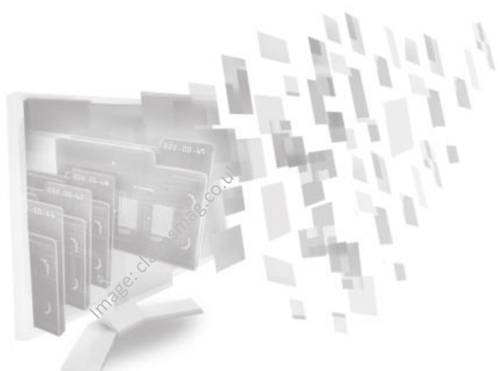
Ensemble Learning in Data Streams

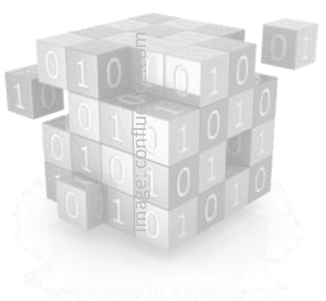
Master's thesis

Hossain **Mahmud**

Supervisor: Prof. Dr. Burkhard Rost

Advisors: Dr. Eirini Ntoutsi, Dr. Lothar Richter





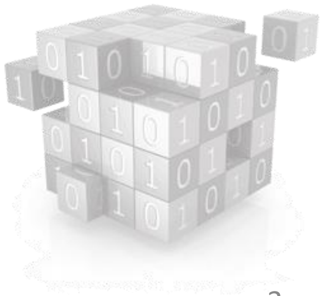
Data Stream

Stream data arrives continuously and rapidly, and if it is not processed immediately or stored, then it is lost forever. Moreover, 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 interact with it later.

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

Introduction

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



Contents

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

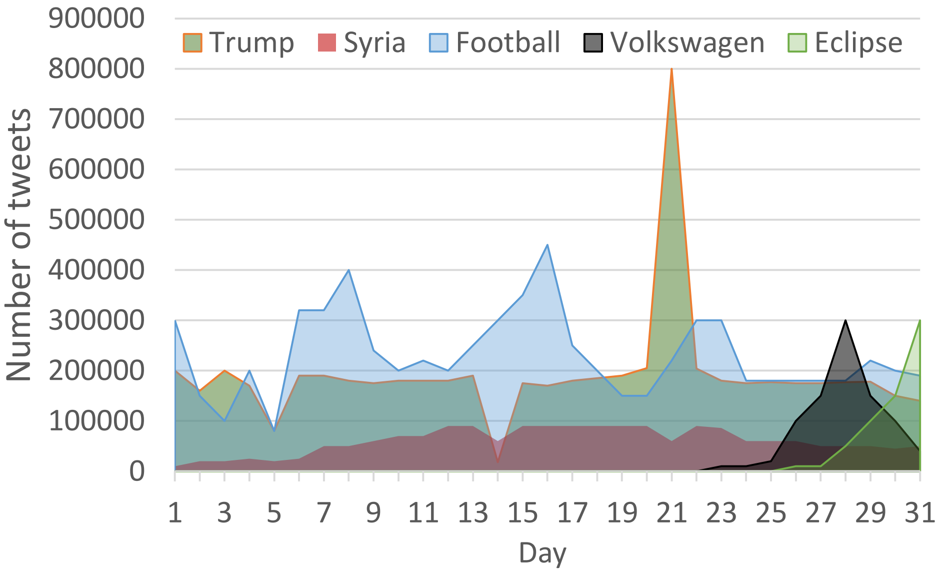
Motivation

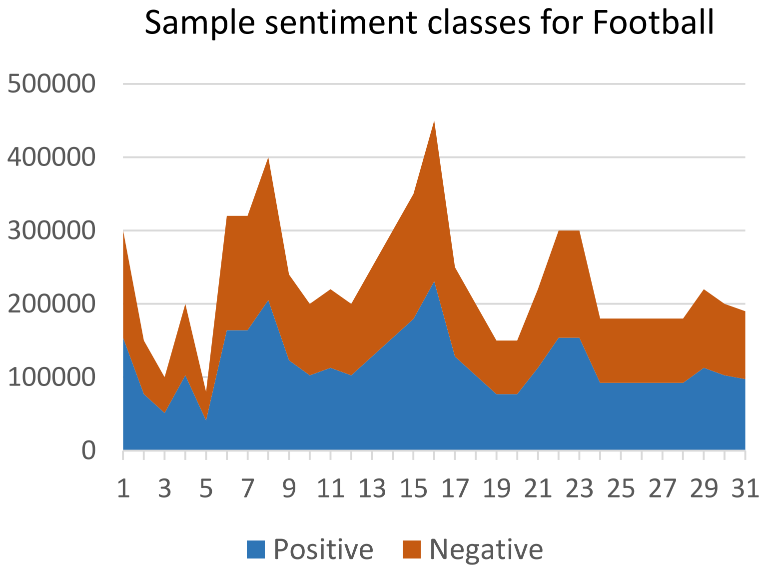
- Search interest in candidates and issues during the first Democratic Party debate, Oct 13, 2015
- Rank is related to the number of searches
- For Bernie Sanders number of searches wes far more than Lincoln Chafee

Source: google.com/trends/story/US_cu_PviRQ1ABAAD_OM_en

Motivation

- Number of tweets in Tweeter for different topics (Aug 28, 2015-Sept 28, 2015)
- The target class could be balanced even though the sources are

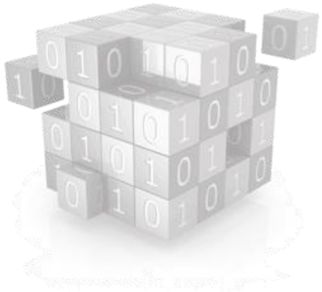




Source: topsy.com

Problem Statement

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



Existing Approaches

Assumption

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

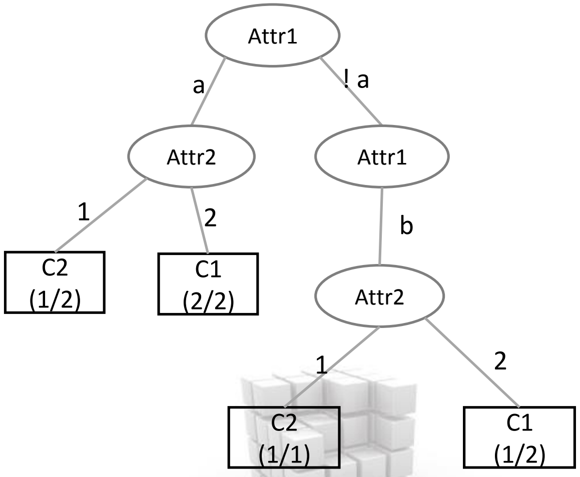
$$\epsilon \leq \sqrt{\frac{R^2 \ln(2/\delta)}{2n}}$$

- A decision taken after observing a certain amount of instances would remain the same after seeing an infinite number of instances

Hoeffding Tree (HT) [3]

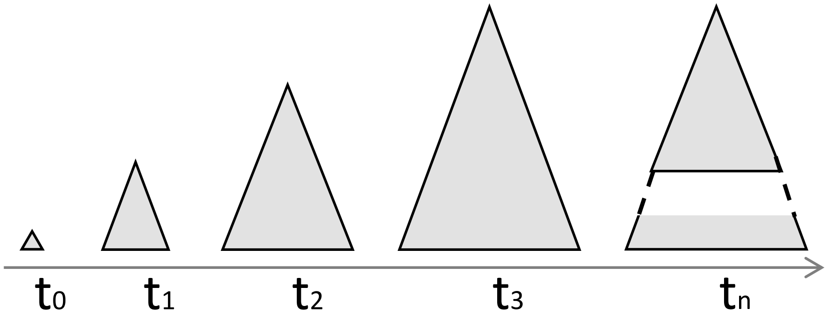
- Decide based on the data in hand
- Try to split if a node/leaf is impure
- Use information gain/ gini index to obtain best two split attribute
- If the best one performs better than the second best by at least a margin of Hoeffding bound, then split
- Could use a grace period to fasten up the process
- A tie threshold may be used to break ties between two similarly good attributes

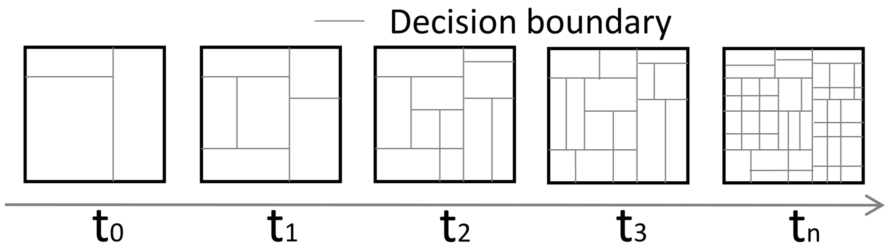
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1



Hoeffding Tree cont.

