A Comparison of Online learning Naïve Bayes Classifier on RSS Feeds using SPARK

Authors:
Ahmet Anil Pala & Franziska Adler
pala@campus.tu-berlin.de adler.franziska@gmx.de

TU Berlin
Advanced Information Management III
Scalable Data Analytics and Data Mining

Motivation & Problem Statement

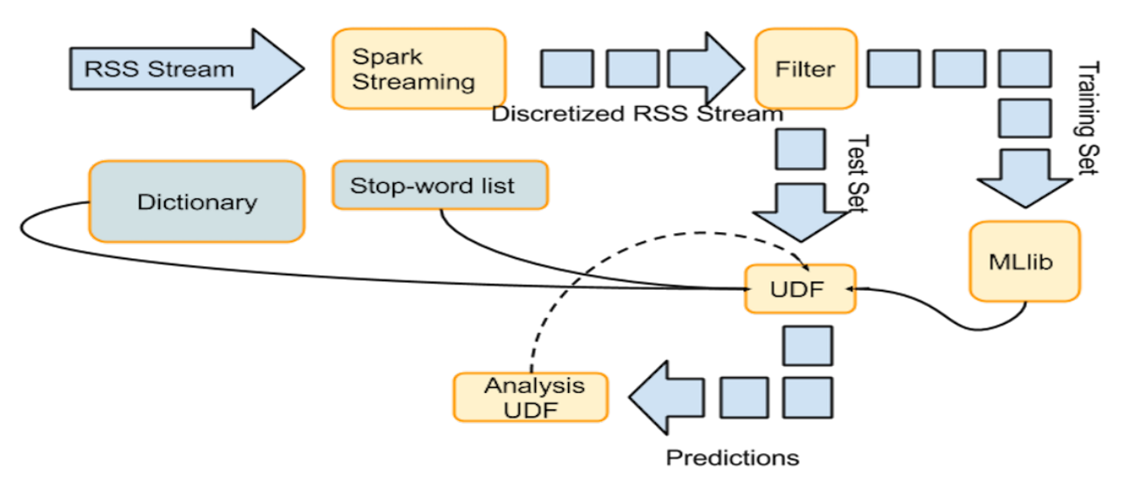
Many practical applications rely on immediate data. The need for solutions to continuously conduct analyses as mathematical computations on large data amounts led to new technologies in the past years. Stream processing operates on real-time data for example through stream windowing and analytical operations within those. Since data distributions might not remain static over time a precomputed (batch) model for analysis can produce poorly results after some time [1]. The reconstruction or updating of the underlying analytical model to receive accurate results is required. This problem is known as Concept Drift. In this project we are using unstructured streaming data from BBC RSS feeds in order to classify them to their corresponding category. We focus on the evaluation of a Naïve Bayes Classifier with different model-update approaches to compare their analytical performance in dealing with conceptual drifts.

Data & Setup

For our evaluation purposes of classifying streamed text data we are using RSS feeds from BCC. An RSS feed is a collection of tags in xml structure which contains tags for title, a description and an url among other tags. Only those listed are used in our application. Our data points are constructed from title and description of the feed, the URL gives us the respective label. Furthermore we are using Apache Spark for parallel execution of streaming and data classification. The machine learning library of Spark, MLlib, provides us with a Naive Bayes Classifier.

Streaming

In order to deal with the continuous nature of the data, traditional programming primitives are not of much help. In order to make programmers's job easier, libraries providing higher lever abstractions are introduced. Knowing that, for a news feed classifier we need to split the 'main' stream into multiple streams to treat different portions of stream differently. For example, there needs to be at least two channels of streams branching out from the main one having the test items and training items. The main flow of the streaming logic we used is depicted in the figure.[2]

