In-Stream Analytics, ADAPTIVE Machine Learning & *an enduring solution architecture!*

In-Stream Analytics:

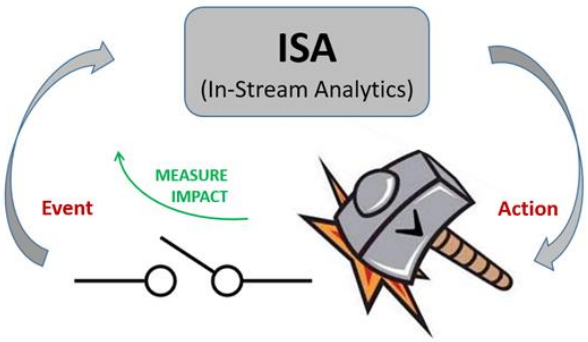
Machine Learning today tends to be "open-loop" – collect tons of data offline, process them in batches and generate insights for eventual action. There is an emerging category of ML business use cases that are called "In-Stream Analytics (ISA)". Here, the data are processed as soon as they arrive and insights are generated fast. However, action may be taken offline and *the effects of the actions are not immediately incorporated back into the learning process*. If we did, it is an example of a "closed-loop" system – **we will call this approach "Adaptive Machine Learning" or AML**.

Why is Time important?

Here is a small list of business use cases summarized from a recent blog, Stream Processing, where "time" is important - real-time applications, if you will.

1. Fraud Detection:
   - Rules and scoring based on historic customer transaction information, profiles and even technical information to detect and stop a fraudulent payment transaction.
2. Financial Markets Trading:
   - Automated high-frequency trading systems.
3. IoT and Capital Equipment Intensive Industries:
   - Optimization of heavy manufacturing equipment maintenance, power grids and traffic control systems.
4. Health and Life Sciences:
   - Predictive models monitoring vital signs to send an alert to the right set of doctors and nurses to take an action.
5. Marketing Effectiveness:
   - Detect mobile phone usage patterns to trigger individualized offers.
6. Retail Optimization
   - In-Store shopping pattern and cross sell; In-Store price checking; Creating new sales from product returns.

One factor that necessitates real-time interaction is the **closed-loop nature** of these use cases. In every case, an external event happens; Analytics module determines a recommended action and creates a response that will impact the external event in a timely manner.



What "real-time" means is case dependent. The rate at which data collection, analysis and action happen could be milliseconds, hours, days, . . . The industry name for this is Event Stream Processing or In-Stream Analytics (ISA). There are specific implementations of computer systems, databases and data flow protocols available to address the stringent requirements of ISA systems.

It appears to me that when "Analytics module determines a recommended action and creates a response that will impact the external event", *one should also measure the IMPACT of the action so that:*

(1) we know if our action was good,
(2) we can use any shortcomings to improve ISA so that the next ISA event will have a better outcome and
(3) we can attribute the right portion of the result to ISA's action.

Explicit definition of LEARNING:
If Learning is the process of "generalization from experience", we can be more explicit and say that *"**generalization from past experience AND results of new action**" is the true definition of Learning!*

With this broader view of "Learning", ALL Analytics applications conform to the picture above . . . *other than purely Descriptive Analytics.* ML algorithms generate outputs that requires action . . . in every ML use case I know, this is not done just once! The learning system may be kept "behind the curtain" but any ML application that has business impact will have to close the loop. Off-line ML system (behind the curtain) will still require a "trickle" of continuous learning lest the learned system becomes "aged" and not responsive to new changes. Even a language translation ML system belongs in the closed-loop category for proper continuous operation (new words and expressions enter the lexicon all the time!).

In short, "Adaptive" ML in a closed-loop configuration is a basic necessary feature for any ML business solution.

Exact Recursive Algorithms

Off-line methods can use batch processing but closed-loop, In-Stream Analytics will require "recursive" online processing so that each data input is processed as it arrives; Recursive algorithms were explicitly developed to do this! Nature of recursive algorithms can be confusing and is often misunderstood in the context of Machine Learning . . .