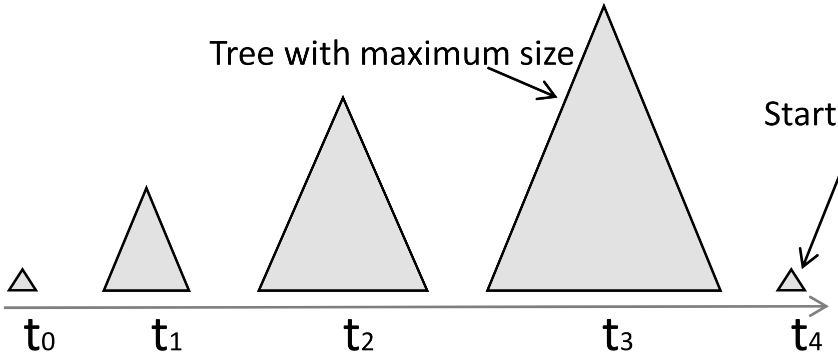
- Basic hoeffding tree
- Keeps updating decision rules with incoming data
- No rule gets deleted
- Produces redundant rules e.g., IF x < 7 AND x < 5 AND x <3 THEN ...

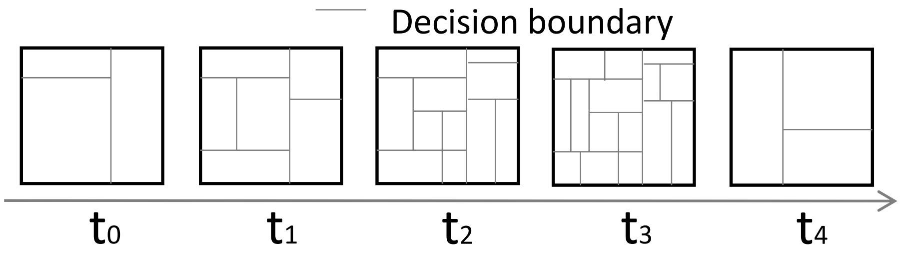




Adaptive Size HT (ASHT) [5]

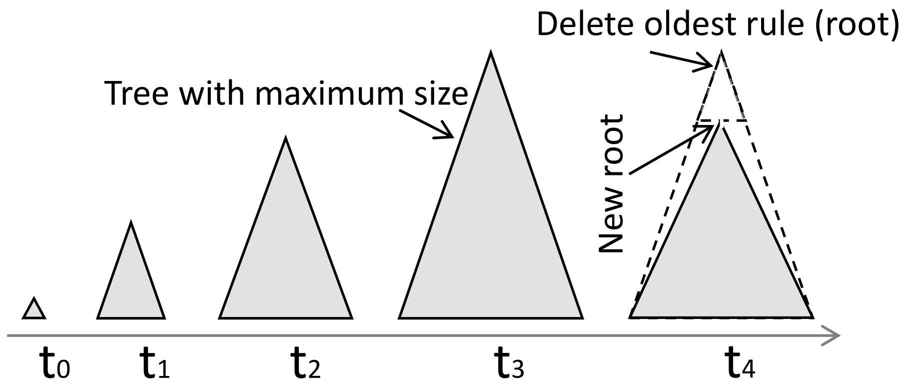
- Set bound on the maximum tree size
  - Number of decision nodes
  - Depth
- Start over when limit is reached
- Looses all information learned thus far

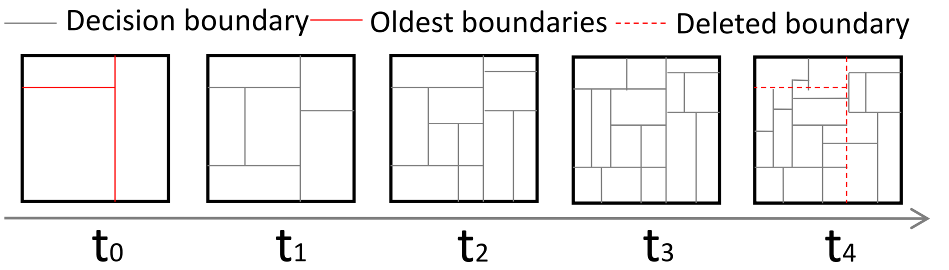


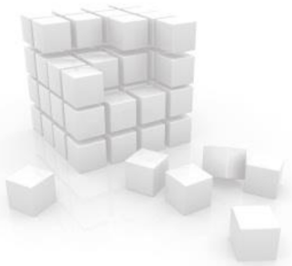


Adaptive Size HT (ASHT)

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

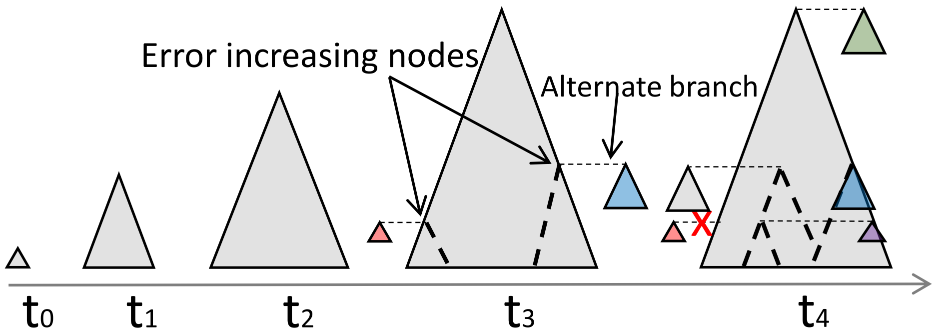


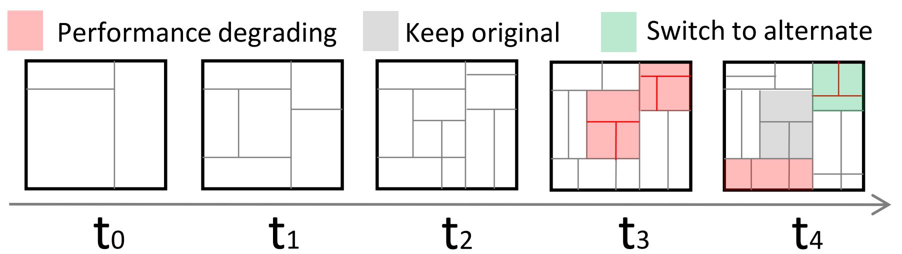




Adaptive HT (AdaHT) [4]

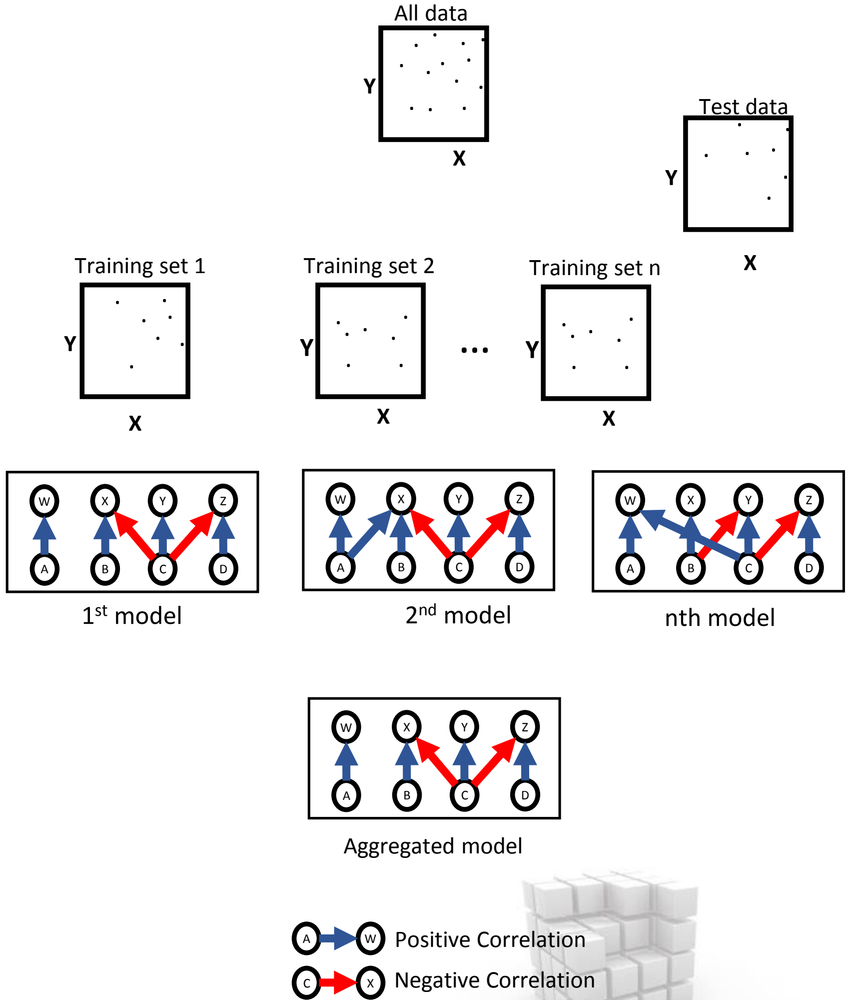
- Start maintaining alternate sub-tree when a node starts performing worse than previous
- When new sub-tree starts performing better, replace the original
- If original sub-tree keeps performing better, delete alternate tree





