电子科技大学

University of Electronic Science and Technology of China

Learning Model Trees From Evolving Data Streams (FIMT-DD)

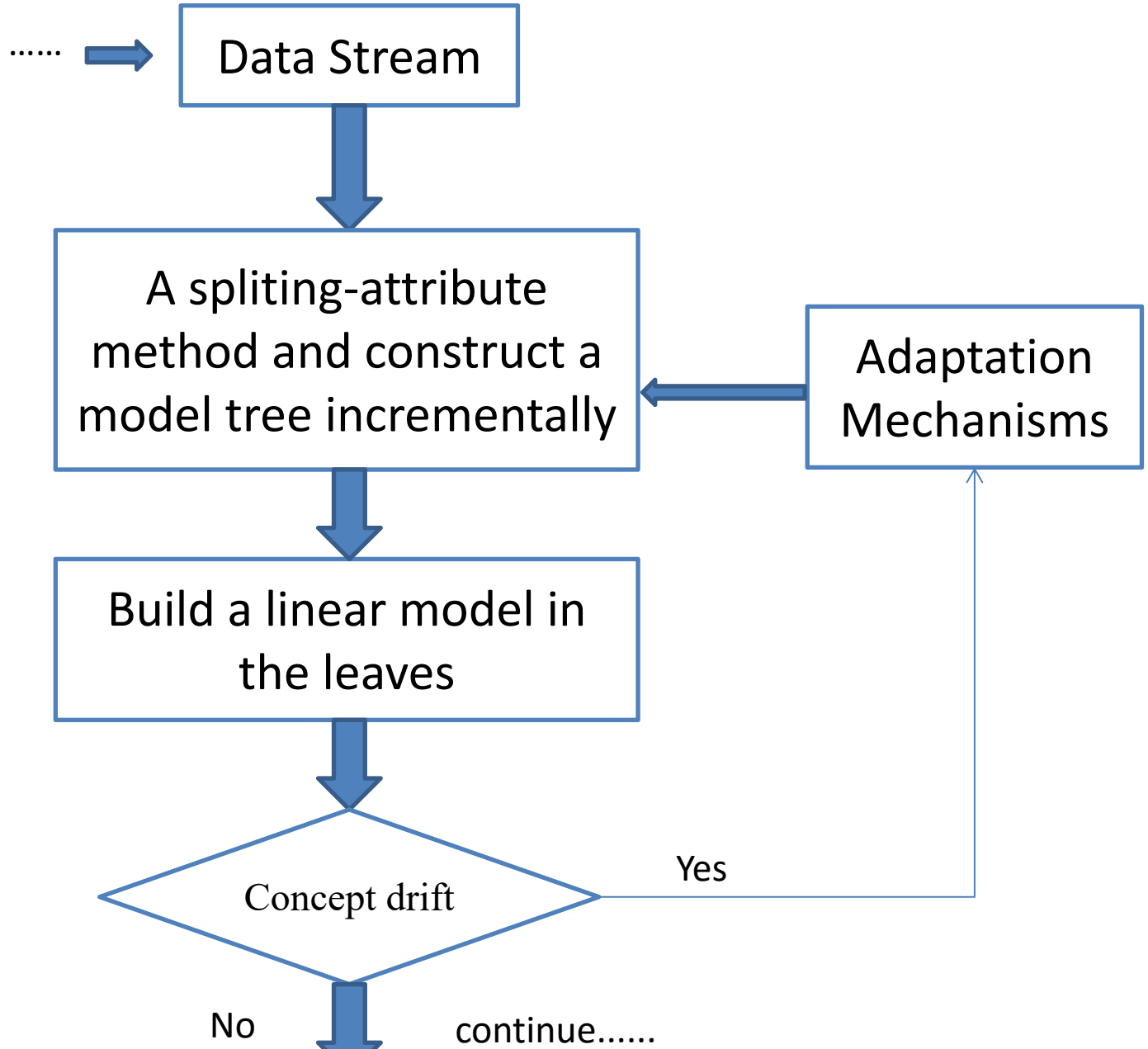
张　恒

hengzhang64@gmail.com

outline

- Introduction
- Splitting-attribute selection
- Numerical attributes consuctuct a tree
- Computing the linear models in the leaves
- Change detection and adaptation mechanisms
- Conclusion

Introduction



Split criterion

There many split criterion strategies,in FIMT-DD the splitting criterion was used is SDR(Standard Deviation Reduction) which can be done incrementally.