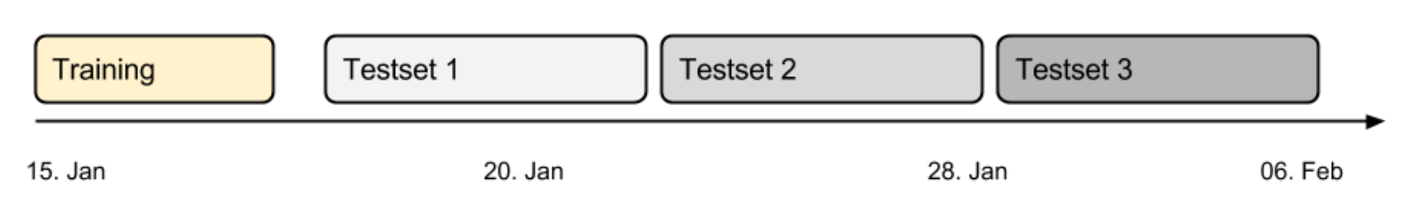


Methodology

Batch

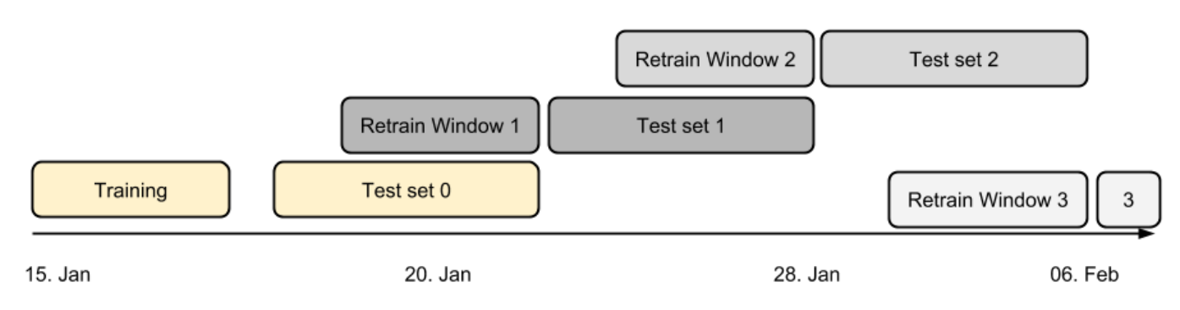
The batch/offline model implements a traditional Traing-Testing division setup. A subset training points are pre-collected from a stream and used to build a final model on which three test sets are applied. The batch model functions as a reference model to observe the performance of the initial training over time.

For training 600 data points used are from the first three days. The following four days with 1248 data points are used as the first test set for this model. A second and a third test set includes the data points comes after the previous test phase and contained 1102 and 1172 elements respectively.



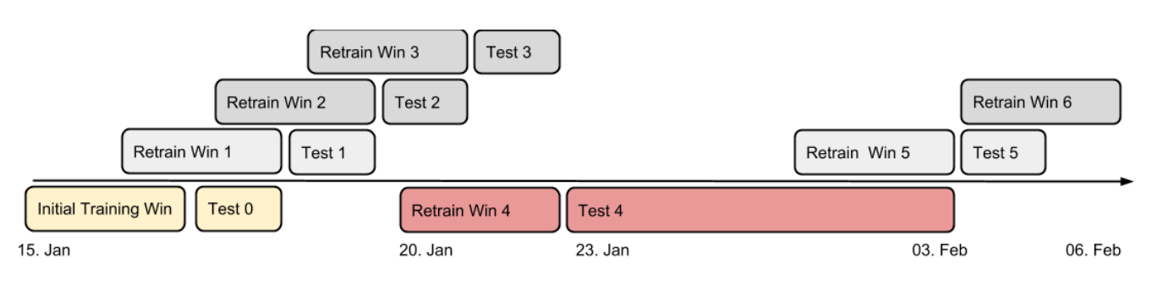
Bruteforce

On the other hand On-line Learning will change the model with the arriving of new data points. The bruteforce approach updates the model after a period of time through retraining. Based on a sliding window over the stream with a constant number of data points the model is rebuilt. The window size for the bruteforce model contains as well 600 data points. After 1000 test points we retrained it with the last 600 data points arrived. We then waited for 1000 data points and second retrain phase with the same window size followed which was tested also with 1000 new data points.