Bagging Ensemble [6,7]

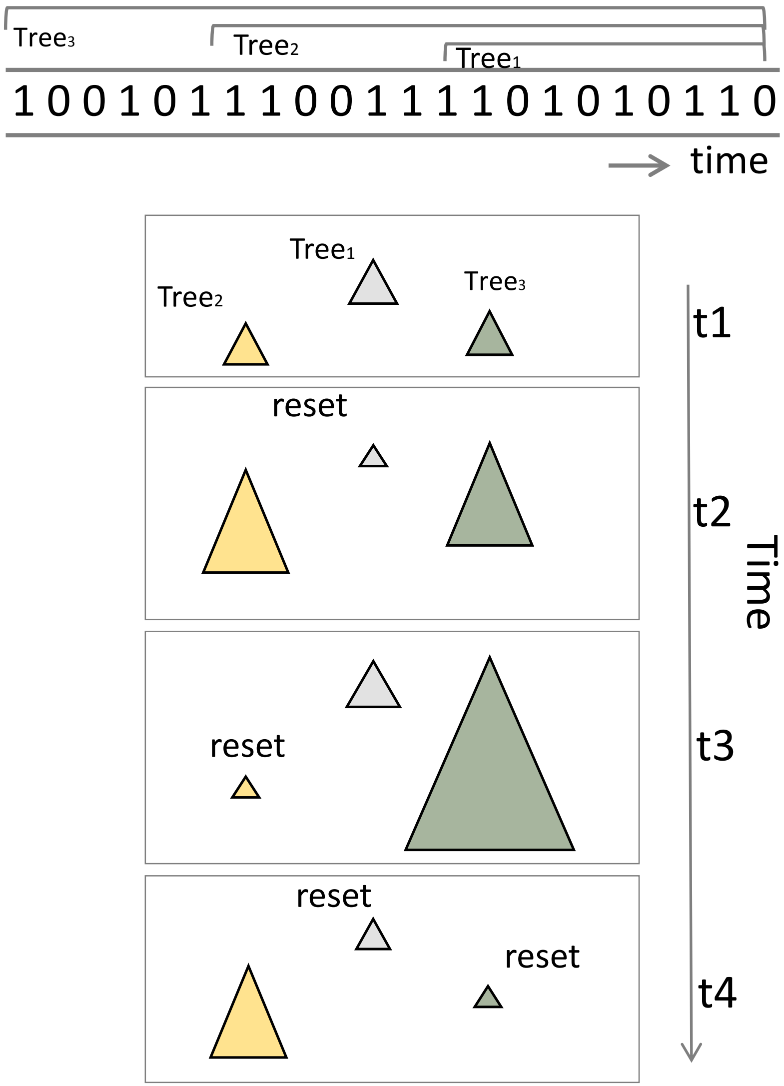
- Generate n training sets by random sampling of original training set
- Learn a model with each training set
- Use majority voting for classifying test data



Bagging with ASHT (BagASHT) [5]

- Maintain an ensemble of HTs of different size limit
- Allows building model for different time-frames
- Everytime a larger tree gets reset, ensemble looses significant information

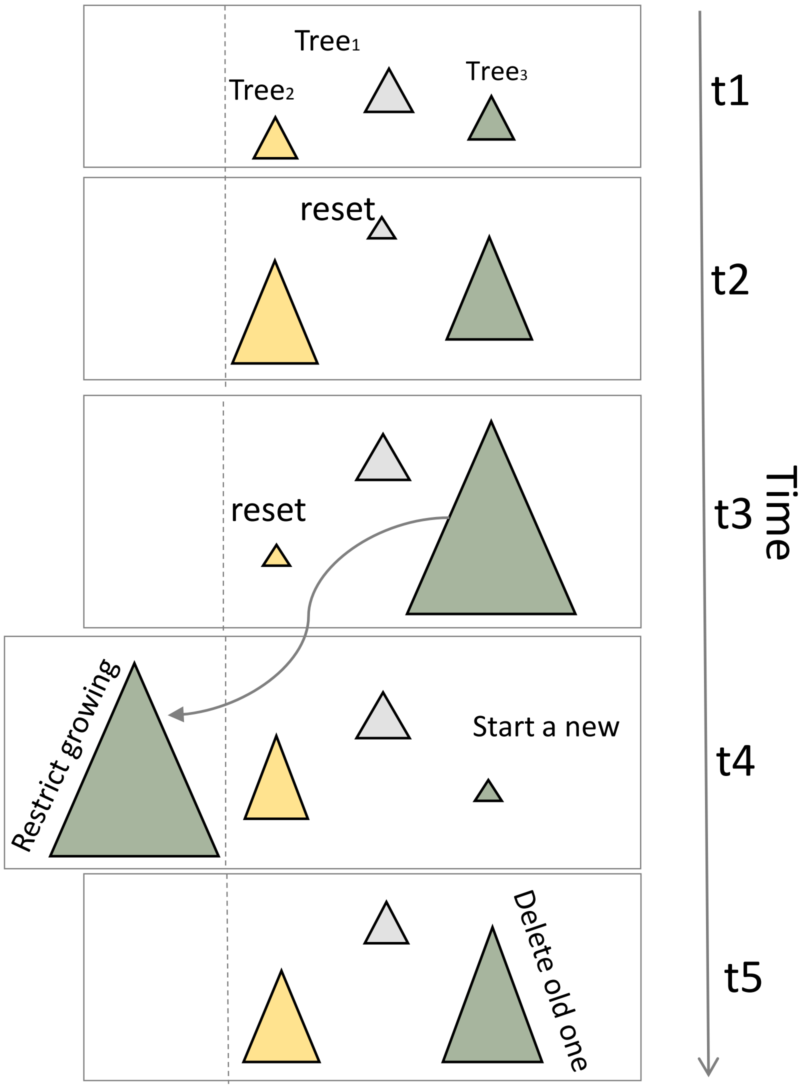
- Bagging with Adaptive Sized Hoeffding Tree (BagASHT)



New Approach

Carry-over Bagging (BagSRHT)

- Defer resetting of larger trees
- Once the limit is reached
  - Move trees to the extra list
  - Retrict it from growing
  - Start maintaining another tree
  - Take vote from both
- Delete oldest from extra list
  - When new one is large enough
  - Or more trees reached their limit
- Use alternate sub-tree method



Algorithm Summary

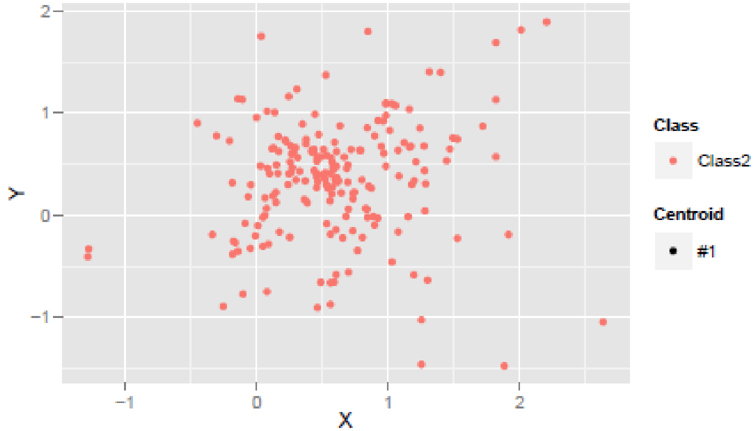
- HT – Hoeffding Tree – Grow indefinitely
- AdaHT – Adaptive HT – Maintain alternate sub-trees
- ASHT – Adaptive Size HT – Reset if limit reached
- SRHT – Size Restricted HT (new method) – defer reset