For example: the dataset S of size N, hA in attribute A will split the dataset into two subset SL and SR (i.e. S = SL *U* SR; N = NL + NR)

The formula for measuring split of SDR hA is :

$$\mathrm{SDR}(hA) = sd(S) - \frac{NL}{N} sd(S_L) - \frac{NR}{N} sd(S_R)$$

$$sd(S) = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} (yi - \bar{y})^2\right)} = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} yi^2 - \frac{1}{N}\left(\sum_{i=1}^{N} yi\right)^2\right)}$$

If hA is the best split of attribute A and hB is the second best split of attribute B,futher consider the ratio of the SDR values for the best two split(hA and hB) as a real-valued random variable *r* That is:

$$r = \mathrm{SDR}(h_B)/\mathrm{SDR}(h_A)$$

r between 0 and 1,in the stream each value $r_1, r_2, \ldots, r_N$
Use the ***Hoeffding inequality :*** $P(\left|\overline{X} - E[\overline{X}]\right| \geq \varepsilon) \leq e^{-2N\varepsilon^2}$
confidence: 1-δ N random i.i.d in the range R is within distance ε of the true mean

$$N \geq -\frac{\log(\delta/2)}{2\varepsilon^2}$$

Why

$$r^+ = r + \varepsilon \quad \text{and} \quad r^- = r - \varepsilon \quad \text{and} \quad r^- \leq r_{true} \leq r^+$$

The best observed attribute over a portion of the data is really the best over the whole distribution. Therefore with confidence $1-\delta$ the split hA is deemed as the best one. In this case, the splitting criterion is satisfied and the split hA can be applied.

Numerical attributes (E-BST)

Selection of the distribution tree is dependent upon the number of points of divisions,traditional methods require initialization before the data is processed. In FIMT-DD, there use E-BST (Extended Binary Search Tree)

E-BST using two arrays of 3 elements :