Threshold-triggered

As a variation of the bruteforce model-update, the threshold triggered one will rebuild the model on a sliding window as soon as the accuracy of the model goes below a certain threshold. The window size of the error-triggered model consists of 600 datapoint like in the previous models. We set the accuracy threshold at 63% which seemed to be a reasonable choice. Sanity window of test points ensures that at least 300 new data points arrive before prematurely throwing out the rebuilt model.

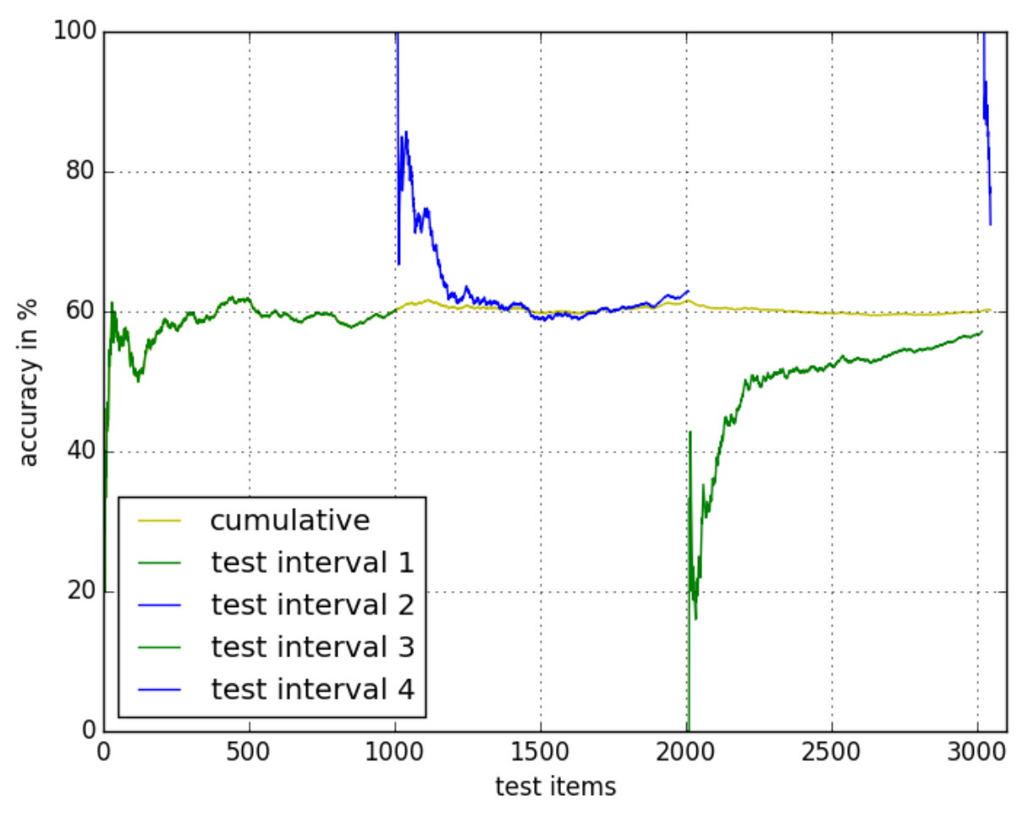


Incremental

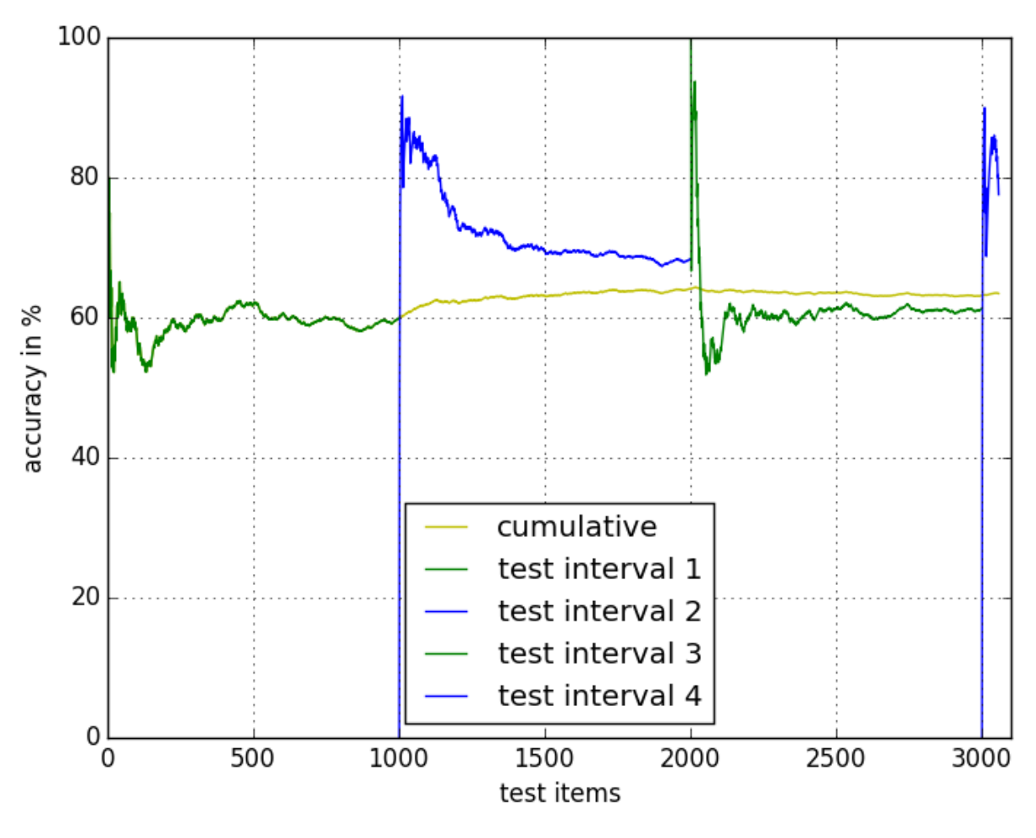
In the incremental model, unlike the other update models, following an initial training-only phase, all the incoming stream items are always used for training right after the prediction. So, this way the immediate information whether the prediction was good or bad can be used for improving the model on the fly. In other words, predictive model building and testing phases perfectly overlap in incremental model update method. We used this to achieve a real-time response to the changes in the data arguably making the text mining application more resilient to the conceptual drifts. The initial trading size of 600 is used for this model update method as well.

