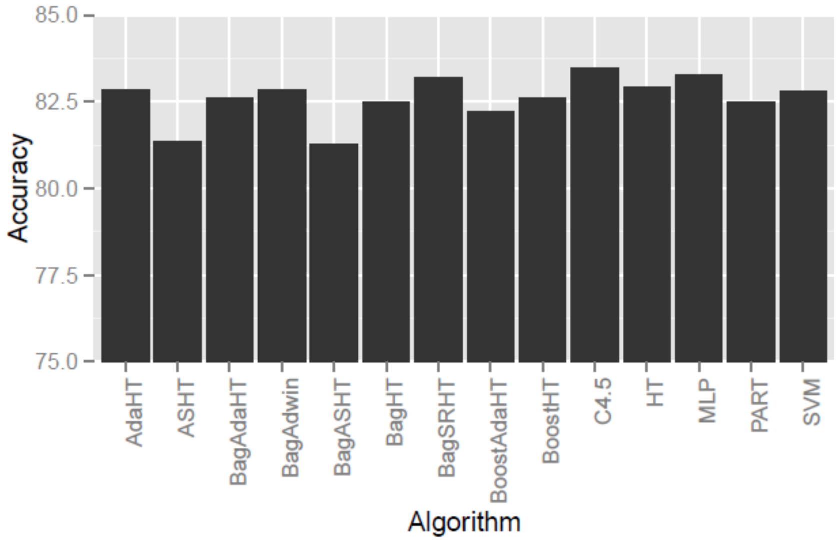
- Grace period is used to reduce unnecessary computations
  - Effectively reduces processing time and tree size

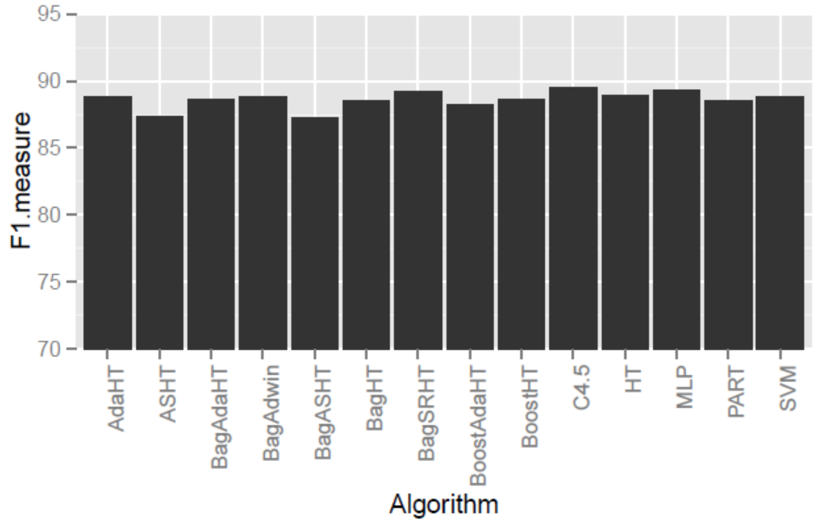




Comparison with Batch Approaches

- Census income data set [8]
- 10 fold cross validation for batch approaches
- ADWIN variants and BagSRHT reach closest to C4.5





Conclusion

- New look at the composition of streams
- New approach to generate varied-speed data streams
- Improvement on ASHT bagging by introducing carry-over bagging
- A comprehensive survey of the existing literature

Future Work

- Fitting the model into text streams
  - Our motivation for this work is taken from text streams
  - Our evaluation currently is performed with numeric data
- Devising an approach to select a subset of the incoming stream to learn the model
  - Equivalent number of instances from both slower and faster streams

References

1. Catlett, J. (1991). Megainduction: Machine learning on very large databases. In *PhD thesis, University of Sydney*.
2. Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. 58:13–30.
3. Domingos, P. and Hulten, G. (2000). Mining high-speed data streams. In *Proceedings of the ACM KDD*.
4. Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time changing data stream. In *ACM KDD*, pages 97–106.
5. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., and Gavaldà, R. (2009). New ensemble methods for evolving data streams. In *SIGKDD*, pages 139–148.
6. Oza, N. C. and Russell, S. (2001). Online bagging and boosting. In *Artificial Intelligence and Statistics*, pages 105–112.
7. Breiman, L. (1994). Bagging prediction.
8. Kohavi, R. (1996). Scaling up the accuracy of naïve-bayes classifiers: a decision tree hybrid. In *Knowledge Discovery and Data Mining*.