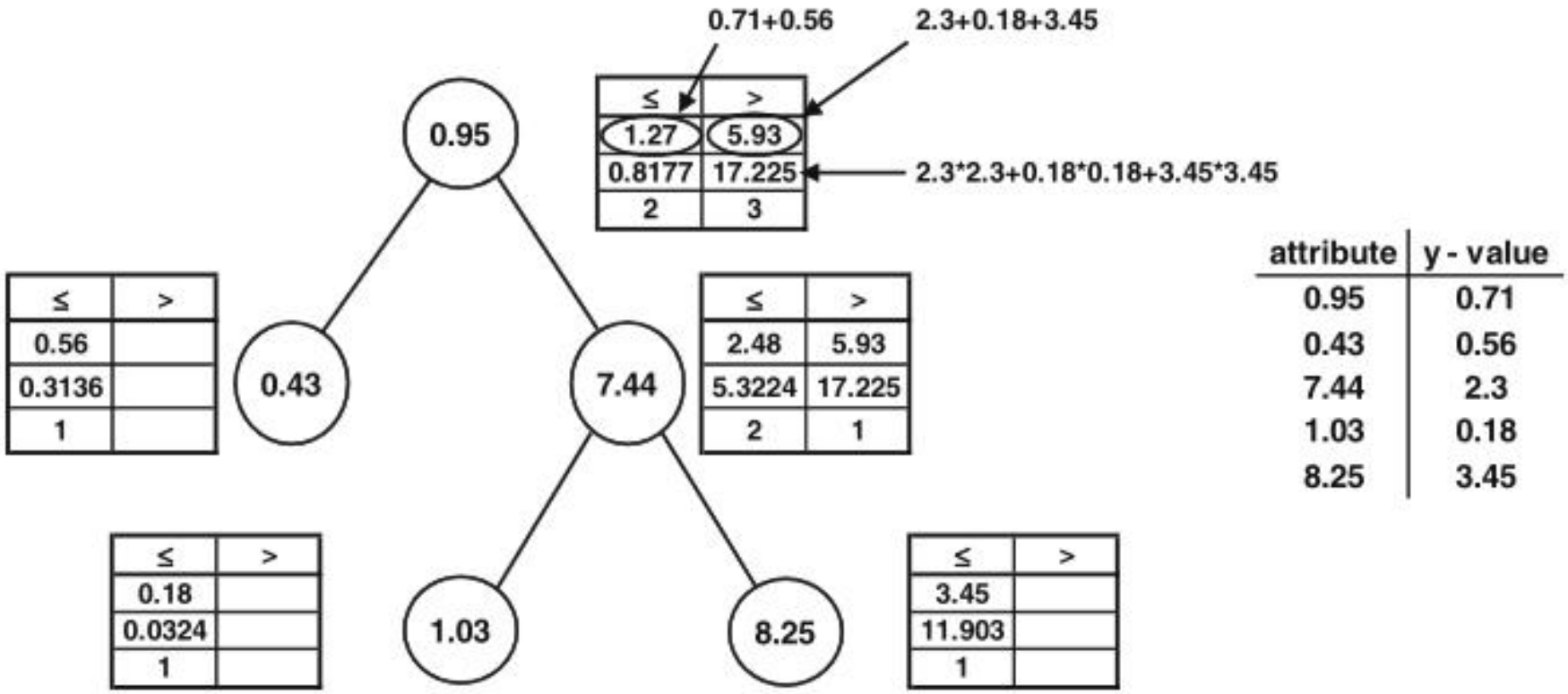


Fig. 2 Illustrative example of a E-BST constructed from the given table of pair of values (*right*)

FindBestSplit

1. IF LeftChiLd exists
2. Call FindBestSplit(LeftChild)
3. Update the sums and counts for computing SDR of the split
4. IF maxSDR < ComputeSDR(current node)
5. Update maxSDR
6. IF RightChild exists
7. Call FindBestSplit(RightChild)
8. Update the sums and counts for computing SDR of the split
9. END

Prun

SDR(h_1) : the greatest reduction in the standard deviation at the last evaluation
SDR(h_2) : the second hightest reduction over a different attribute.

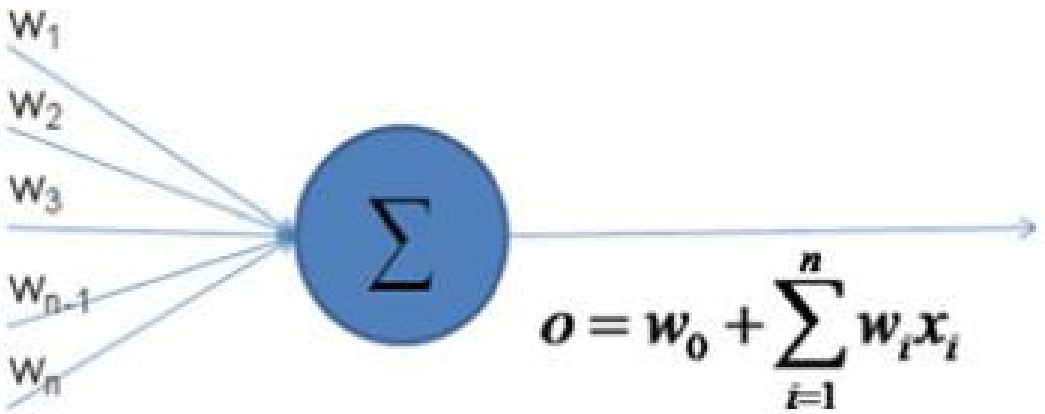
$$r = SDR(h_2) / SDR(h_1)$$

for each split point with $SDR(h_i) / SDR(h_1) < r - 2\varepsilon$ is considered bad split and it will be pruned

When a node is removed from the tree no imformation is lost because its sums and counts ars held by upper nodes

Linear models in leaves

Given a data stream $< x_i, y_i >$,the perceptron regression:



The difference is the weights are updated when every data arrival.

$$\omega_{i+1} = \omega_i + \eta(o - y)x_i$$

o:predict value y:real value η :learning rate

$$\eta = \frac{\eta_0}{(1 + \eta_d)}$$

The initial learning rate η_0 and the learning rate decy parameter η_d should be set to appropriately(e.g., η_0 =0.1 and η_d = 0.005)

add:Each perceptron learning phase is done in parallel with the addition of nodes

Change detection

The detection mechanism that we propose is on-line and enables local change detection.monitor the evolution of the error at every region of the instance space.

Update only the affected parts of the model.

