INFO-H-515 : Scalable analytics
Project 2019-20

Gianluca Bontempi, Jacopo De Stefani
Computer Science Department, ULB

The project counts for 10 points (50% of your grade). This project can be completed in groups of **maximum** 2 persons. It shall be completed independently and it shall represent the sole efforts of the individual/group submitting the assignment. The result of another student's efforts, or the copy of another student's efforts (current, or past, semester(s)), is considered academic dishonesty and will be punished accordingly.

1 Goal

The goal of the assignment is to design a scalable distributed online forecasting system that extends the `RLS_ML_Streaming` notebooks.

The `RLS_ML_Streaming` notebooks provide a set of examples for a distributed online prediction system that relies on linear (e.g. RLS) and nonlinear models. The notebook `KafkaTimeSeriesProducer` allows to send a stream of data where one output variable y is a noisy linear combination of a set of inputs x ($y = x^T\beta + w$), and the notebook `SparkStreamingRLSMLConsumer` allows to receive the data stream, and run concurrently a number of different models (e.g. linear models with different forgetting factors).

The goal of the project if to extend these notebooks to address a time series forecasting problem related to the following data generating process :

$$y(t+1) = -0.4 \cdot \frac{(3 - y(t)^2)}{(1 + y(t)^2)} + 0.6 \cdot \frac{(3 - (y(t-1) - 0.5)^3)}{(1 + (y(t-1) - 0.5)^3)}$$

In order to perform scalability tests, you could use the aforementioned data generating process to generate time series of different lengths (i.e. $y(t), t = 1, \ldots, T$ for $T \in \{10^3, 10^4, 10^5, \ldots\}$). In order to ensure replicability of your experiments, you will need to set the seed for the random generators to the value 2452020515.

2 Forecasting model

A common way to implement a forecasting model is to model the dependency between the past and the future values of the series in an auto-regressive manner

$$y(t+1) = f(y(t), \ldots, y(t-n+1))$$

where $n > 0$ is the embedding order and f is a function estimated on the basis of historical data.

Note that this setting is equivalent to a supervised setting where $y(t)$ denote the output variable and $x = [y(t), \ldots, y(t-n+1)]$ the input vector.

3 Implementation

The student should :

— implement a data production script that posts sequentially messages containing the values of the series to a Kafka topic. The student is free to choose the frequency and the size of the messages.

— implement a data receiving and processing script which returns a one-step-ahead prediction of the time series according to the following procedure :

1. Implement a persistence model which returns $y(t)$ as predictor of $y(t+1)$ (1 point)
2. Implement a weighted persistence model which returns as predictor of $y(t+1)$ the average of the last V values for $V = 2, 3, 4$ (1 point)
3. Implement four linear models of order $n = 1$, $n = 2$, $n = 3$ and $n = 4$, respectively, which analyse concurrently the dataset. The linear fitting should use the Recursive Least Squares (forgetting factor set to 1) introduced during the lectures and presented during the practicals. (4 points)
4. Choose the most accurate model (in terms of MSE) over the interval $t = 500, \ldots, 1000$ and implement three variants of this model with forgetting factors 0.99, 0.95 and 0.9 respectively. Again the three models should run concurrently. (3 points)
5. (bonus) Implement a machine learning strategy (e.g. inspired to the mini batch example in the `RLS_ML_Streaming` notebooks) and compare it in terms of accuracy to the best model obtained in the steps above (1 point)

The prediction system should be scalable (in terms of executors) in the number of models. The student must use tables and graphics in the report to illustrate the main results and justify the implementation choices.

4 Deliverables

The student will deliver :

1. the implementation (in Jupyter notebook format) of the online distributed prediction system on the Docker container.
2. a report (in PDF format) presenting
   — Description of the overall architecture, and why it is scalable
   — Experimental results in terms of scalability
   — Experimental results in terms of predictions accuracy
3. a video (AVI format) of max 5 minutes to present this project. The presentation should address the main points illustrated in the report.

A template describing the structure of the project is available on the UV in the Project section.

Rules for project submission

To be read carefully !

1. The assignment should be solved in groups of **maximum** two students.
2. The assignment will be graded on the implementation, the report and the video presentation.
3. Your code should be **commented**.
4. The assignment will be handed in through the dedicated Homework module on the Virtual University.
5. All the deliverables will be put in a single archive, named `INFOH515_<STUDENT_ID>_<LAST_NAME>.zip` where `<STUDENT_ID>` and `<LAST_NAME>` should be replaced by the actual student id and last name of the student(s) in the group.
   The archive should include :
   — **Python Jupyter Notebook** `(*.ipynb)`
   — **Report** `(*.pdf)`
   — **Video of the presentation** `(*.avi)`
   N.B. The maximum allowed size for the archive is 100 MB. If your video is bigger than 100 MB get in touch with the assistant (`jdestefa@ulb.ac.be`) to find an alternative solution.
6. Your project should be submitted on the UV no later than **11PM of May the 24th 2020**.
7. All the projects submitted after the deadline :
   — Penalized of one point if submitted **before 11PM of May the 25th 2020**.
   — Considered as late and will not be accepted anymore if submitted **after 11PM of May the 25th 2020**.
8. Sharing of code is not allowed (you may, however, verbally discuss ideas on how to tackle the project).
9. This project counts for 50% of your grade (10 points). This project **shall be completed individually and it shall represent your sole efforts**. The result or the copy of another group efforts (current, or past, semester(s)), is considered academic dishonesty. Plagiarism, in the sense of copy-pasting from existing reports or code is a serious issue.
10. Each project producing any error during its execution will receive a grade of 0/10.