THANK
YOU!

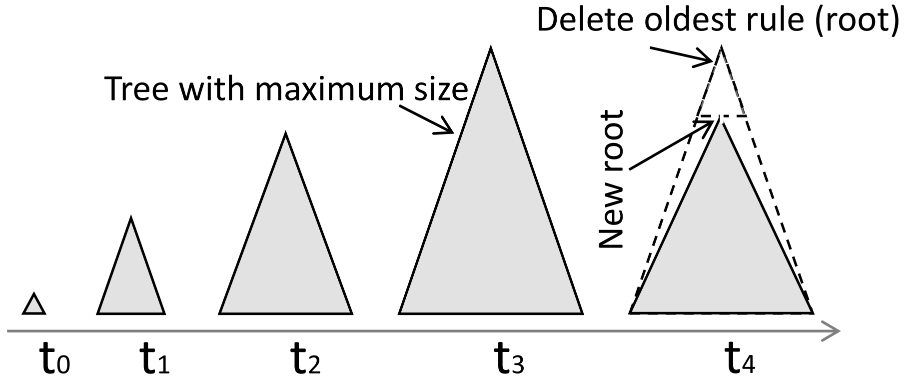
Algorithm Summary

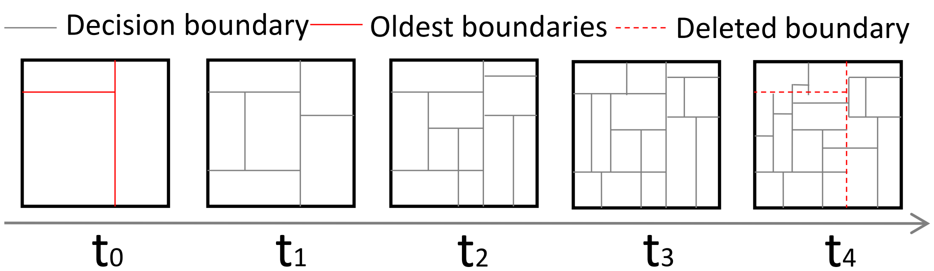
- Singleton models
  - HT – Hoeffding Tree – Grow indefinitely
  - AdaHT – Adaptive HT – Maintain alternate sub-trees
  - ASHT – Adaptive Size HT – Reset if limit reached
  - SRHT – Size Restricted HT (new method) – defer reset

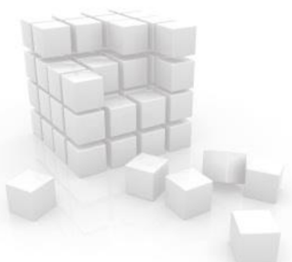
- Ensemble models
  - Bag* – Bagging using HT/AdaHT/ASHT/SRHT
  - Boost* – Boosting using HT/AdaHT

Adaptive Size HT (ASHT) cont.

- Delete oldest rule i.e. root when limit is reached
- Delete all the children of the root except for the one that will be new root
- Retains most of the learned information
- Rearranges the decision boundaries







Discussions Generalization Error Bound

- Generalization error indicates how well a learning method generalizes to unseen data
  - Distance between the error on training and testing set
  - Averaged over all possible training sets
- In stream mining, prequential evaluation uses an instance first to test and then to train the model
  - Used only once, no iteration
- Notion of generalization error thus doesn't apply
- Rather, confidence of the model is the main focus
- Allowable error for each split decision was set to 0.0000001

Background Change Detection

- ADaptive WINdow (ADWIN) change detection method
- Ensures the property there has been no change in the average value inside the window for the maximally statistically consistent length

