Analyzing data streams for social scientists

Lianne Ippel

orcid.org/0000-0001-8314-0305

Maurits Kaptein

orcid.org/0000-0002-6316-7524

Jeroen Vermunt

orcid.org/0000-0001-9053-9330

Abstract

The technological developments of the last decades have created opportunities to efficiently collect data of many individuals over time. While these technologies provide exciting research opportunities, they also provide challenges: datasets collected using these technologies grow increasingly large, or be continuously augmented with new observations. These data streams make the standard computation of well-known estimators inefficient, as computations are repeated each time new data enter. This chapter details online learning, an analysis method that updates parameter estimates instead of re-estimating them to analyze large and/or streaming data. The chapter presents several simple (and exact) examples of the online estimation for independent observations.

Additionally, social scientists are often faced with nested data: pupils are nested within schools, or repeated measurements are nested within individuals. Nested data are typically analyzed using multilevel models. Estimating multilevel models, however, can be challenging in data streams: the standard algorithms used to fit these models repeatedly revisit all data points, which becomes infeasible in a data stream context. We present a solution to this problem by introducing the Streaming Expectation Maximization Approximation (SEMA) algorithm for fitting multilevel models online. We end this chapter with a discussion of the methodological challenges that remain.

1. Introduction

One of the new challenges of our current digital age is processing and analyzing of the vast amounts of data (Gaber, 2012; Gaber et al., 2005; L'Heureux et al., 2017). Data are collected via many different devices, e.g., smartphones or wearables, and in various contexts like at home, while navigating, or in a hospital. The common characteristic of these data is that the data is often too large to process at once, and/or has an accumulative nature where new data points continue to augment the dataset. Even though computational power is increasing exponentially, the storage, processing, and analysis of such data remains challenging (Ippel et al., 2019, 2016a; Yang et al., 2017). Storing all data might be expensive and computations of complex models can be time consuming. Even the computations of 'simple' models like linear regressions can become too time consuming when using large, or even worse growing, datasets. Moreover, methods typically used to analyze such large datasets are often black boxes, making it difficult to explain the results of these methods (Rudin, 2019).

In this chapter, we address several approaches to analyzing large datasets or even data streams using methods frequently used by social scientists, which are both computationally feasible to analyze large data and maintain their explain-able character. In the next section, we identify four approaches to analyzing large data. We continue this chapter with some examples of models for independent observations illustrating one of these approaches in particular, namely online learning (Bifet et al., 2010; Shalev-Shwartz, 2011). After these examples, we introduce SEMA, an algorithm to estimate Generalized Linear Models for analyzing dependent observations using online learning. This chapter ends with a discussion of future developments and further readings.

2. Approaches to analyze large volumes of data

In this section, we detail four approaches to analyzing large and/or streaming data. We start with two techniques, which are especially beneficial for analyzing large data: subsampling and parallel computing. We end this section with two techniques, which are tailored to analyzing streams of data namely sliding windows and online learning.

2.1 Processing large data

Given that the data is either large and/or computational time is lengthy, one can choose to select only a smaller number of observations to analyze. This process is called subsampling, where one randomly samples observations (i.e., the rows) from the entire dataset and uses this subsample of observations for their analysis (Wang et al., 2019, 2018). This is an appropriate method when the aim of the analysis is to obtain insights of the associations between a set of variables. Repeating the process of subsampling several times will in addition provide insights in the stability of the parameter estimates. However, when using this technique for prediction purposes, much information about the unit of analysis is lost due to subsampling. This is likely to negatively affect prediction performances and at the very least increases standard errors (due to smaller number of observations).

An alternative method for dealing with lengthy computations is dividing the data over several machines. The computations, which would previously be done sequentially are now done in parallel (Böse and Högqvist, 2010; Chu et al., 2007). The result of the computations of each of the separate machines are combined afterwards. Parallel computing is an efficient method to deal with large data, when the methods used allow for parallel computing, i.e., the computations of one part of the data should not depend on other parts of the data. In addition, when data are streaming in, the demand for more or more powerful machines remains and the overhead caused by combining the results from several machines will grow.

2.2 Processing data (as) streams

Instead of processing all data at once, whether it is a subsample or divided across several machines, data might also be analyzed sequentially, either out of necessity for the data is arriving over time or because data are stored in a cloud solution, and are processed sequentially. A sliding window comes down to a subsample of data of either a certain time interval or a certain number of data points, i.e., the window (Gaber, 2012). As new data are streaming in a sliding window ‘moves’ forward excluding the oldest data points and including the new or yet unseen data points. While this method has several strong advantages such as user control about exactly how many resources are used for the analysis and temporal fluctuations are well accommodated, this method also comes with an important downside. It requires domain knowledge to determine the appropriate size of the window. While large windows will increase the chances of capturing enough significant events, it will also increase the demand on the resources. Especially in a

situation with re-occurring fluctuations of events, a sliding window approach can easily miss these patterns due to a too narrow window.

The last approach we discuss is called online learning. This approach updates parameter estimates with information from the most recent data points, instead of computing the parameter estimates again every time new data enters. In doing so, this approach never returns to historical data, thereby speeding up computations vastly and lowering the burden of data storage. Obviously, also online learning does not go without domain knowledge. First, one has to ensure all required variables are included in the analysis as additional information will be 'forgotten', and hence, lost. Secondly, deciding how much weight the new information is given is not always straightforward. Especially in situations with concept drift (Elwell and Polikar, 2011), one needs to make a decision in the bias-variance trade off (Belkin et al., 2019) in terms of how close to follow changes in the data. Following the changes too closely, i.e., relatively much weight is on the most recent data points, or not close enough both will negatively affect the prediction performances as well as the estimates of the associations between variables.

