Speeding up splits in Hoeffding Tree Regressors

Saulo Martiello Mastelini[1]

[1] Institute of Mathematics and Computer Sciences, University of São Paulo, Brazil.
MASTELINI@USP.BR / SAULOMASTELINI@GMAIL.COM

March 31, 2021

Schedule

1. Regression trees
   - Incremental regression trees
2. The (incremental) variance problem
   - Naive algorithm
   - Welford's algorithm
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

Disclaimer

This presentation is grounded on the following paper:

- Mastelini, S.M. and de Leon Ferreira, A.C.P., 2021. Using dynamical quantization to perform split attempts in online tree regressors. *Pattern Recognition Letters*, 145, pp.37-42.

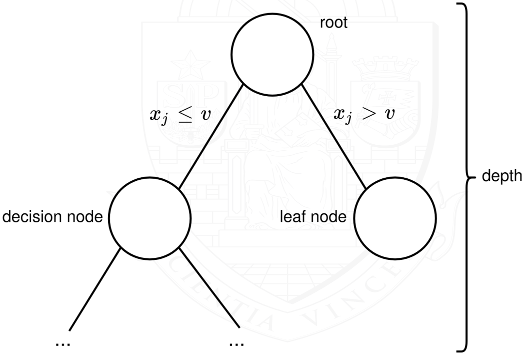
Schedule

1 Regression trees
2 The (incremental) variance problem
3 Extended Binary Search Tree Observer
4 Quantization Observer
5 Benchmarking
6 Final Remarks

Introduction

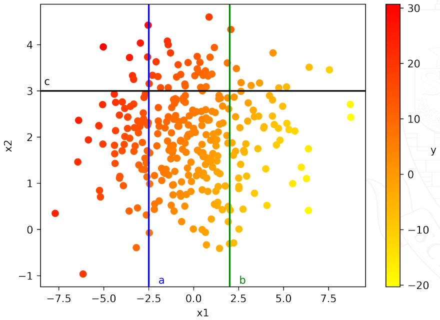
Context:

- Numerical input features
- Axis-aligned splits
  - $x_j \leq v$ (left branch)
  - $x_j > v$ (right branch)



Decision splits

- We need partitions that make the sub-spaces maximally homogeneous



- How to pick the best threshold?

Variance reduction

- Regression trees usually aim at reducing the variance within the created partitions
- Variance Reduction heuristic

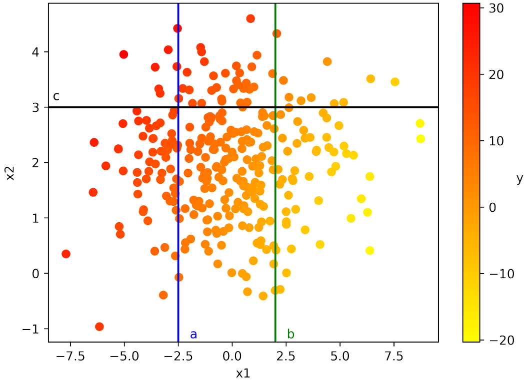
$$\text{VR}(y, x, \Theta) = Var(y) - \frac{|y_{x \leq \Theta}|}{|y|} Var(y_{x \leq \Theta}) - \frac{|y_{x > \Theta}|}{|y|} Var(y_{x > \Theta})$$

- Best split candidate

$$(x_*, \Theta_*) = argmax_{(x_i, v), x_i \in \{x_1, .., x_m\}, v \in \mathbb{R}} \text{VR}(y, x_i, v)$$

- Equivalent to minimizing the Mean Squared Error (MSE)
  - The created partitions are maximally compact

Variance Reduction



- $VR(x_1, a) = 85.445 - (\frac{65}{300}) \times 22.895 - (\frac{235}{300} \times 59.75) = 33.683$
- $VR(x_1, b) = 35.570$
- $VR(x_2, c) = 7.811$

We are all set to build regression trees!

Incremental regression trees: the needed tools

1. How to assure that our split candidates are indeed the best ones?
   - Hoeffding Bound ✓
2. How to evaluate split candidates?

   2.1 We need to calculate the elements of the VR equation
   - **Incremental variance calculation!**

   2.2 For any given partition (x_i, v): $y_{x_i \leq v}$ and $y_{x_i > v}$
   - **How to do that incrementally and with reduced memory footprint?** (and running time)
   - Answer: attribute observer (AO) algorithms (a.k.a. splitters)

Schedule

1. Regression trees
2. The (incremental) variance problem
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

The naive approach

- Keep:
  - n: number of observations
  - $\sum y$: sum of y values
  - $\sum y^2$: sum of the squared y values
- $Var = \frac{1}{n-1}\left(\sum y^2 - \frac{1}{n}\left(\sum x\right)^2\right)$
- Used in Fast Incremental Model Tree with Drift Detection[1] (FIMT-DD)

[1] Ikonomovska, E., Gama, J. and Džeroski, S., 2011. Learning model trees from evolving data streams. *Data mining and knowledge discovery*, 23(1), pp.128-168.