1 0 0 1 0 1 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0 0

→ time

- Threshold value is calculated based on the sizes of the windows

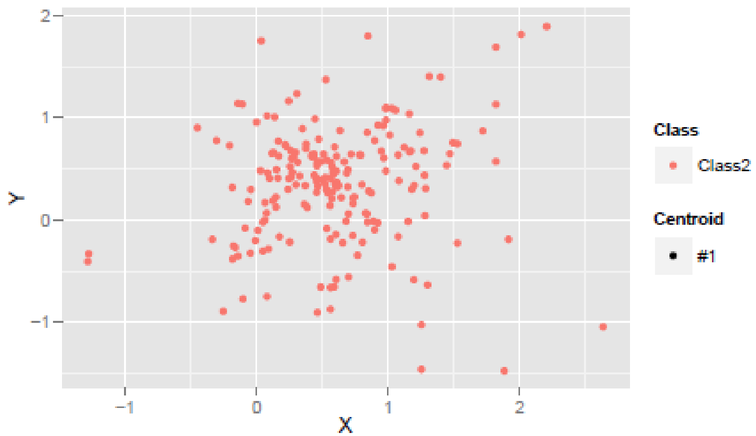
$$m = \frac{2}{1/|W_0| + 1/|W_1|}$$

$$\epsilon_{cut} = \sqrt{\frac{1}{2m} \ln \frac{4|W|}{\delta}}$$

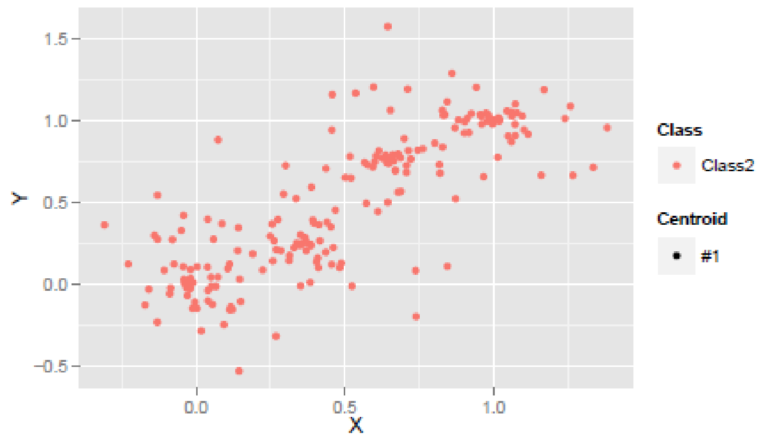
Random RBF Generation

- Random radial basis function (RandRBF) generator is challenging for DT based approaches
  - Randomly chooses user-defined number of centroids in the hyperspace
  - Assigns class label, drift coefficient, standard deviation, weights to each of the centroids
  - Instances are generated by selecting a centroid at random (weighted), and choosing a point using normal distribution