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

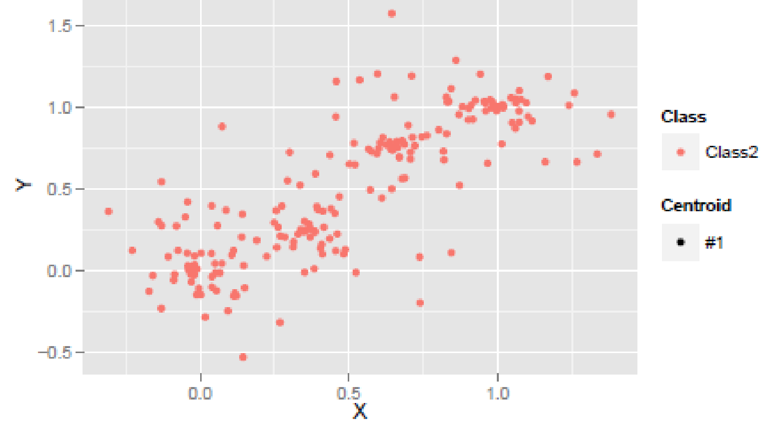
Data Generation

- Most existing generators uses randomized approach
- 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 entroids
  - Instances are generated by selecting a centroid at random (weighted), and choosing a point using normal distribution
- Not possible to generate varying speed data set

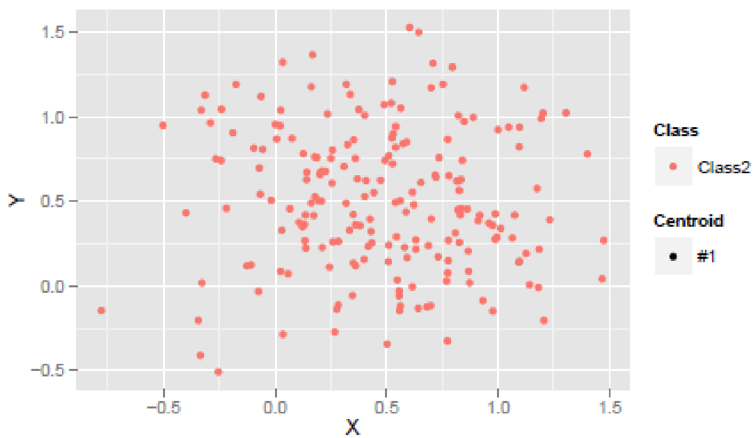
RandRBF Data Set



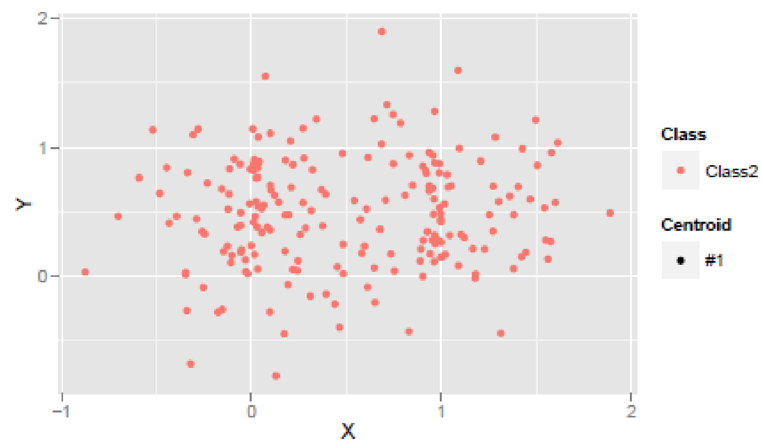
Drift Coefficient 0.0



Drift Coefficient 0.01



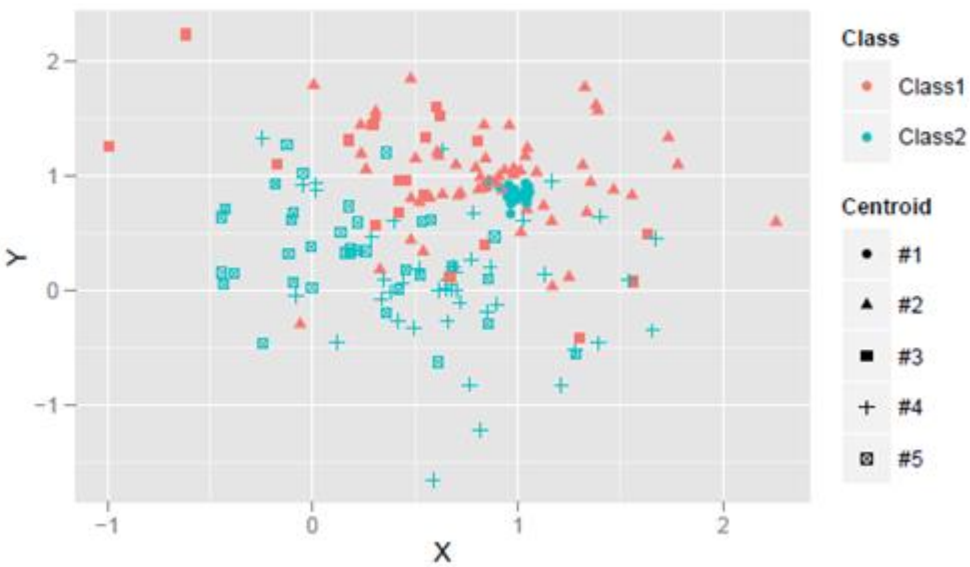
Drift Coefficient 0.1

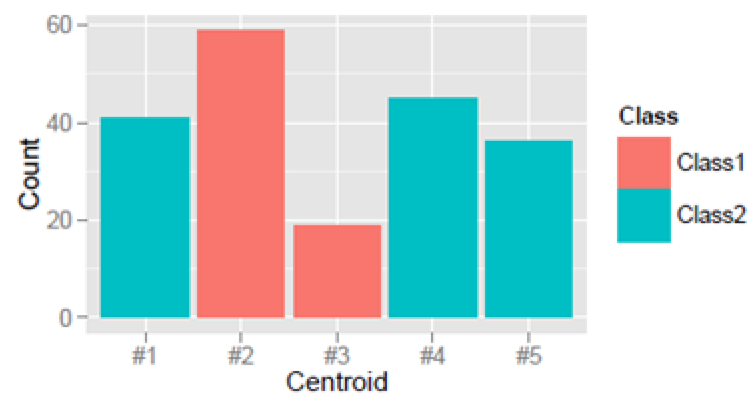


Drift Coefficient 1.0

Data Set

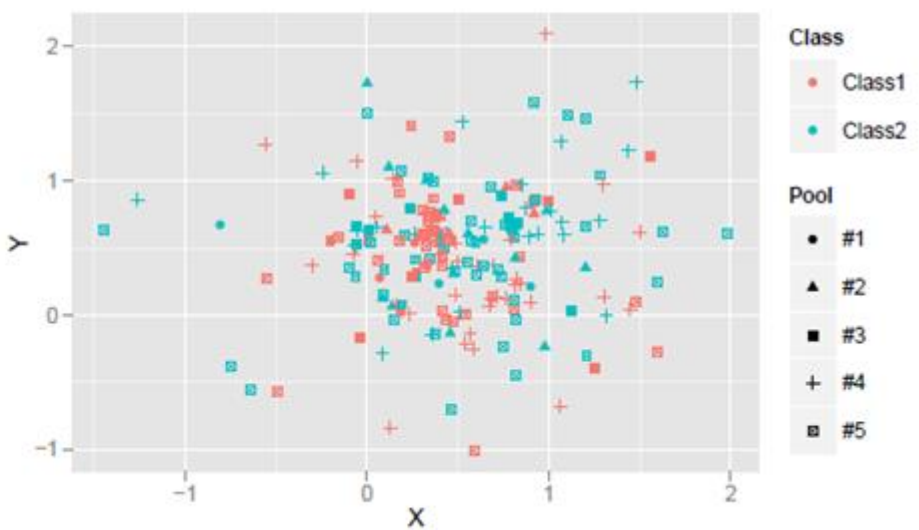
- Centroids contributes to the final weights depending on their weights
- All centroids are active all the time