PH(Page–Hinckley)

$$m_t = \sum_{t=1}^{T} (x_t - \bar{x}_t - \alpha)$$

$$\bar{x}_t = \frac{1}{t} \sum_{l=1}^{t} x_l$$

$$M_T = \min\{m_t, t = 1, \ldots, T\}$$

m_T : cumulative variable

M_T :minimum value of m_T

At every moment, the PHtest monitors the difference between the minimum M_T and m_T :

$$PH_T = m_T - M_T$$

The parameter α corresponds to the minimal absolute value of the amplitude of the jump to be detected

When this difference is greater than a given threshold(λ), The threshold parameter λ depends on the admissible false alarm rate. Increasing λ entails fewer false alarms, but might miss some changes.

A general guideline, when α has a smaller value λ should have a larger value: This is to reduce the number of false alarms.

TD AND BU

Top-Down (TD) method:If the error is computed from the node where the PH test in the node, the error will passing from node to leaf.

Therefore, the loss will be monitored in the direction from the top towards the “bottom” of the tree.

Bottom-Up (BU) method:If the error is computed using the prediction in the leaf, the example must first reach the leaf. The computed difference at the leaf will be then back-propagated to the root node.

While back-propagating the error of the PH tests located in the internal nodes will monitor the evolution.

Adaption

There are three possible adaptation strategies for model trees:

1. **Further splitting and growing** new sub-trees to the old structure.

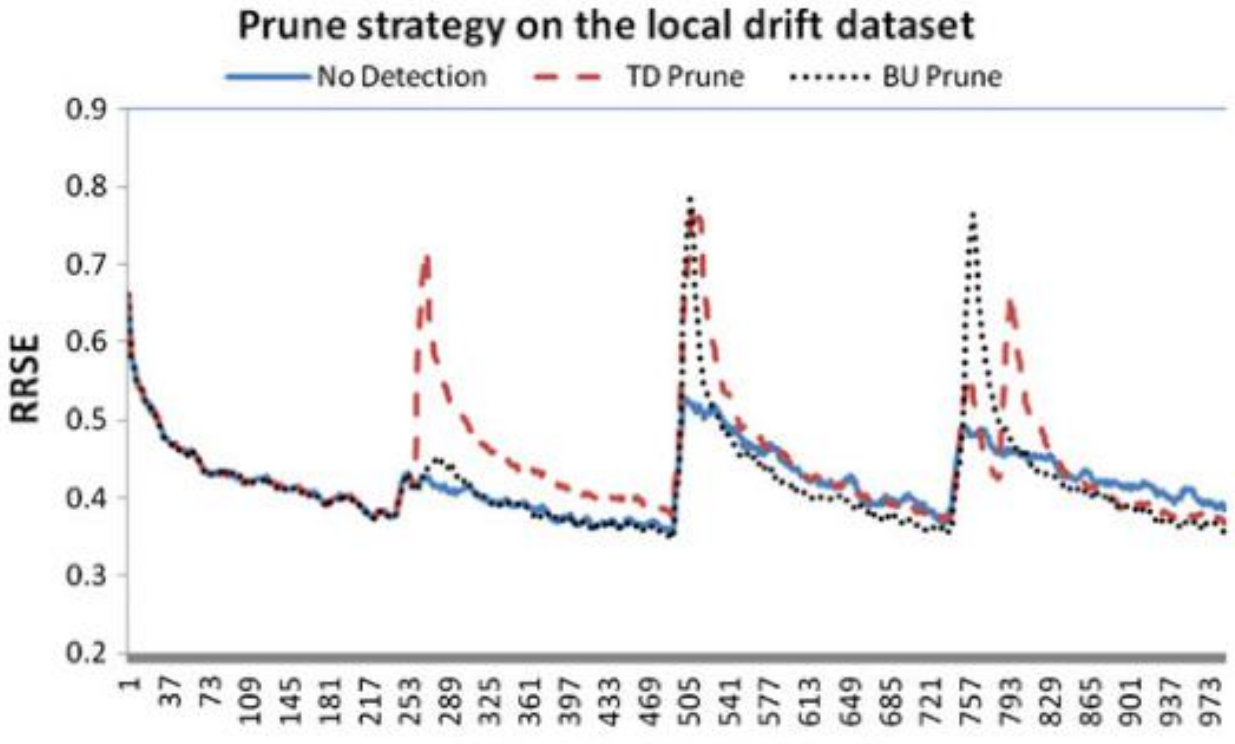
2. **Prune** the parts of the tree where concept drift was detected.