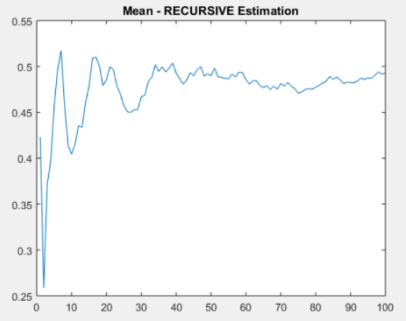
For example, you must have encountered Recursive Least Squares (RLS). It is exact in the sense that "batch" Least Squares solution using all the data at one time and the solution obtained by RLS are identical. Let us explore . . .

Consider the case of finding the mean height of students in a class. We decide to use the standard formula for Sample Mean. We measure the heights of N students and apply the estimate –

$\bar{x}_N = \frac{1}{N}\sum_{i=1}^{N} x_i$ where x_i is the height of an individual student. With simple algebraic manipulations, we can write the sample mean estimate as a "recursive" algorithm so that *Sample Mean is calculated as soon as each individual student's height is measured.*

$\bar{x}[n] = \frac{x[n]}{n} + \frac{(n-1)}{n}\bar{x}[n-1]$ with $\bar{x}[n] = 0$. Here 'n' is a "counter" variable. When n = N, the usual sample mean and the recursive sample mean are EXACTLY equal. It is in this sense that recursive

algorithms are "exact". Re-writing the **Exact Recursive Algorithm (ERA)** for sample mean to make it easy to compare and contrast with Learning algorithms,



$\bar{x}[n] = \frac{(n-1)}{n}\bar{x}[n-1] + \frac{x[n]}{n}$. Here we can think of the first term on RHS to be the "past" estimate of the sample mean and the second term, a "correction" term. Compare to the Steepest Descent learning formula for weight update -

$w[n+1] = w[n] + \Delta w[n] = w[n] + \eta\ e[n]\frac{\partial \tilde{y}[n]}{\partial w[n]}$. This has the same form as our Sample Mean ERA above; past estimate plus a correction term! There IS a difference however. ERA is in the form of a "filter" where the correction term is based on the current value and updates the current Sample Mean; where as in the steepest descent formula, the correction term is based on the current value and updates the future weight ("prediction" form).

RLS is an Exact Recursive Algorithm (ERA) which applies when the least squares problem is linear admitting exact solution at each step. In the non-linear case, we resort to Steepest Descent where **f(.)** is non-linear in the model, $\mathbf{y_i = f(x_i) + e_i}$. Then, the error surface is not smooth (quadratic) and an approximate learning method is needed where it searches the error surface to find local or global optima.

So is ERA a "learning" method? Sure. Solving the whole problem up to that step at every step is the same as *learning "as much as it can" from each new piece of information*, keeping in mind the linear constraint! It is well-known in Systems Theory and Signal Processing that many methods can be cast as an ERA because many problems are solutions of the well-known "Normal Equation" reproduced below -

$\underline{\mathbf{w}}^* = (\underline{\mathbf{x}}^T\underline{\mathbf{x}})^{-1}\underline{\mathbf{x}}^T\underline{\mathbf{y}} = \underline{\mathbf{R}}^{-1}\underline{\mathbf{r}}$, $\underline{\mathbf{R}}$ is the Autocorrelation matrix of input, $\underline{\mathbf{x}}$, and $\underline{\mathbf{r}}$ is the Cross-correlation vector between input, $\underline{\mathbf{x}}$, and desired output, y.

When $\underline{\mathbf{R}}$ matrix inversion is replaced by the celebrated Matrix Inversion Lemma, the result is an Exact Recursive Algorithm, which solves the Normal Equation exactly at each step with the information available till that step.

From the Sample Mean ERA discussion above, you would have noticed that storage requirements are drastically less for ERA since only the past average and current data is to be processed. For very large N, this could be a significant implementation factor. Memory savings is the least of the desirable properties for ERA . . .

Exact Recursive Algorithms (ERAs):
ERAs solve the estimation problem as each new data arrive - **this ability to "TRACK" is the enormous value of ERA algorithms.** In other words, if **f(.)** in our model, $\mathbf{y_i = f(x_i) + e_i}$, is time-varying, ERAs have the ability to track the changes since exact solution is calculated at each step. *In practice, the nature of variation (speed of change, complexity, etc.) will limit the ability of ERAs be fully change-adaptive.*

- ***Recursive Least Squares (RLS) is an example of exact solutions for Linear TIME-VARYING Least Squares models.***