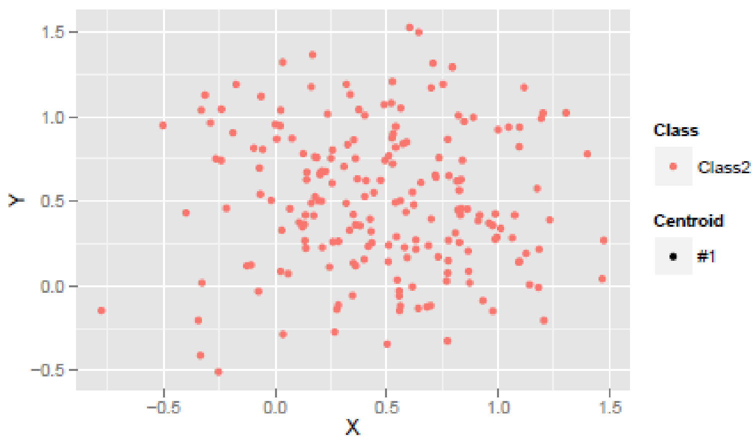
Random RBF Data Set



Drift Coefficient 0.0



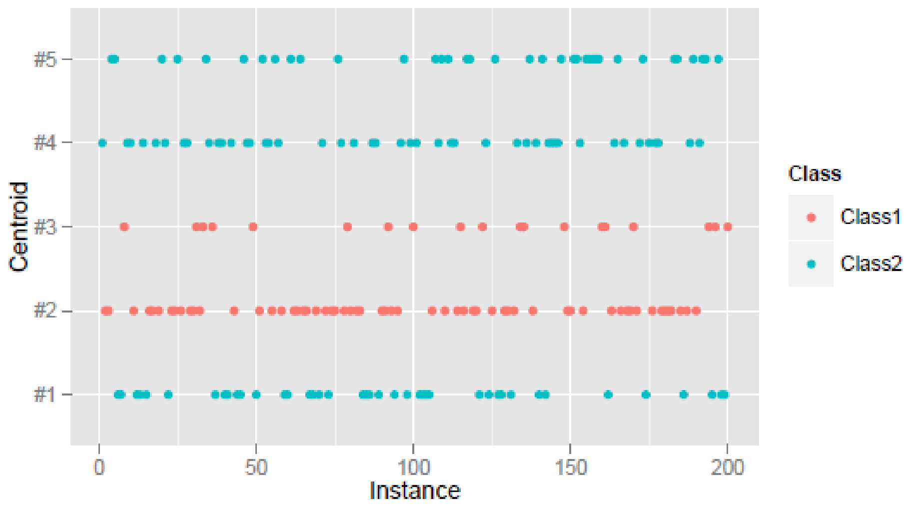
Drift Coefficient 0.01



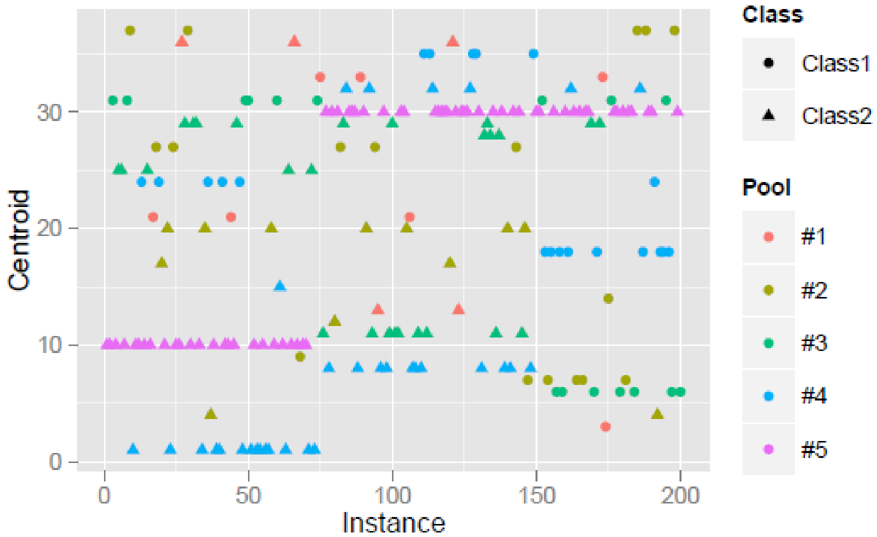
Drift Coefficient 0.1

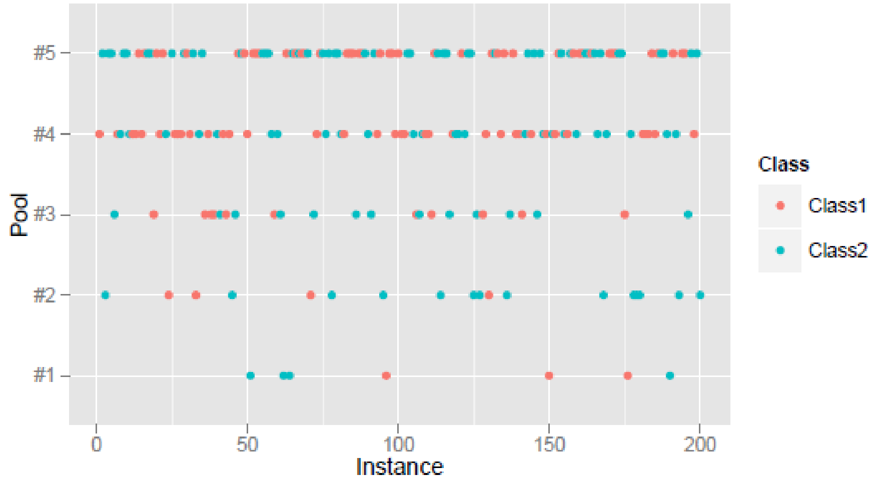
Drift Coefficient 1.0

Data Set Comparison



Random RBF Generator

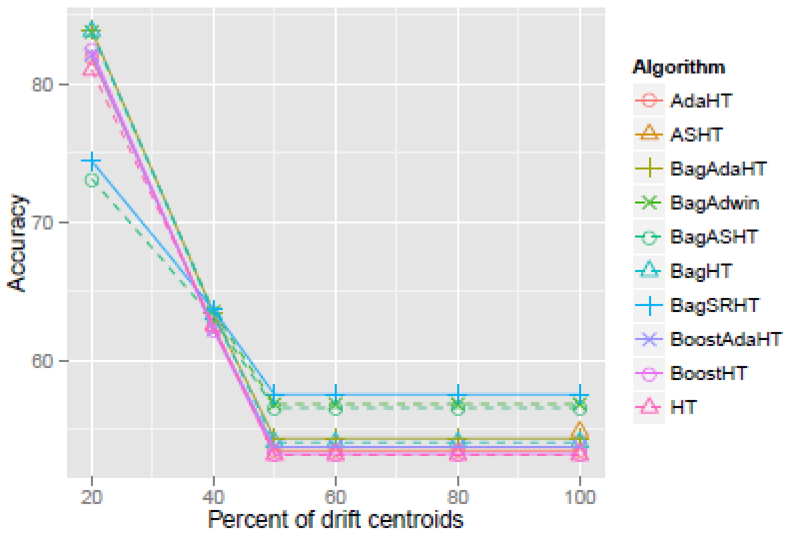


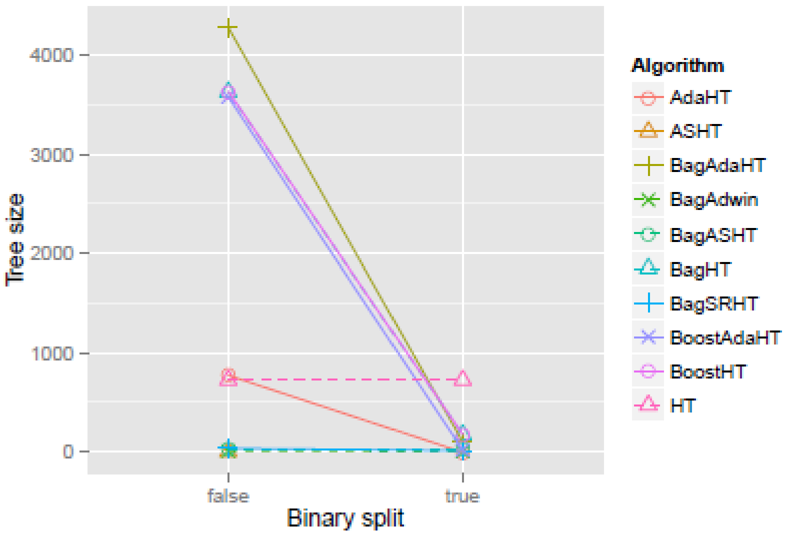


VSRBF Generator