Results

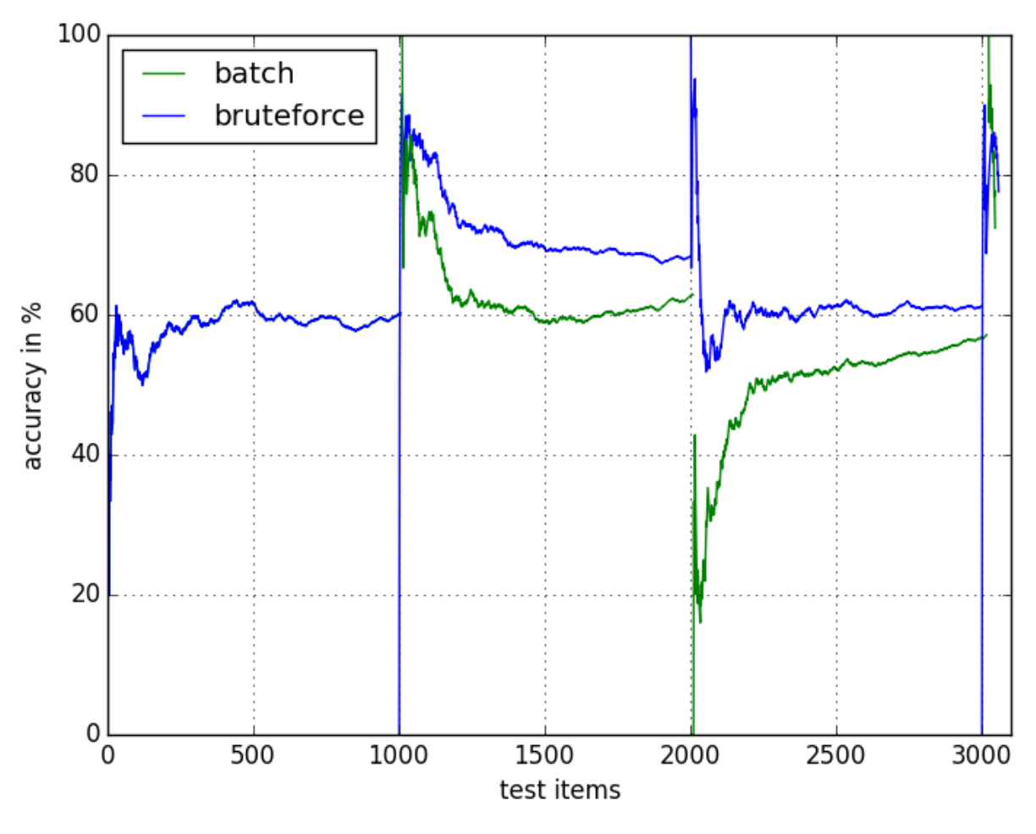
To evaluate the performance of our classification approaches we used the cumulative accuracy per interval and the cumulative accuracy over all points which. The batch/ offline method performs similar the first two test phase with but seams decreasing in the last interval.



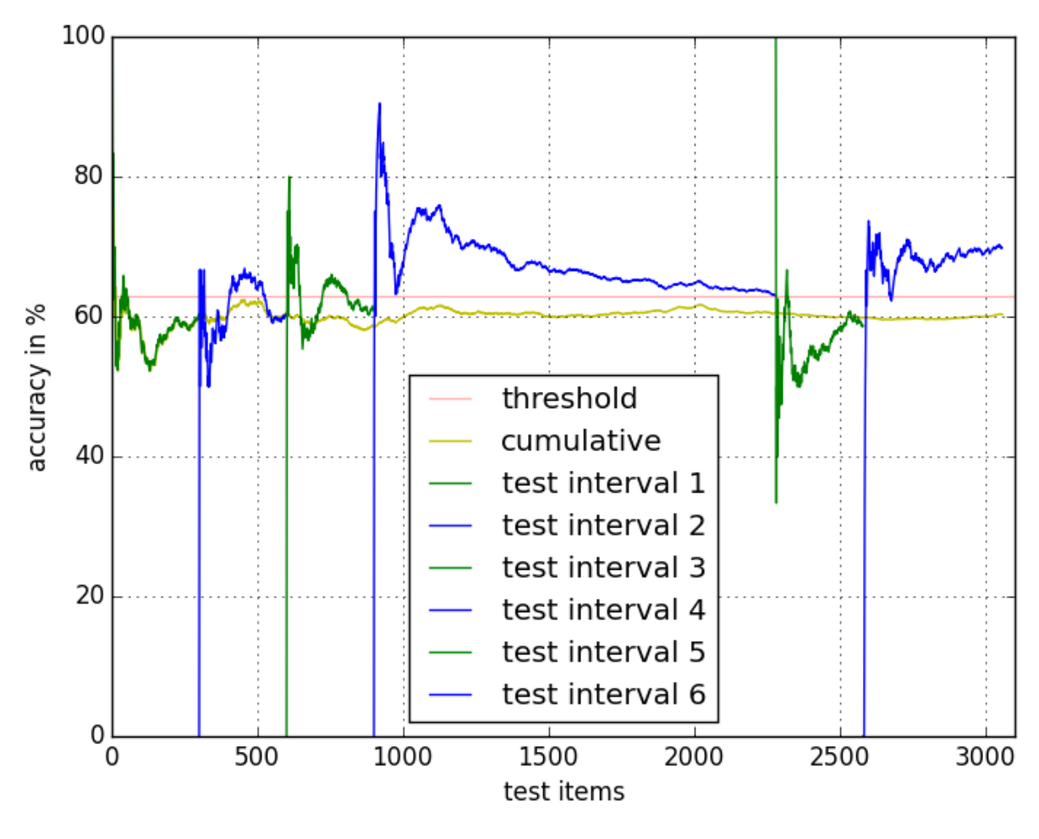
In the second scenario we applied the bruteforce model which rebuilt the model in the beginning of every interval.