3. Build an **alternate sub-tree** for the region where drift is detected.

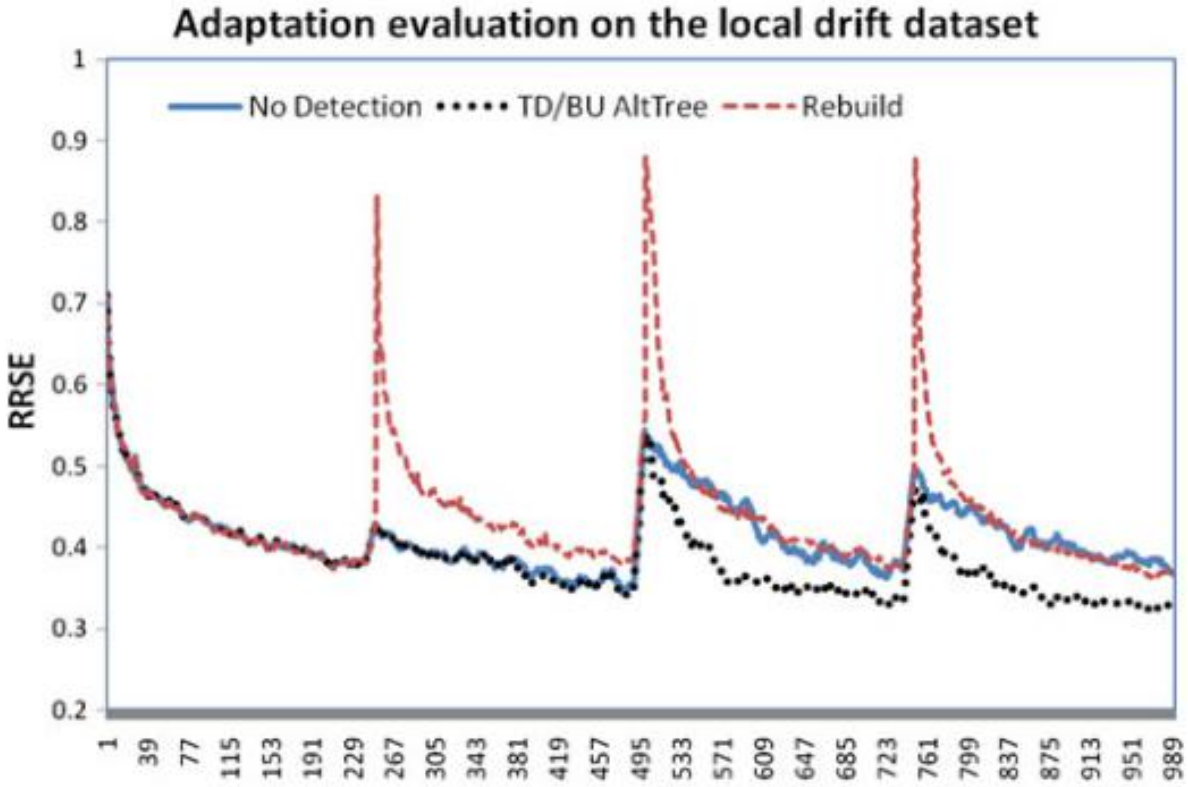
Experiment

Algorithm	RE%	RRSE%	Leaves	Time (s)
FIRT	0.16	0.19	2452.23	24.75
CUBIST	0.13	–	37.50	104.79
FIMT_Const	0.11	0.14	2452.23	27.11
FIMT_Decay	0.11	0.14	2452.23	26.93
LR	0.46	0.51	1.00	2468.61
BatchRD	0.08	0.10	27286.30	5234.85
BatchRA	0.71	0.69	56.97	2316.03
OnlineRD	0.10	0.13	6579.50	3099.82
OnlineRA	0.70	0.68	57.77	2360.56