Machine Learning has not paid much attention to "tracking" solutions. Why is that? May be because business use cases which demand closed-loop solutions may be assumed to be non-varying – in time or space or other dimensions – ***as a simplification****!* But in reality, every model evolves slowly or quickly. I have not seen a business problem that is fully satisfied with a "one and done" solution! Any serious solution is like flu-shots - adjust the mix and apply on a regular basis! ***Why do we do that? Because entities involved vary (over time, in this case) and "tracking" will improve the results and may even be necessary on an on-going basis if changes are significant.***

ERAs adapting to change can be categorized as leading to a model that -

1. Changes to a "new normal".
2. Changes to "abnormal".

From a steady-state, ML algorithm learns what represents "normal" (think of an IoT example where a piece of machinery is operating fine); when deviations occur, ERAs will quickly learn the new model. There are 2 ways to flag this change:

a) Monitor ERA model parameters for change.
b) Monitor the output (such as prediction error) of a non-tracking algorithm; the change in the system will cause a larger than normal error in the monitored output.

As you can see, BOTH will flag an event but **ERA is more informative** – it characterizes the change (in terms of new model parameters) which is valuable to know: Is it a -
(1) change to a "new normal" or
(2) change to "abnormal"?

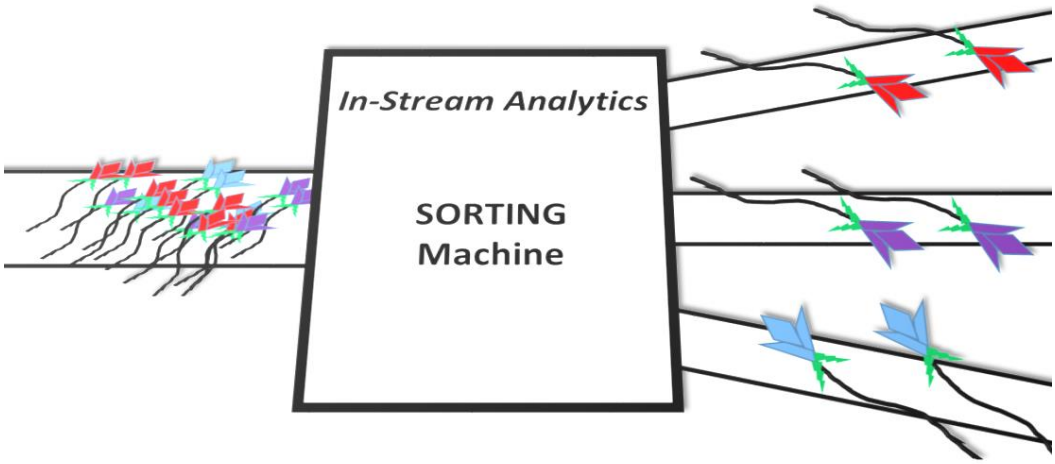
The response of the human operator to (1) and (2) will be dramatically different! Plus accumulated information about the new "normals" add to the knowledge base which will allow more advanced versions of the ML solution. This learning at a higher level enhances the power of ML solutions as time goes on – we call this "Adaptive" ML solutions. **Thus Exact Recursive Algorithms (ERAs) leads to Adaptive ML.**

AML methods are necessary for closed-loop solutions. Off-line or batch methods are suitable for investigations and explorations but solution to any business problem will require it to be closed-loop with the time available between event and action varying from milliseconds to hours, days and months. *The adaptive nature of AML, where the solution adapts to the changing environment, renders the solution fully automatic removing the need for human intervention in the best case.*

Practical example of AML using ERA:

To consolidate the concept of Adaptive ML, let us consider a practical (but "*made up*") ML problem related to Fisher Iris data.

Case Study of 'Fisher Iris Farm' and Automated Flower-sorting Systems:
The figure below is a rough caricature of their industrialized flower sorting system. Big heaps for Iris flowers of 3 types are dropped off by the farm equipment on the left side. The automatic sorting system has to classify each flower as belonging to SETOSA, VIRGINICA or VERSICOLOR and send them on to 3 separate conveyer belts so that the 3 types of Iris can be packaged separately.



We design an optical system that can measure 4 attributes, Sepal length, Sepal width, Petal length and Petal width automatically while the flowers pass through the sorting machine.