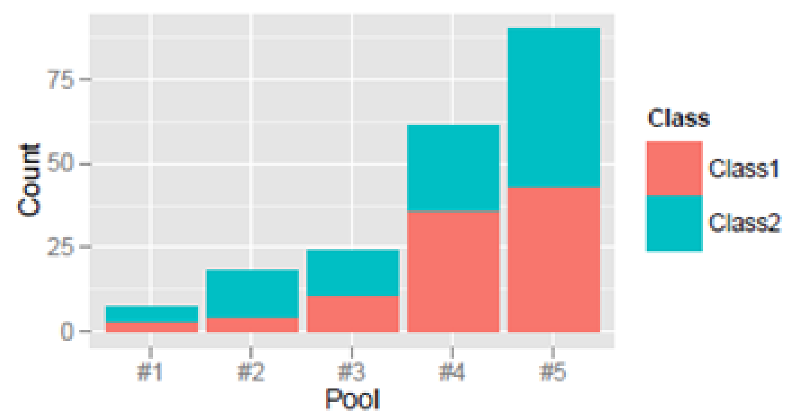




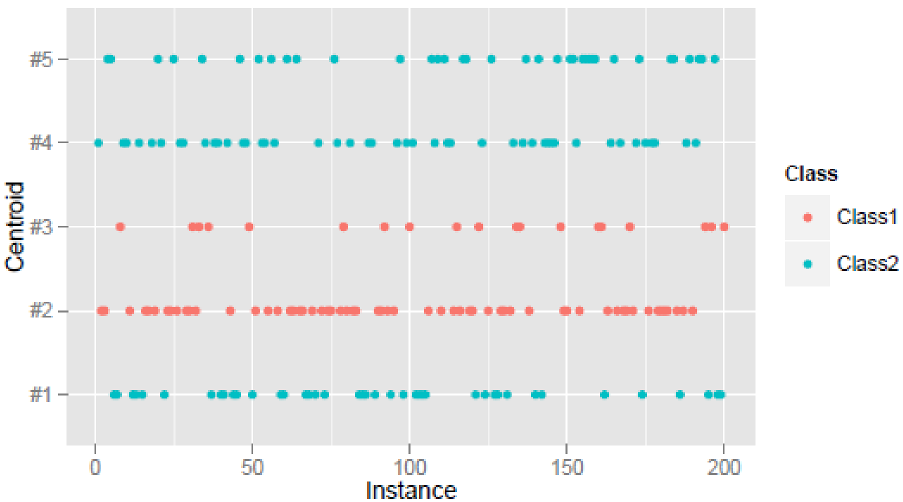
Varying Speed RBF Generator

- Replace the concept of centroids with concept of pools
- Each pool contains a number of centroid and has different activation and contribution rate

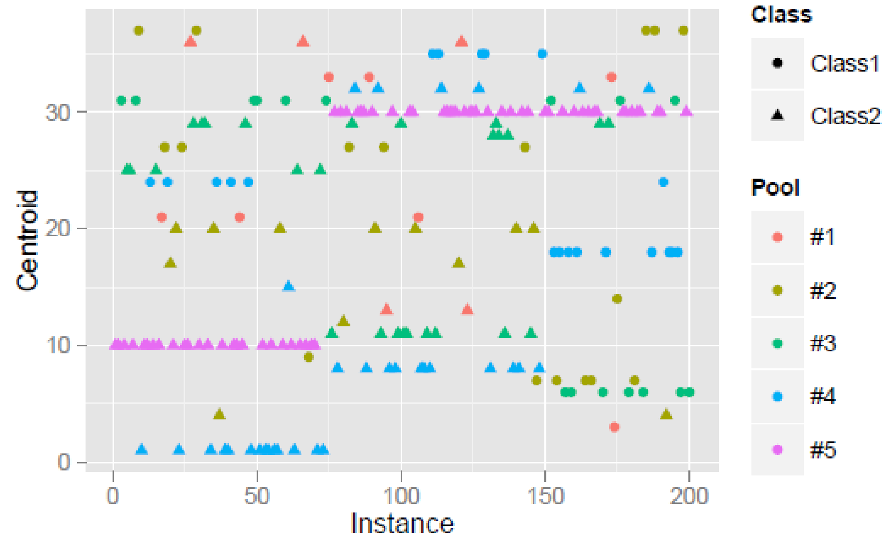


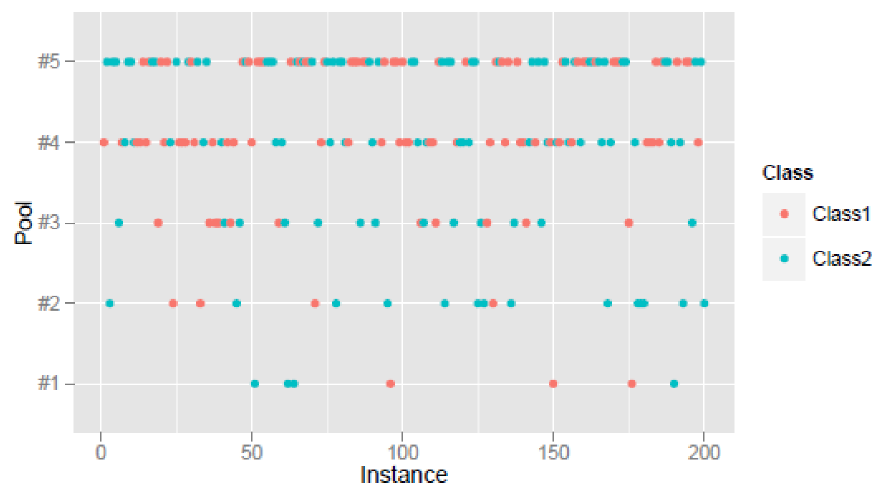


Data Set Comparison



Random RBF Generator





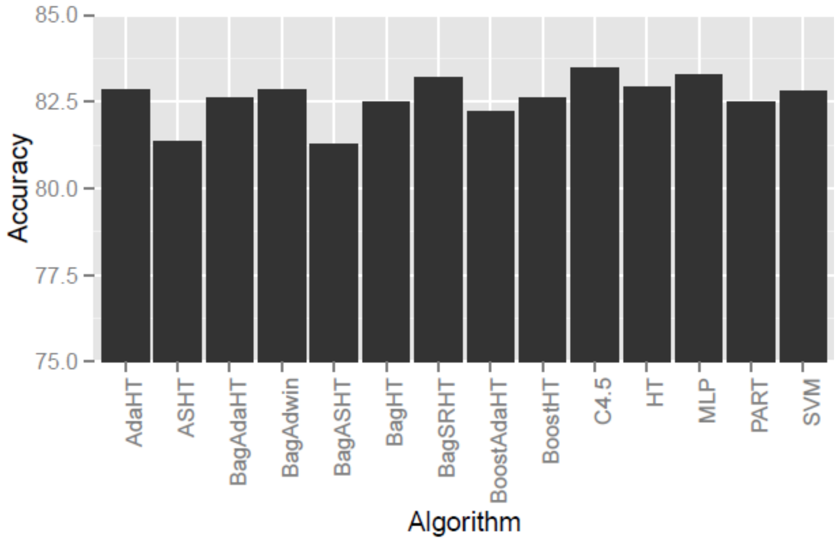
VSRBF Generator

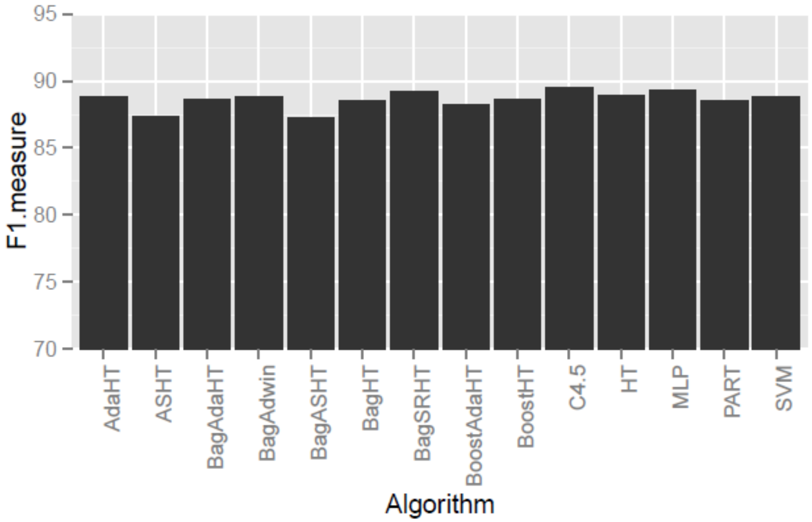
Evaluation

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

Comparison with Batch Approaches

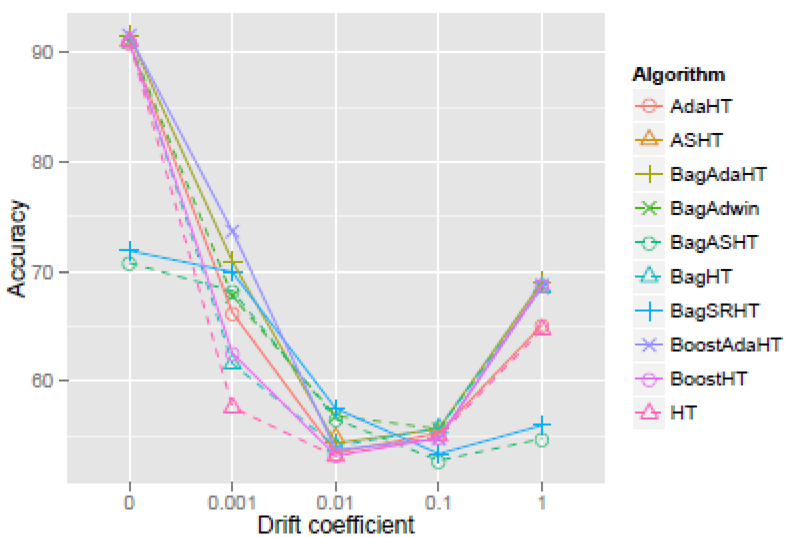
- 10 fold cross validation for batch approaches
- ADWIN variants and BagSRHT reach closest to C4.5