Effect of Parameters

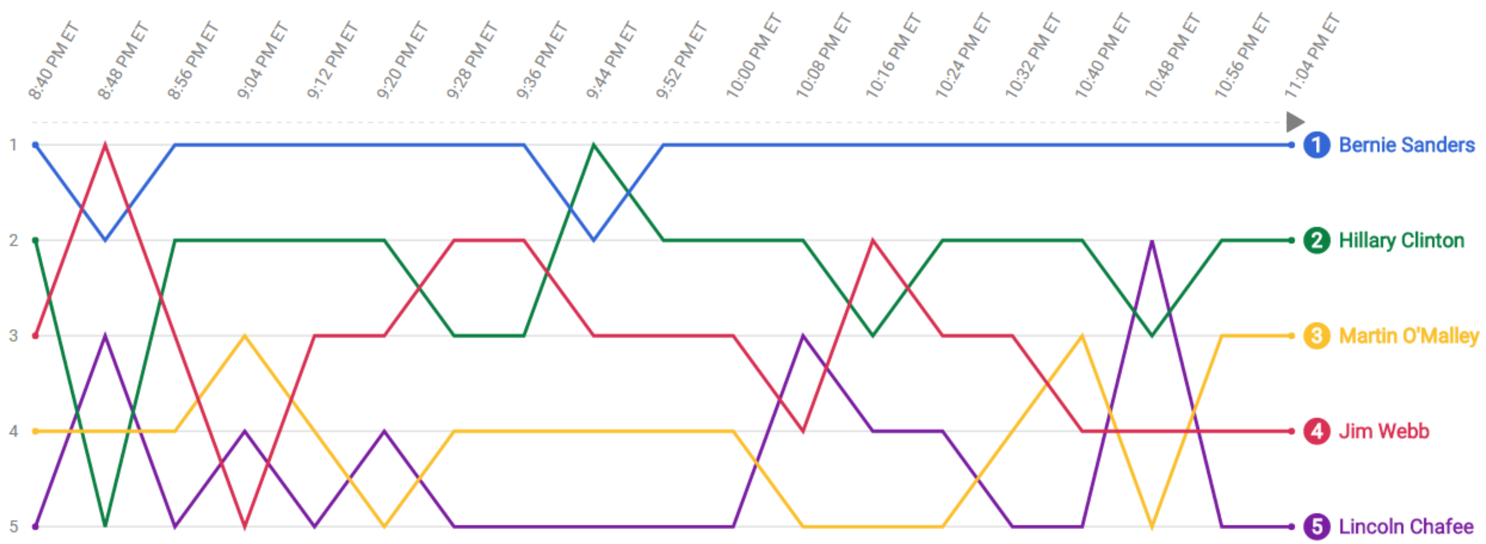
- Larger tie threshold causes massive trees
- Binary splits are nearly as effective as non-binary splits
  - Negligible loss of accuracy (1-2%)





Motivation

- Search interest in candidates and issues during the first Democratic Party debate, Oct 13, 2015
- Rank is related to the number of searches
- For Bernie Sanders number of searches was far more than Lincoln Chafee