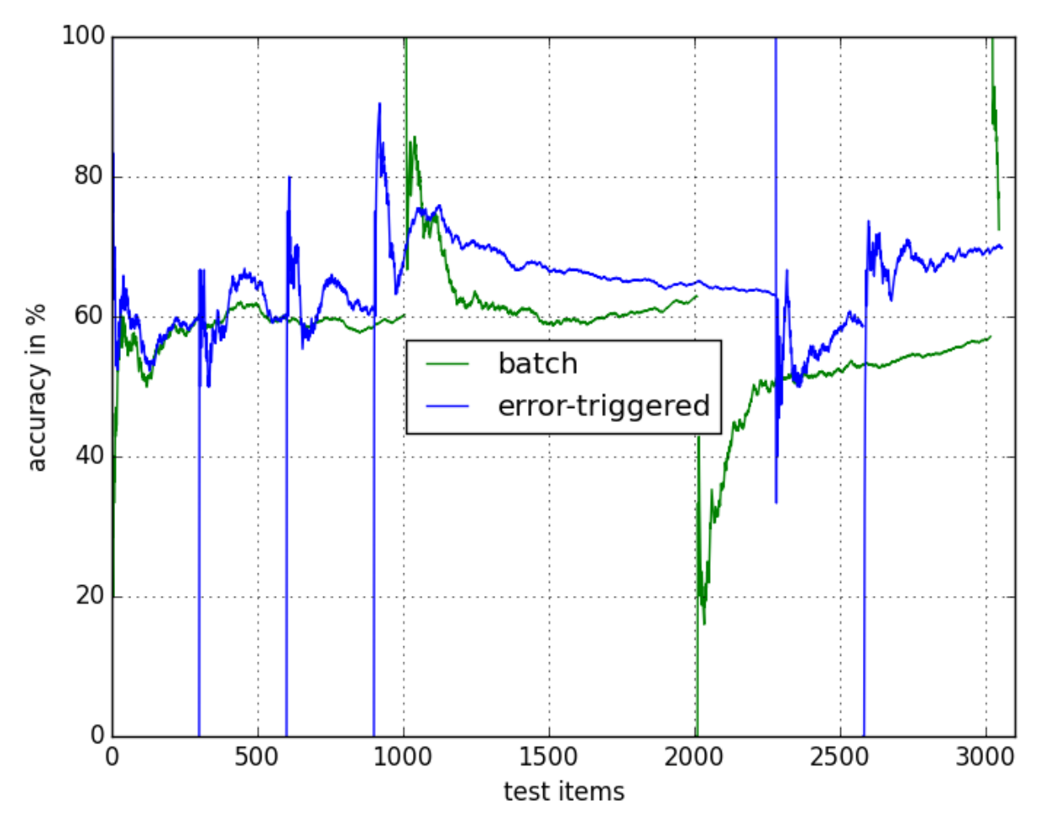
In comparison with the batch version the bruteforce model gets higher accuracies in the intervals.