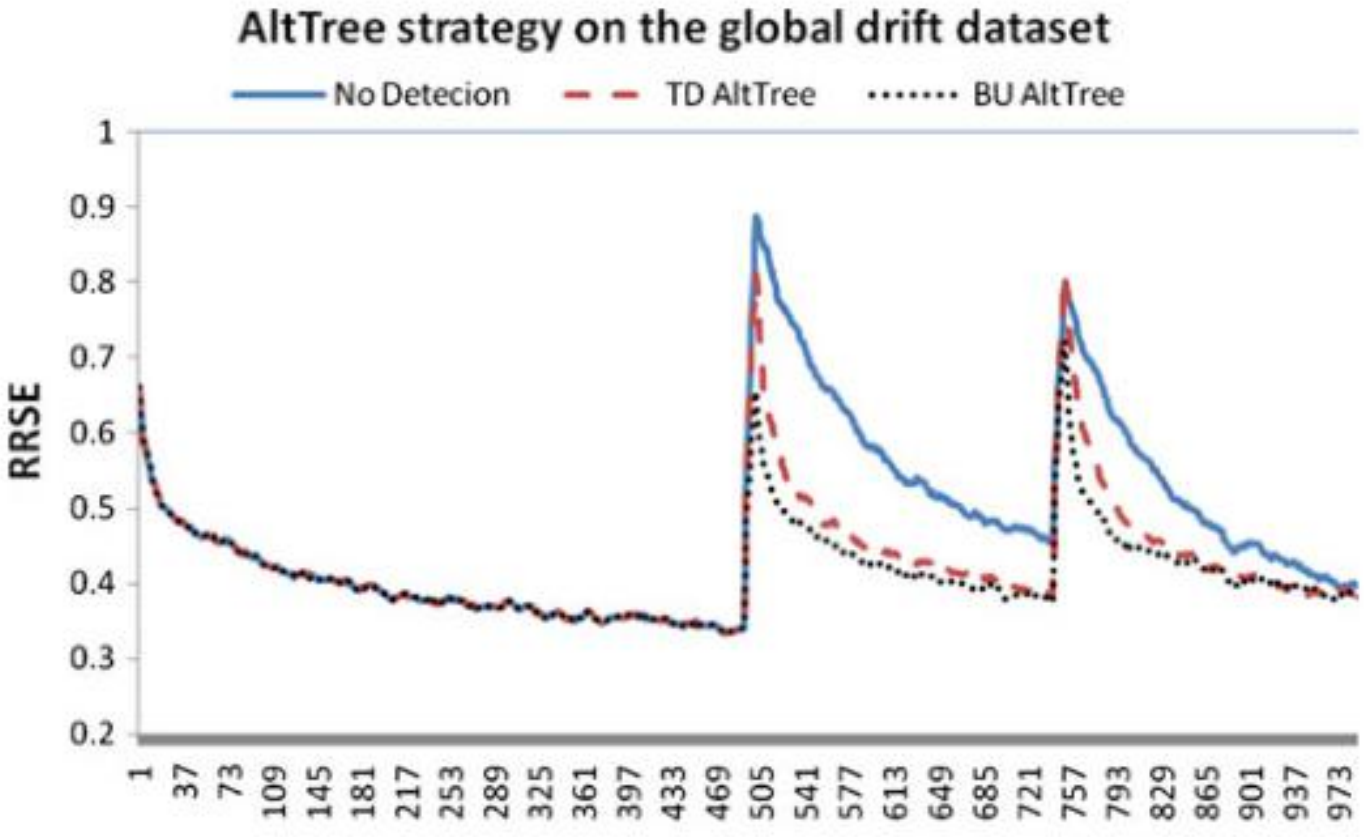
Local drift dataset: three points of abrupt change in the training dataset, the first one at 1/4 of the examples, the second one at 1/2 of the examples and the third at 3/4 of the examples




AltTree strategy on the local drift dataset
No Detection
TD AltTree
BU AltTree
RRSE
0.7
0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
1
38
75
112
149
186
223
260
297
334
371
408
445
482
519
556
593
630
667
704
741
778
815
852
889
926
963


Adaptation evaluation on the local drift dataset
No Detection
TD/BU AltTree
Rebuild
RRSE
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
1
39
77
115
153
191
229
267
305
343
381
419
457
495
533
571
609
647
685
723
761
799
837
875
913
951
989

Global drift dataset: There are two points of concept drift, the first of which occurs at 1/2 of the examples and the second at 3/4 of the examples



Conclusion

- Using tree structure to deal with incremental learning really is a good choice.

- Using another regression method to deal with concept drifting is an urgent job such as GBM(Gradient Boost Machines)

Thanks