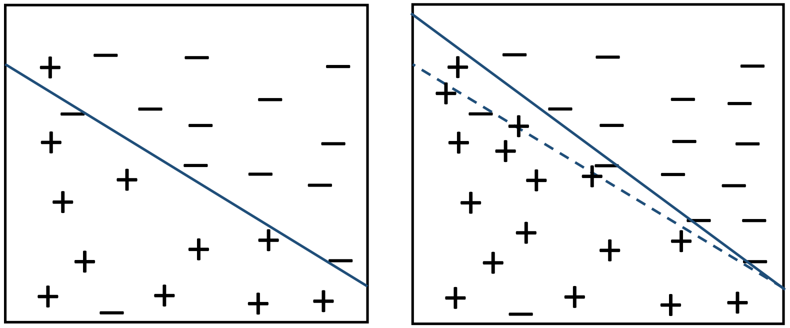
Source: google.com/trends/story/US_cu_PviRQ1ABAAD_OM_en

Contents

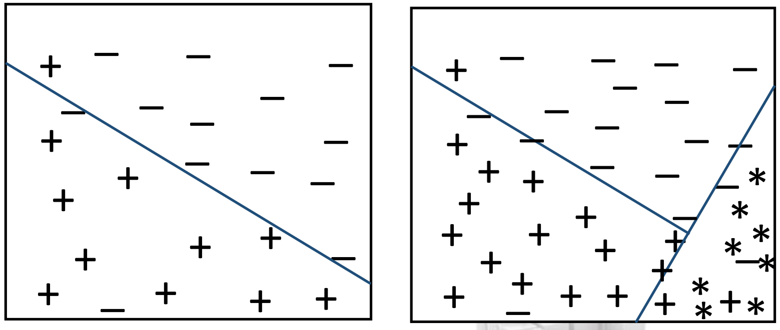
- Motivation
- Problem Statement
- Background
- A New Ensemble Approach
- Data Set Description
- Results
- Discussions

Background Challenges

- Concept drift
  - Underlying data distribution changes over time
  - Example: seasonality, faulty electrical device, etc.