Naive approach: the cool part

- Imagine that we keep two variance estimators:
  - Total variance of y: $var_T(y)$
  - Variance of y for elements that satisfy $x_i \leq v$: $var_l(y)$ (left tree branch)
- How do we get the complement, $var_r(y)$? (right tree branch)
  - $n_r = n_T - n_l$
  - $\sum y_r = \sum y_T - \sum y_l$
  - $\sum y_r^2 = \sum y_T^2 - \sum y_l^2$
- We do not need to keep var_r!
  - Memory savings to the attribute observers

Why naive?

- Both $\sum y^2$ and $\sum y$ can become really big
- Numerical cancellation
- Sometimes can even yield negative variance values (??)
- Text books do not advice to use this estimator in real-world applications[1]

[1] Knuth, D.E., 2014. Art of computer programming, volume 2: Seminumerical algorithms. Addison-Wesley Professional.

Welford's algorithm: a stable solution

Initialize: $\overline{x}_1 = 0$, $M_{2,1} = 0$. For any $n > 1$:

- $\overline{x}_n = \overline{x}_{n-1} + \dfrac{x_n - \overline{x}_{n-1}}{n}$
- $M_{2,n} = M_{2,n-1} + (x_n - \overline{x}_{n-1})(x_n - \overline{x}_n)$
- Variance: $\frac{M_{2,n}}{n-1}$

In the next slide we drop the *n* indexing, for simplicity

Handling partial statistics

Handling addition[1] and subtraction[2]

Addition:

- $n_{AB} = n_A + n_B$
- $\overline{x}_{AB} = \dfrac{n_A\overline{x}_A + n_B\overline{x}_B}{n_{AB}}$
- $M_{2,AB} = M_{2,A} + M_{2,B} + \delta^2 \dfrac{n_A n_B}{n_{AB}}$
- In the expressions above, $\delta = \overline{x}_B - \overline{x}_A$

Subtraction:

- $n_A = n_{AB} - n_B$
- $\overline{x}_A = \dfrac{n_{AB}\overline{x}_{AB} - n_B\overline{x}_B}{n_A}$
- $M_{2,A} = M_{2,AB} - M_{2,B} - \delta^2 \dfrac{n_A n_B}{n_{AB}}$

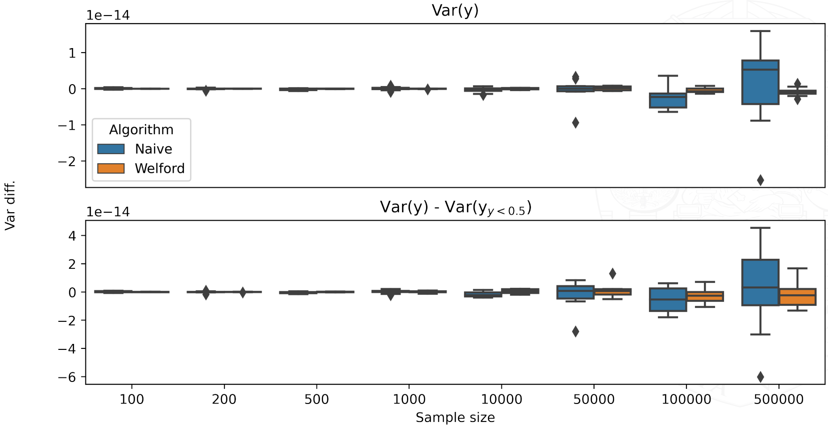
[1] Chan, T.F., Golub, G.H. and LeVeque, R.J., 1982. Updating formulae and a pairwise algorithm for computing sample variances. In COMPSTAT 1982 5th Symposium held at Toulouse 1982 (pp. 30-41). Physica, Heidelberg.

[2] Mastelini, S. M., de Leon Ferreira, A. C. P., 2021. Using dynamical quantization to perform split attempts in online tree regressors. Pattern Recognition Letters, 145, (pp. 37-42).