Step 1: Offline classifier development

- We collect 150 sets of {Attributes, Iris Type}.
- Using 90, we train a classifier using RLS.
- We test the classifier using 60 "held-out" pairs and find that the classification accuracy is acceptable for this industrial application.

Step 2: Online operation

- Optical system in the Sorting machine automatically measures the Sepal length, Sepal width and Petal length of the passing flower.
- This data in passed on to the In-Stream Analytics system that uses the classifier trained in Step 1 above.
- An arm mechanism directs the right flower onto the right conveyer belt for that flower.

As we know, the classifiers are not perfect and there will be misclassification every now and then. For business reasons (wrong flower in the wrong box generating customer complaints), the Data Scientist is asked to improve the performance.

Step 3: Performance improvement using ERA

- Quality Control department inspects the operations for 30 minutes during the day at random times.
- When a flower is on the wrong conveyer belt, a switch is pressed that informs ISA automatically of the wrong classification and ISA extracts the Attributes for that flower from the logs.
- This information is then used by the Exact Recursive Algorithm to update the classifier online.

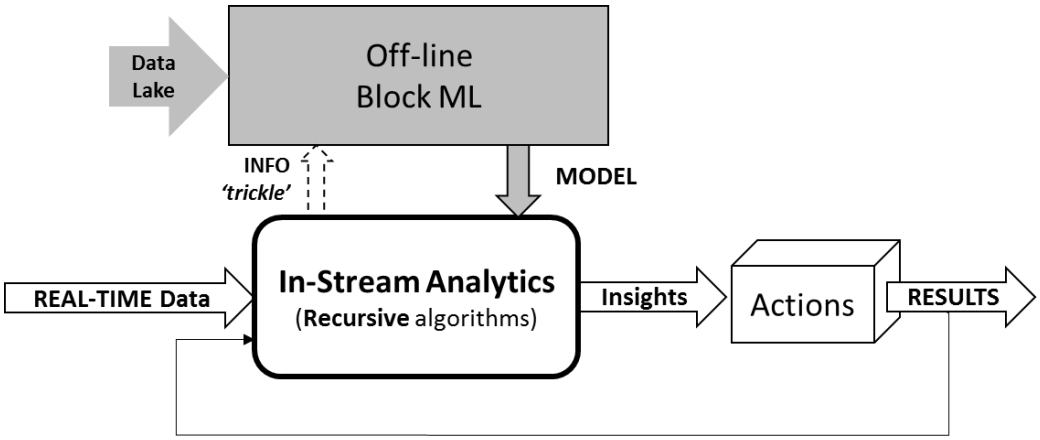
Offline Training → Operation → Closed-loop feedback → Recursive Online Update of the Classifier.

If there was a continuous automatic quality control system, closed-loop feedback and recursive online update can be done continuously with the classifier getting better all the time! In addition, suppose that attributes changed due to a particularly dry Spring or because of a new fertilizer, the recursive online update will "track" these changes, learn online and keep up the performance of the classifier.

Clearly, there are other applications of ML where insights are generated and resulting report is submitted to the business executive - here the loop is not closed online. But you can be sure that as ML use by the business matures, the executive will want the insights to be applied to the business problem and -

(1) know if the action produced good results and
(2) if so, what portion of the good result can be attributed to ISA.

Therefore, the need to close the loop is unavoidable for any sustaining ML business application! Here is the conceptual architecture for a PRACTICAL Adaptive Machine Learning system:



The main operating path process data in real-time, the interval being business-case dependent. Before In-Stream Analytics become operational, prior work is done in developing a "base solution" for the ML problem. From the mountain (or lake!) of data available in the archives, "block" solutions can be trained and the ML solution with the best generalization becomes the candidate for ISA. As ISA becomes operational, TWO online learning opportunities arise:

1. An exact recursive algorithm uses the Results to update itself online.
2. ISA information is "trickled" back into the offline learner which can use the new information in an opportunistic manner at some less frequent interval to update the "base" solution.

Such an Adaptive ML system is bootstrapped into existence with past data and kept current using real-time data, thus providing ***an "enduring" solution for your business case.***

About the Author:

Dr. PG Madhavan is the Founder of Syzen Analytics, Inc. He developed his expertise in Analytics as an EECS Professor, Computational Neuroscience researcher, Bell Labs MTS, Microsoft Architect and startup CEO. PG has been involved in four startups with two as Founder.
More at www.linkedin.com/in/pgmad