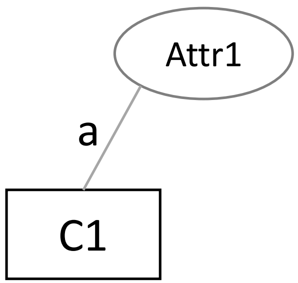
- Concept evolution
  - New class emerges
  - Example: Tweeter stream



- Concept recurrence
  - Example: new year

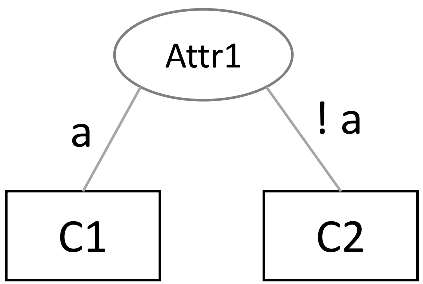
Visualization

Background Hoeffding Tree



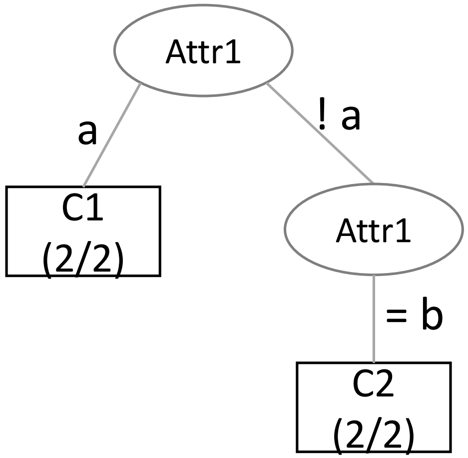
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



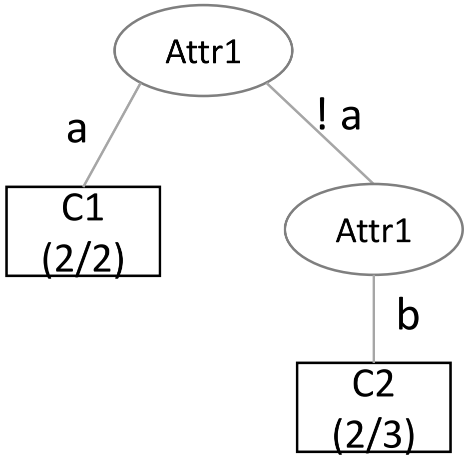
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



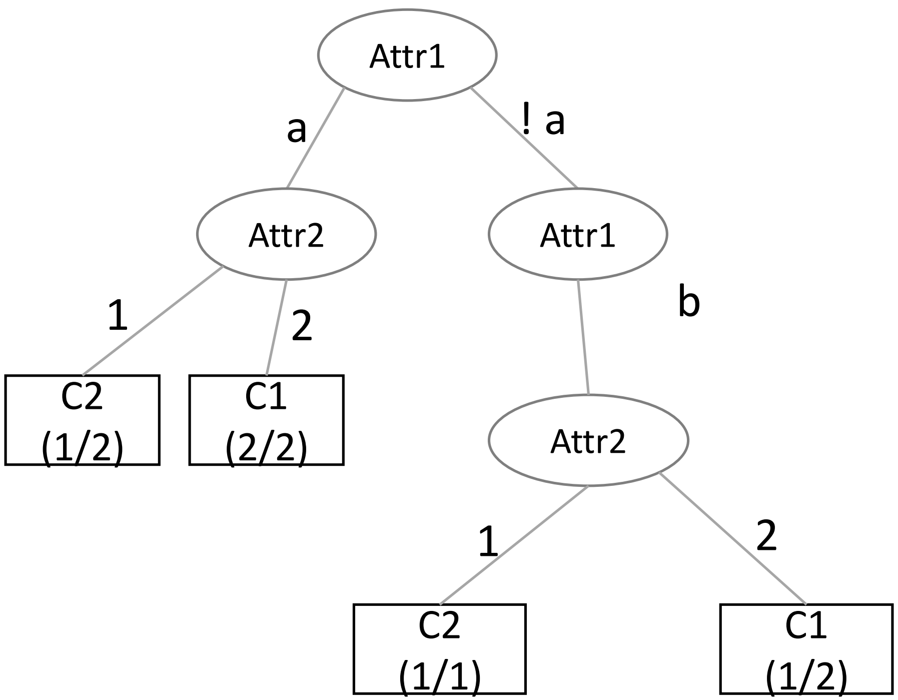
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



Attr1	**Attr2**	**Class**
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1