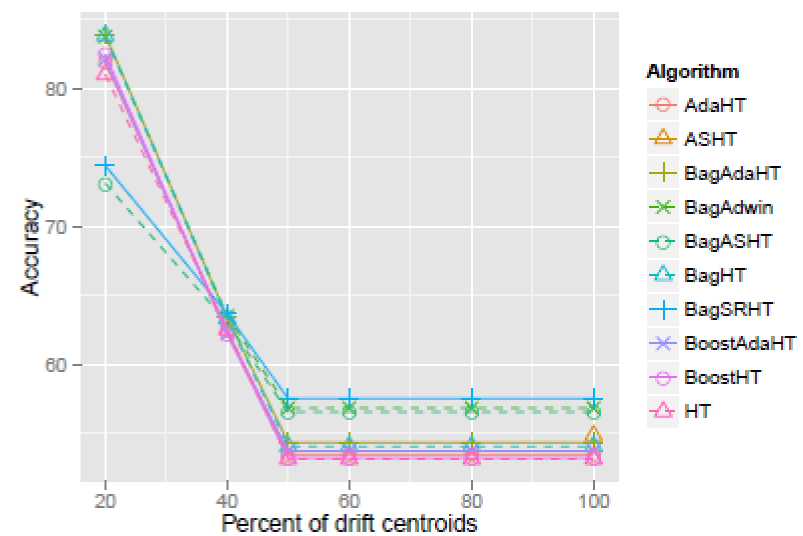




Effect of Parameters Data Generation

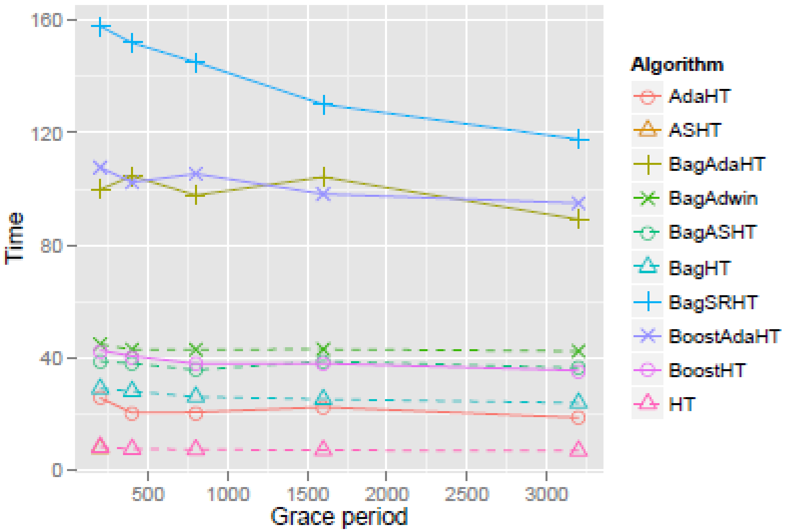
- Without any drift, all method performs the same
- With small drift ADWIN and boost variants performs best
- BagSRHT and BagASHT performs similar in all cases

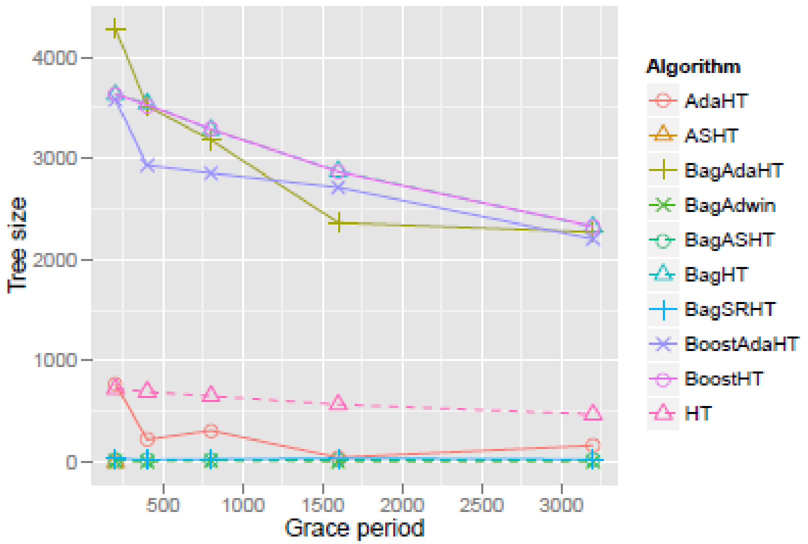




Effect of Parameters Algorithm

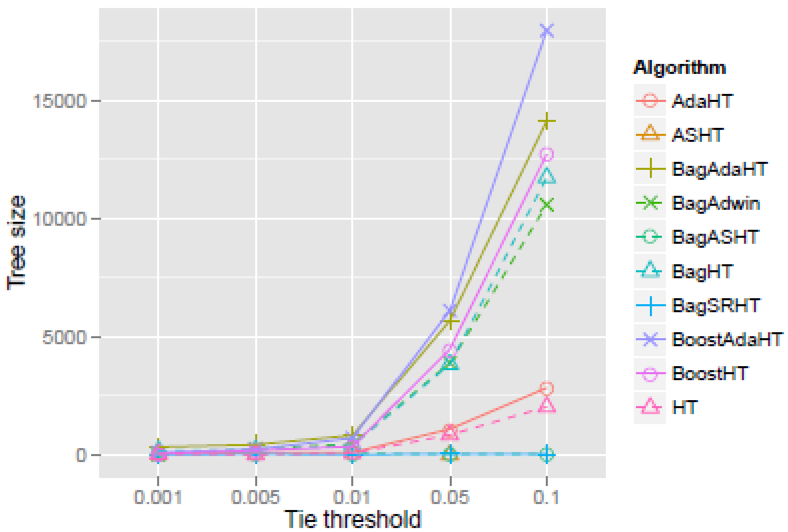
- Grece period effectively reduces processing time and tree size

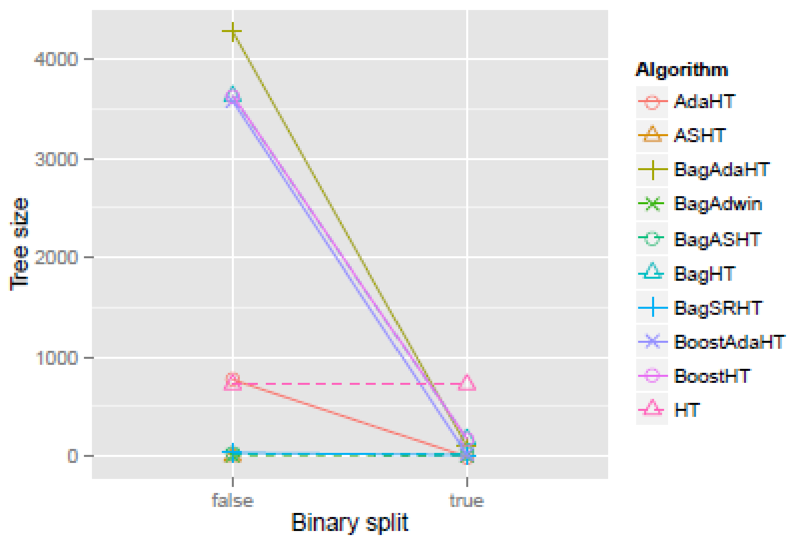




Effect of Parameters Algorithm

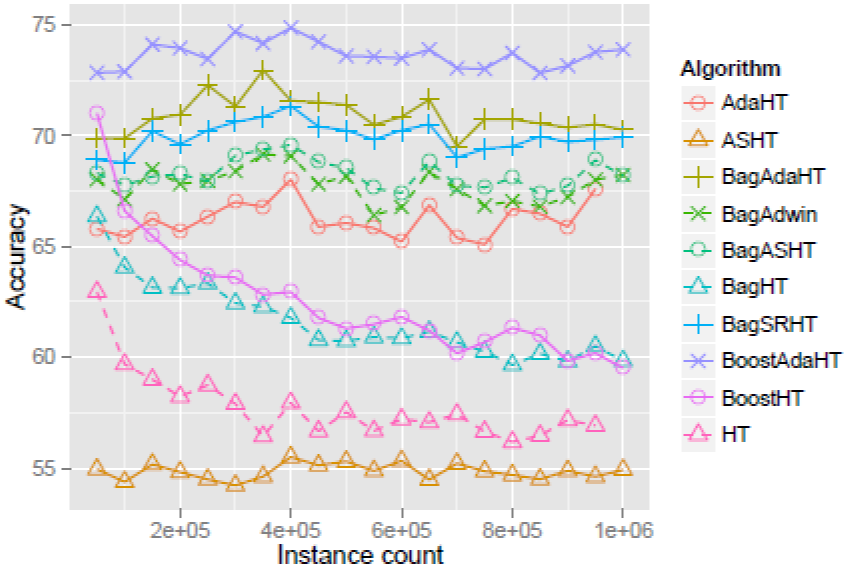
- Larger tie threshold causes massive trees
- Binary splits are nearly as effective as non-binary splits
  - Negligible loss of accuracy (1-2%)



