UNIVERSITÉ LIBRE DE BRUXELLES
Faculté des Sciences
Département d'Informatique

INFO-H-515
Big Data Scalable Analytics

Report - Phase 2

Antoine Passemiers

Academic year 2018 - 2019

CONTENTS

1. Introduction 1

2. Architecture 2

2.1 Methodology 2

2.1.1 Data generation 2

2.1.2 Recursive Least Squares (RLS) with forgetting factor 3

3. scalability 5

1. INTRODUCTION

The objective of this assignment is to implement a scalable online algorithm that is able to perform forecasting in a distributed fashion. More specifically, such algorithm should be able to predict the values of some explained variables based on the values of some explanatory variables by having recourse to linear models. In the framework of this project, the Recursive Least Squares (RLS) algorithm has been privileged as it is capable of learning the coefficients of a linear model incrementally.

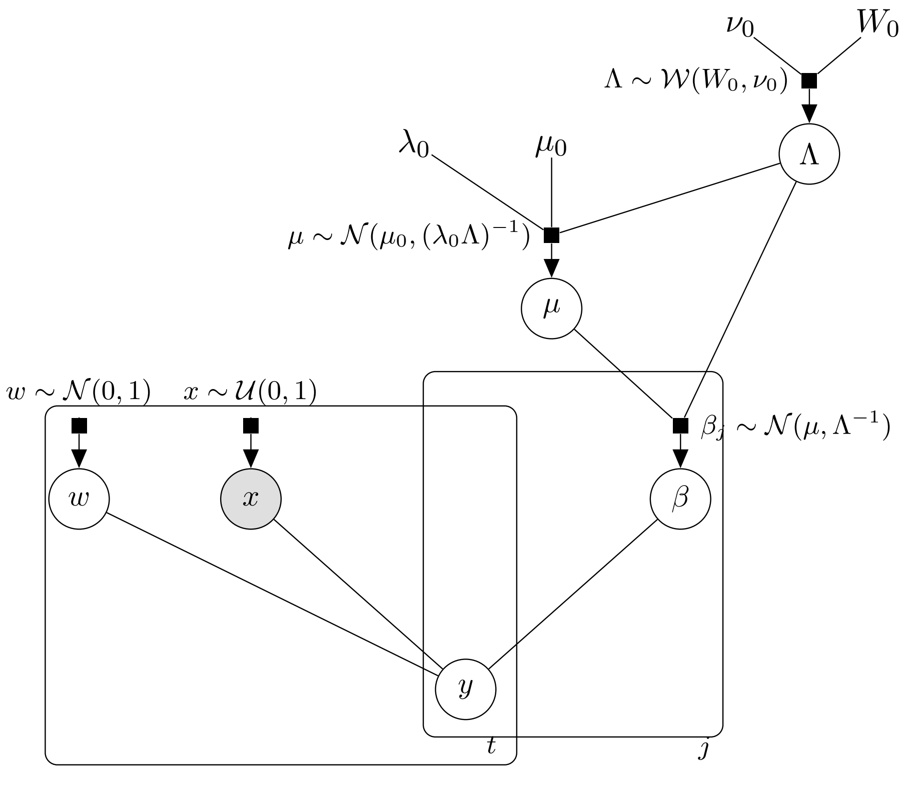
The implementation extends the notebooks that have been provided along with the project statement. There are mainly three extensions of these notebooks:

- The coefficients of the underlying linear models are generated randomly using a multivariate Gaussian distribution. The parameters of the later are drawn at random using a normal-Wishart distribution and fixed before the learning process starts. Also, multiple output/explained variables are returned instead of a single one.
- The system is made scalable with respect to the number of models to be run in parallel. Running multiple models in parallel with different hyper-parameters allows for fast validation and computation of the optimal hyper-parameter values. In this project, the only hyper-parameter to be tuned is the forgetting factor of the RLS algorithm.
- The system is made scalable with respect to the number of explained variables. Also,

2. METHODOLOGY

2.1 Data generation

Observations x are drawn from a uniform distribution, while each latent coefficient vector β_j is drawn from a multivariate Gaussian distribution. Each vector β_j is a column of the coefficient matrix B and fixed beforehand. Instead of choosing arbitrary mean vector and covariance matrix as parameters of the multivariate Gaussian distribution, these parameters are drawn at random. Also, the covariance matrix should be positive semi-definite. Therefore, a normal-Wishart is used to generate a random mean vector and a precision matrix. The final covariance matrix is obtained by inverting the sampled precision matrix.