For the error-triggered model we set a threshold at 63% of accuracy at which the model is recreated. After each model recreation, a sanity interval of 300 items takes place. In our experiments, we saw that the accuracy did not go below for a long time delaying the retraining period. Then two more retraining phases follow.

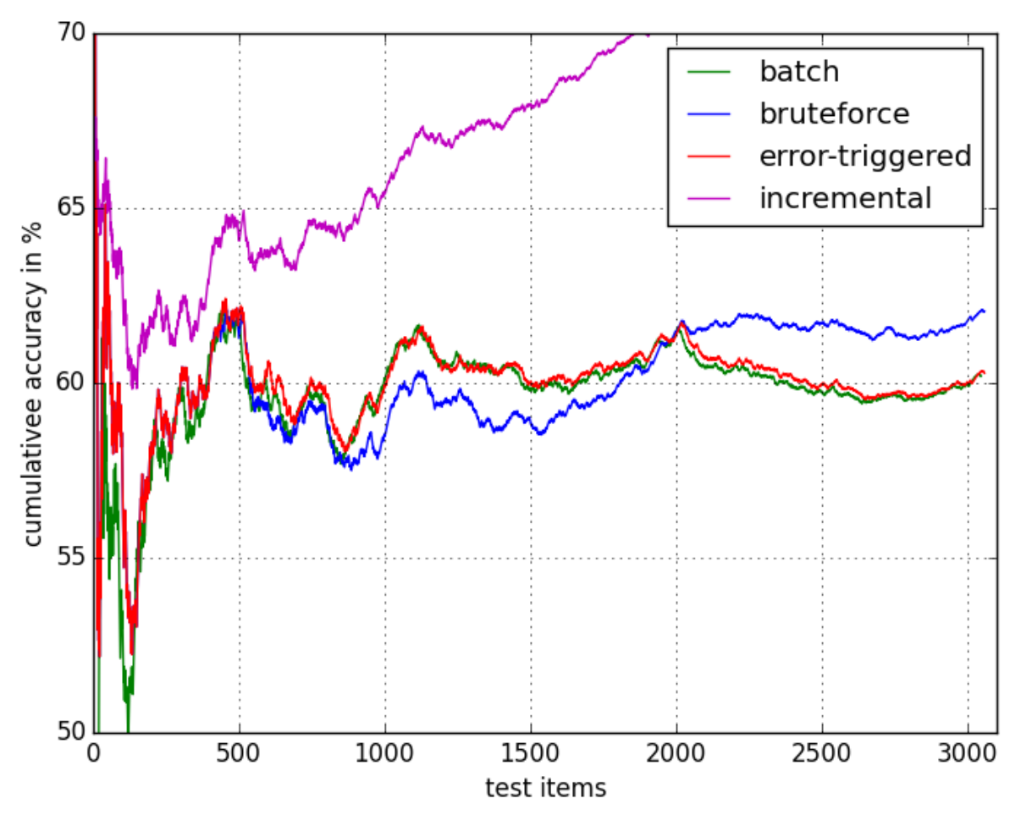


In comparison to the batch update model, error-triggered has higher interval accuracy but due to the short retraining intervals, their accuracy results are similar on average.



Bruteforce and error-triggered methods have both higher interval accuracies in the test phases around the data points of 1000 and 2000. This might be caused by concept drift.

The incremental one clearly outperformed all the other model-update methods although the learning rate was mostly between 0.02 and 0 except for three peak times when it is around 0.6.



Literatur

[1] César A. Astudillo and Javier I. Gonzalez*Concept Drift Detection Using Online Bayesian Classifier*,Proceedings of the XXXII International Conference of The Chilean Computer Science Society, 2013.

[2] Zaharia, Matei and Das, Tathagata and Li, Haoyuan and Shenker, Scott and Stoica, Ion, *Discretized Streams: An Efficient and Fault-tolerant Model for Stream Processing on Large Clusters*, Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing, 2012.