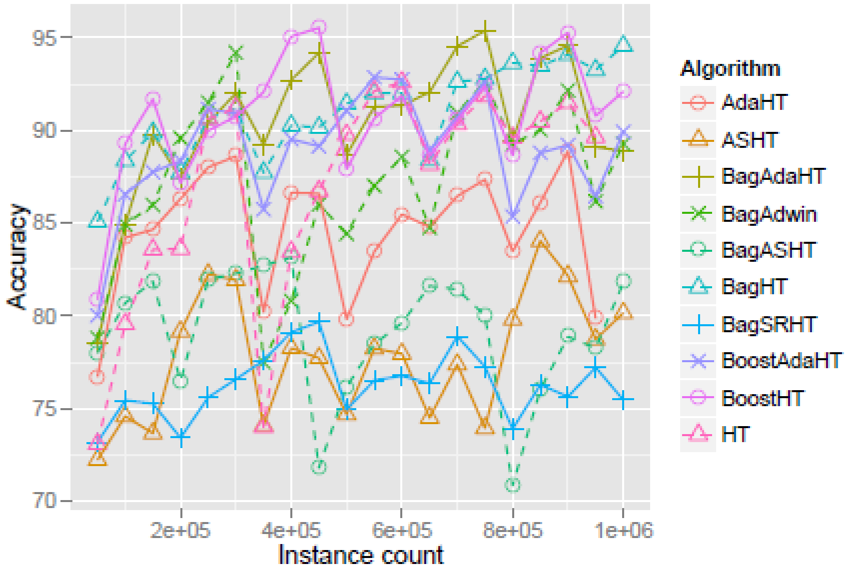


BagSRHT vs BagASHT

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



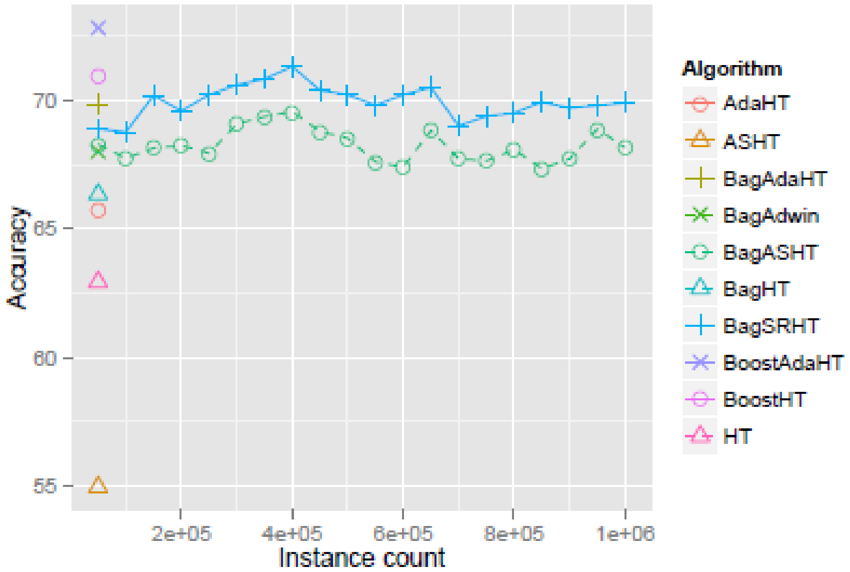
Random RBF



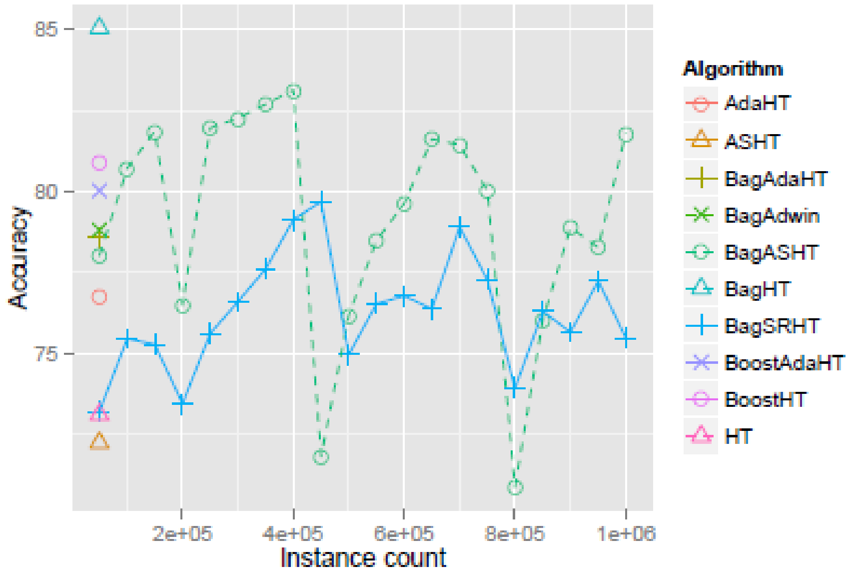
VSRBF

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



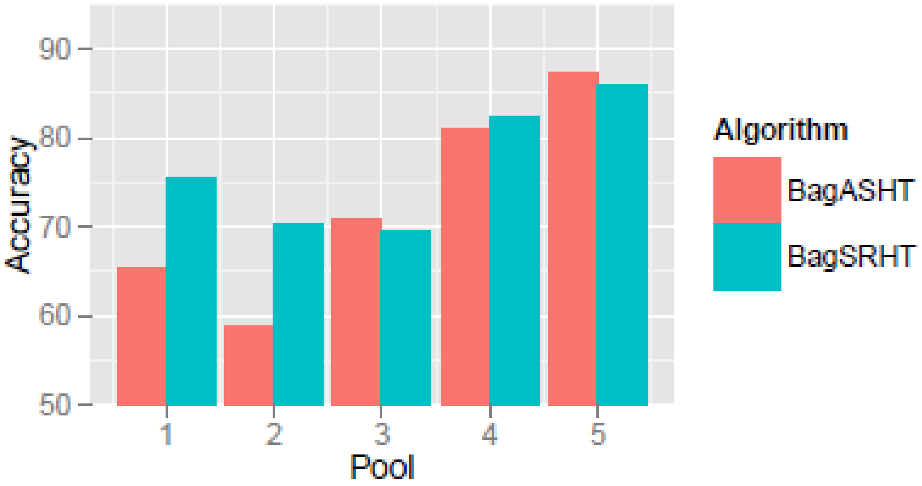
Random RBF



VSRBF

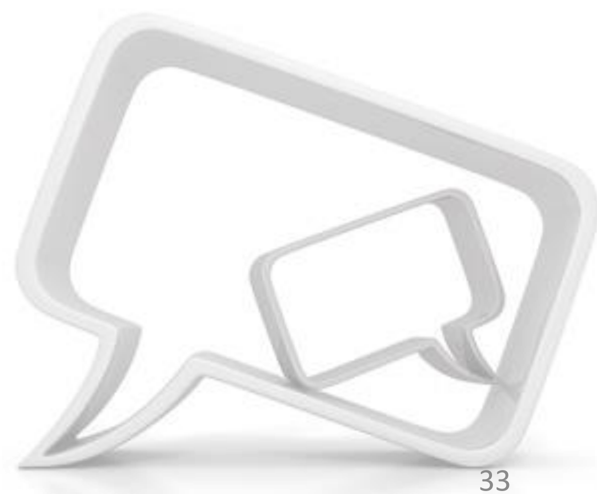
BagSRHT vs BagASHT

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



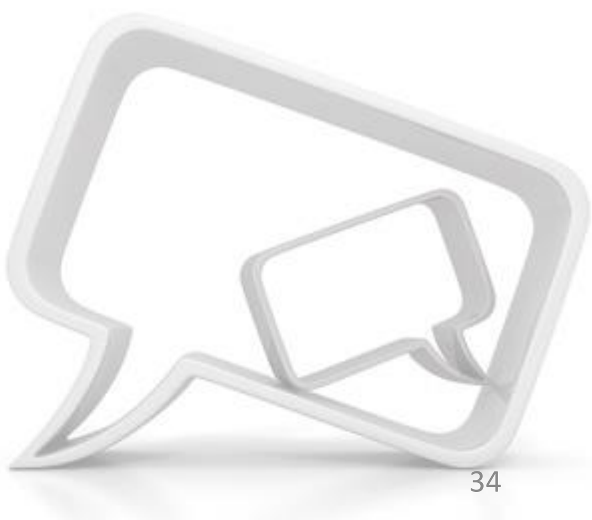
Conclusion

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



Future Work

- Fitting the model into text streams
  - The motivation is taken from text based streams
  - Evaluation is performed with numeric data for thesis scope
- Devising approach to select a subset of the incoming streams 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.