Benchmarking the variance estimators

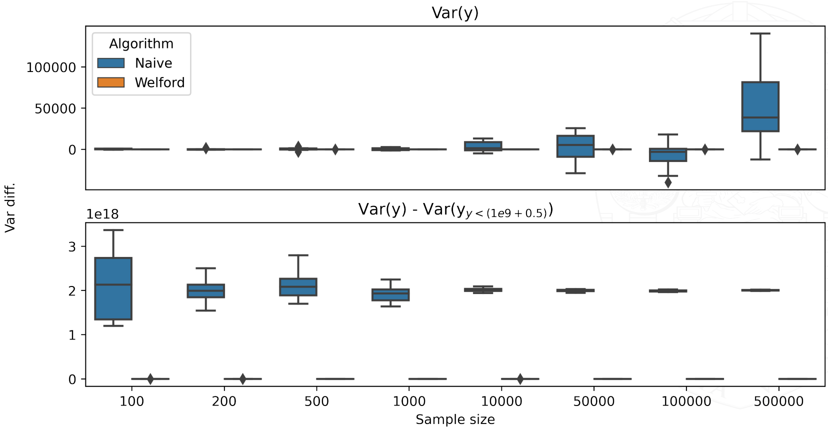
- Naive and Welford against the non-incremental variance estimator
- Increasing sample size
- Difference between the obtained variances
  - Ground truth: non-incremental variance

Benchmarks: uniform data between (0, 1)



- So far, so good

Benchmarks: adding constant shift of 10^9

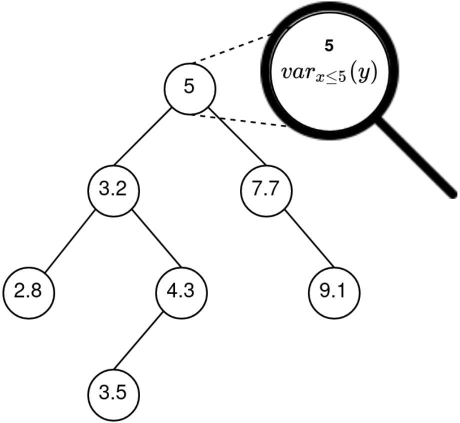


- Hence, the Welford's algorithm will be our preferred choice

Schedule

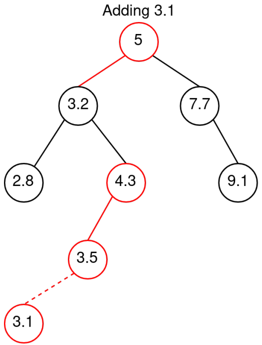
1. Regression trees
2. The (incremental) variance problem
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

E-BST: using trees to build trees



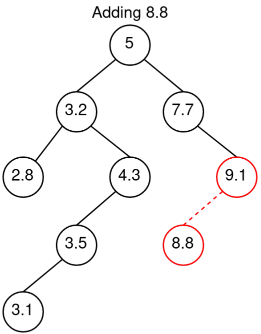
- Keep numerical features in a binary search tree:
  - Each node carries the feature value and a var estimator for *y*
- BST is not balanced

E-BST: adding a new node (I)



- The variance estimates are updated as new values are sorted down the BST
- Only statistics of the left branches are updated as new values are inserted into the BST
  - Partial statistics are kept
- The updated nodes are in red

E-BST: adding a new node (II)



The complete statistics are retrieved by performing a complete in-order traversal:

1. Create an auxiliary variance estimator var_{aux}
2. If traversing to:
   2.1 **left** branch: pass var_{aux} without modification
   2.2 **right** branch: update var_{aux} with the current node's statistics before descending
3. The complete statistics (test $\leq$) are given by aggregating var_{aux} and the current node's variance estimator
4. Undo changes to var_{aux} when backtracking

E-BST: cost and variant

Insertion	$O(\log n)$ or $O(n)^*$
Memory	$O(n)$
Query time	$O(n)$

* the worst case, when incoming instances are ordered

Some alternatives to alleviate these costs:

- Limit the number of nodes **x**
- Round the incoming data before insertion (Truncated E-BST – TE-BST) ✓
- From time to time, remove bad split candidates from the BST[1] ✓

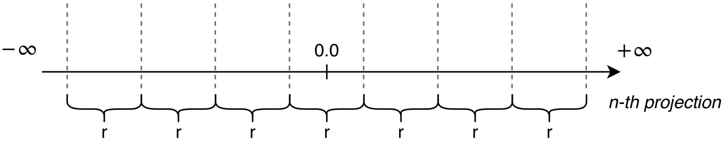
[1] Ikonomovska, E., Gama, J. and Džeroski, S., 2011. Learning model trees from evolving data streams. *Data mining and knowledge discovery*, 23(1), pp.128-168.