ν_0
W_0
$\Lambda \sim \mathcal{W}(W_0, \nu_0)$
Λ
λ_0
μ_0
$\mu \sim \mathcal{N}(\mu_0, (\lambda_0\Lambda)^{-1})$
μ
$w \sim \mathcal{N}(0,1)$
$x \sim \mathcal{U}(0,1)$
$\beta_j \sim \mathcal{N}(\mu, \Lambda^{-1})$
w
x
β
y
t
j

Fig. 2.1: Bayesian network representing the randomly generated samples. Plate notation indicates variable repetition across time and output variables, respectively.

2.2 Recursive Least Squares (RLS) with forgetting factor

In the standard RLS implementation with forgetting factor, the weights β are estimated incrementally using the following formulas:

$$\begin{cases} V^{(t)} & = \frac{1}{\nu}\left(V^{(t-1)} - \frac{V^{(t-1)} x_t^T x_t V^{(t-1)}}{1+x_t V^{(t-1)} (x_t^T} \right) \\ \alpha^{(t)} & = V^{(t)} x_t^T \\ e & = y^{(t)} - x_t \hat{\beta}^{(t-1)} \\ \hat{\beta}^{(t)} & = \hat{\beta}^{(t-1)} + \alpha^{(t)} e \end{cases} \tag{2.1}$$

where

First approach – Fully-vectorized version:

$$\begin{cases} \alpha_t & = V^{(t)} x_t^T \\ e & = y^{(t)} - x_t \hat{B}^{(t-1)} \\ \hat{B}^{(t)} & = \hat{B}^{(t-1)} + \alpha_t^T e \end{cases} \tag{2.2}$$

Second approach – Distributed version:

$$\begin{cases} \alpha_t & = V^{(t)} x_t^T \\ e & = y_j^{(t)} - x_t \hat{B}_{\cdot j}^{(t-1)} \\ \hat{B}_{\cdot j}^{(t)} & = \hat{B}_{\cdot j}^{(t-1)} + \alpha_t^T e \end{cases} \tag{2.3}$$

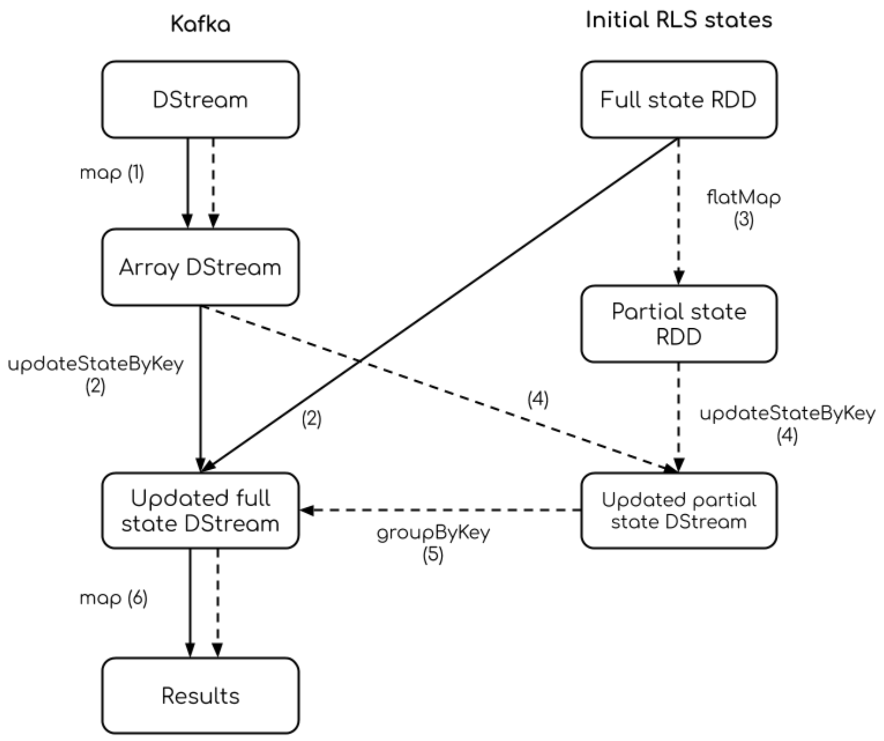


Fig. 2.2: Lineage graph for the proposed architecture. Dashed lines indicate the Spark transformations applied in the distributed version and plain lines indicate the transformations applied in the fully-vectorized version.

3. SCALABILITY

Scalability is the ability of a big data system to process an increasing amount of incoming data by having recourse to an increasing amount of resources. It can be measured by the scaleup, the ratio between the amount of data processed by the model with two different amount of resources but while being run for the same amount of time.

TODO: sub-linear scaleup?