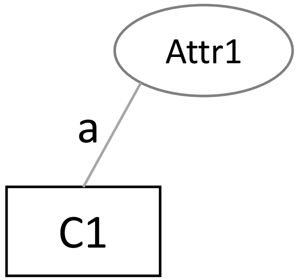

THANK
YOU!

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

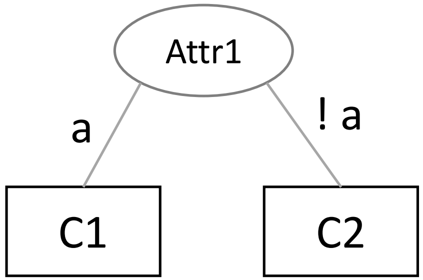
Visualization

Background Hoeffding Tree



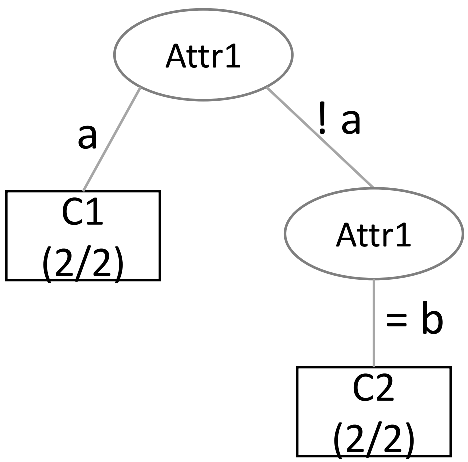
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



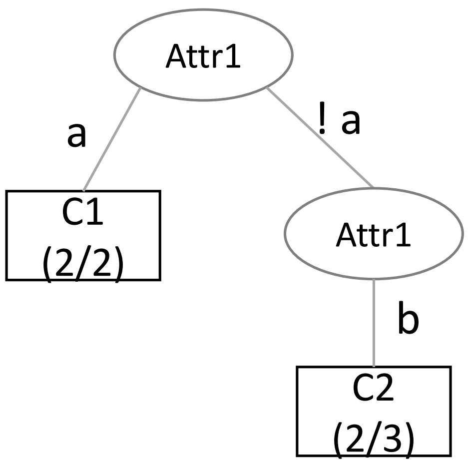
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

Background Hoeffding Tree



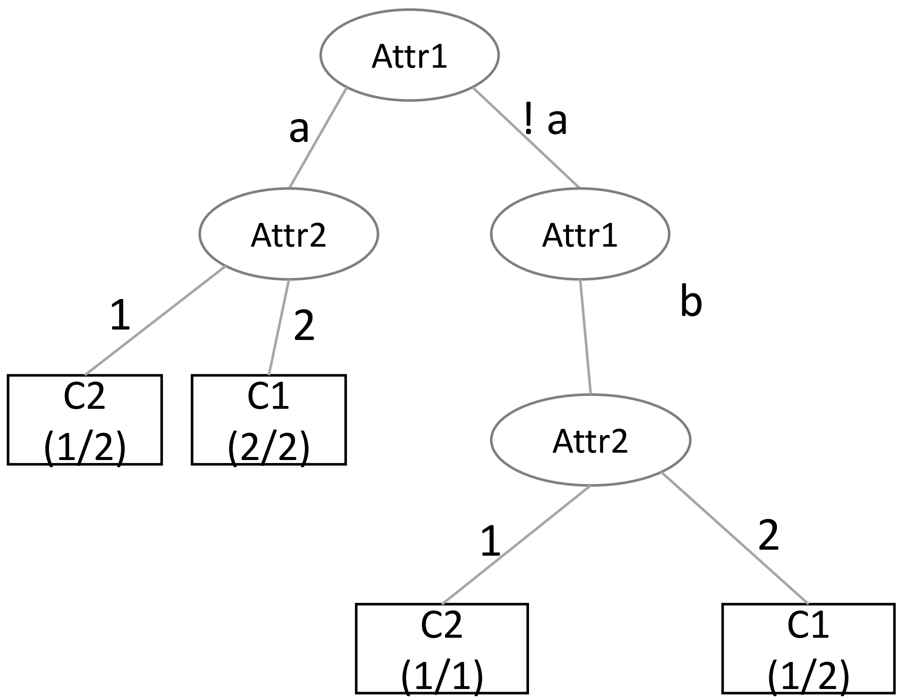
Attr1	Attr2	Class
a	1	C1
a	2	C1
b	1	C2
b	2	C2
b	2	C1
a	1	C2
a	2	C1

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

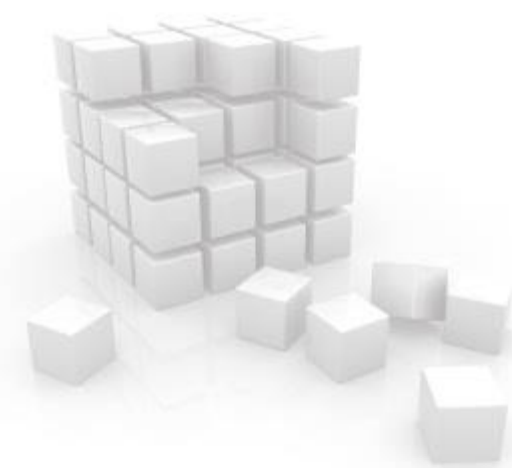
Background Hoeffding Tree



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

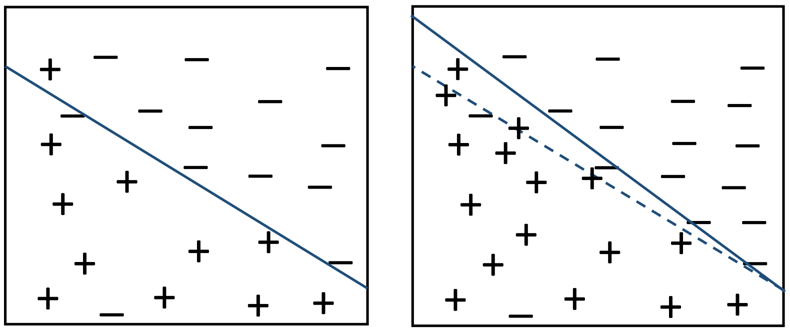
Background Challenges

- Speed of data arrival
  - …
- Single-pass learning
  - …
- Memory restriction
  - …
- Lack of labeled data
  - …

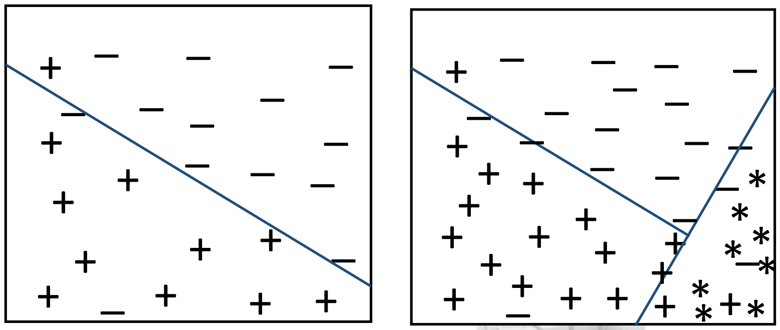


Background Challenges

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



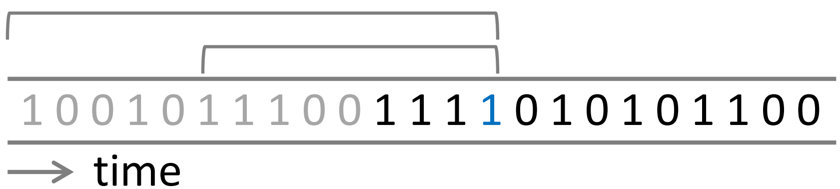
- Concept evolution
  - New class emerges
  - Example: Tweeter stream



- Concept recurrance
  - Example: new year

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



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

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

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

Goals

Gratitude