Schedule

1. Regression trees
2. The (incremental) variance problem
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

QO: simple yet effective solution

- Can we get rid of the logarithmic cost per insertion?
  - What if we reached a cost of $O(1)$ per insertion? Answer: Hashing!
- Inspiration in Locality Sensitive Hashing (LSH)[1]:
  - Instead of mapping each element to its own hash slot, map similar elements to the same slot
- Straightforward projection rule: $h = \left\lfloor \frac{x_i}{r} \right\rfloor$, where r is the quantization radius (hyperparameter)



[1] Datar, M., Immorlica, N., Indyk, P. and Mirrokni, V.S., 2004, June. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the twentieth annual symposium on Computational geometry (pp. 253-262).

QO: example

Let's assume $r = 0.25$ and an empty hash table H

- Insertion points: 2.3, 3.1, 7.78, 7.8
  - $h_{2.3} = \left\lfloor \frac{2.3}{0.25} \right\rfloor = \lfloor 9.2 \rfloor = 9$
  - $h_{3.1} = \left\lfloor \frac{3.1}{0.25} \right\rfloor = 12$
  - $h_{7.78} = 31$
  - $h_{7.8} = 31$
- For each slot we keep the mean x value and a variance estimator for y
  - Split points: Middle point between two consecutive slots

QO: cost and limitations

Cost	E-BST	TE-BST	QO
Insertion (per instance)	$O(\log n)$ or $O(n)$*	$O(\log n')$ or $O(n')$*	$O(1)$
Memory	$O(n)$	$O(n')$	$O(\|H\|)$
Query time	$O(n)$	$O(n')$	$O(\|H\| \log \|H\|)$

* the worst case, when incoming instances are ordered

- $|H|$: number of slots in the hash
- $n' \leq n$ (depending on the rounding procedure)
- The costs of QO depend on the choice of r

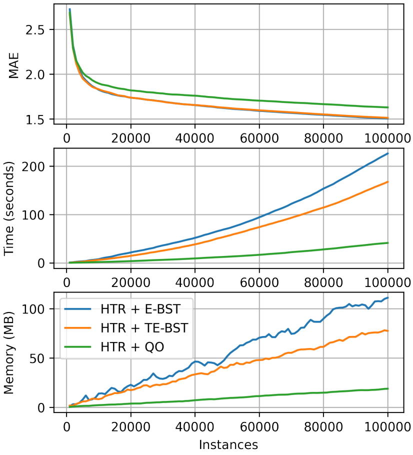
Schedule

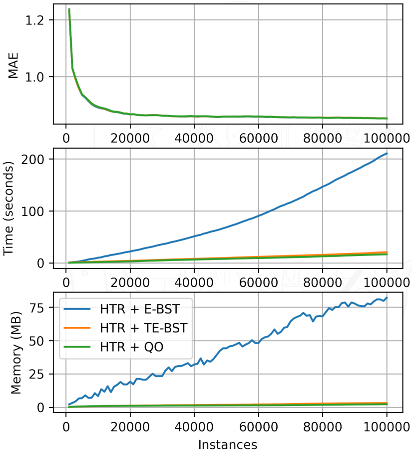
1. Regression trees
2. The (incremental) variance problem
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

Friendly reminder

- Regression trees maximize the VR when making splits
- Incremental decision trees use attribute observers (AO) to evaluate split candidates
- Each tree leaf carries one AO per input feature

100k samples: Friedman and Planes2D





Schedule

1. Regression trees
2. The (incremental) variance problem
3. Extended Binary Search Tree Observer
4. Quantization Observer
5. Benchmarking
6. Final Remarks

Summary

- We apply a **robust** and incremental **variance estimator** to Hoeffding Tree Regressors
- We proposed a **simple yet effective attribute observer** algorithm for incremental regression tree construction
- **QO** is able to deliver **faster** tree construction with **reduced memory** footprint, while keeping the **error increase minimal**
- You can check everything in `River` :-D

What next?

- Mini-batches and distributed processing
  - QO is mergeable!
  - We can take advantage of *vectorization* and distributed processing units
- Investigate in depth the impact of r (quantization radius) and alternatives to select it automatically

Acknowledgments

- FAPESP for the financial support (grant #2018/07319-6)
- My advisor, André C. P. L. F. de Carvalho for his valuable advice and guidance

Thank you so much for your attention!

Questions?

Speeding up splits in Hoeffding Tree Regressors

Saulo Martiello Mastelini[1]

[1] Institute of Mathematics and Computer Sciences, University of São Paulo, Brazil.
MASTELINI@USP.BR / SAULOMASTELINI@GMAIL.COM

March 31, 2021