3. Online estimation with independent observations

In this section, we will go into more detail of online learning for independent observations by illustrating the estimation of several parameters. Some of these online estimated parameters are identical to their offline (i.e., computed using all data at once) counterparts (more examples can be found here, Ippel et al., 2016a). Other parameters have to be estimated iteratively, e.g., the regression coefficients of logistic regression. For these kinds of parameters, the online approach might not always result in the exact same result. This might be due to the order in which the data entered, the weight given to new observations and the number of iterations set in the offline setting. First, we discuss some of the identical estimators, followed by an online learning approach, which is an approximate solution to estimate logistic regression coefficients.

3.1 Exact estimators

One of the keystone statistics used in most models used by social scientists is the covariance between two variables. We now illustrate how to compute this parameter online. Let *X* and *Y* be data vectors with *n* entries. Now, the covariance between *X* and *Y* is denoted by:

$$s_{xy} = \frac{\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{n-1} \qquad (1)$$

where i is the index of an entry of the data vector and $\bar{X}$ and $\bar{Y}$ are respectively the averages of X and Y. The first step, however plain perhaps, is keeping the count, n. In online learning, we often use the following notation

new state := previous state + new information,

where ':=' is an assignment sign, replacing the previous state with a new state. Hence, writing an online counter, n, using this online learning formulation, we write

$$n := n + 1$$

Secondly, we compute the average of the variable. Assuming that the number of observations is already updated, we write the online estimation of the mean as follows:

$$\bar{X} := \bar{X} + \frac{X_i - \bar{X}}{n}$$

Here, the weight of the new observation is given by $\frac{1}{n}$, i.e., each data point receives an equal weight. However, when one chooses to alter this weight, for instance $\frac{1}{\min(n,1000)}$ more recent data points receive relatively a higher weight and will therefore influence the parameter estimates more than the older data points. The last part for the online computation of the covariance is more complicated and exist of multiple steps. Assuming n is already updated:

$$\begin{aligned} \bar{X} &:= \bar{X} + \frac{X_i - \bar{X}}{n}, \\ \sum xy &:= \sum xy + \left(X_n - \bar{X}\right)\left(Y_n - \bar{Y}\right), \\ \bar{Y} &:= \bar{Y} + \frac{Y_i - \bar{Y}}{n}, \\ s_{xy} &:= \frac{\sum xy}{n-1}, \end{aligned} \tag{3}$$

where subscript n denotes the most recent data point and $\sum xy$ is the sum of cross products. While $\bar{X}$ is updated first in Eq. 3, the result is the same whether one chooses to update $\bar{Y}$ first (Pébay, 2008). This equation can also be used to compute the variance of a variable, however with a minor adjustment. Line 2 of Eq. 3 in words is, 'new sum of cross products is the previous sum of cross products plus the difference between the last data point and a mean that includes this last data point multiplied by the differences between the last data point and a mean that excludes this last data point'. Now, if one wants to compute the variance, one has to resort to an

auxiliary variable to temporarily store that difference between the last data point and the mean excluding the most recent data point, i.e,

$$\begin{aligned} d &= X_i - \bar{X}, \\ \bar{X} &:= \bar{X} + \frac{X_i - \bar{X}}{n}, \\ \sum xx &:= \sum xx + d\left(X_n - \bar{X}\right), \\ s_x^2 &:= \frac{\sum xx}{n-1}, \end{aligned} \tag{4}$$

where d is an auxiliary variable, $\sum xx$ the sum of squares, and s_x^2 the sample variance. In combining Eq. 3 and Eq. 4 one can also compute correlation estimates in an online manner. Now let us move to an analysis often used by social scientists: linear regression.

This analysis, like covariance estimates, yields identical parameter estimates in both the traditional offline, using all data at once, approach as in an online learning approach. To briefly remind the reader, linear regression coefficients are computed as follows:

$$\hat{\beta} = (X'X)^{-1}X'Y \tag{5}$$

where $\boldsymbol{X}$ is a $n \times p$ data matrix, where p is the number of variables, including a column of 1's for the intercept. Once there are enough observations available for $\boldsymbol{X'X}$ to be invertible (i.e, $\boldsymbol{X'X}$ should be positive definite), one only needs to invert this $X'X$ matrix once, and afterwards directly update the inverted matrix as follows, using the formulation of Sherman-Morrison:

$$(X'X)^{-1} := (X'X)^{-1} - \frac{(X'X)^{-1}x_n x'_n (X'X)^{-1}}{1 + x_n (X'X)^{-1} x'_n} \tag{6}$$

where x_n is the most recent data point or vector with p entries. Second part of Eq. 5 is computed online similar to Eq. 3 line 2,

$$X'Y := X'Y + X_n Y_n \tag{7}$$

Next, multiplying the results of Eq. 6 and Eq. 7 yields identical regression coefficient estimates to the offline estimated coefficients. Unfortunately, not all parameters can be estimated using a closed form expression, and therefore not all parameters estimates are identical in their online and offline estimation approach. We now continue with the discussion of Stochastic Gradient Descent, an online estimation approach to fit, for instance, logistic regression models.

3.2 Approximating estimators

While data scientists often use logistic regression as a classification method, social scientists are often more interested in the regression coefficients of the logistic regression. Now, even fitting a

logistic regression using all data at once is an optimization problem, since there is no closed form expression. This even amplifies the demand for an online estimation procedure as iterative procedures for model fitting quickly become infeasible when data are streaming in. The iterations required to obtain a stable solution for the regression coefficients will require more time as more data are entering and with the new data entering, one has to redo the analysis to remain up to date.

There are various estimation methods and algorithms to estimate a logistic regression model (Hilbe, 2009; Heinze, 2006) and for the purpose of this chapter we look into the Maximum Likelihood framework and use the (Stochastic) Gradient Descent algorithm (Bottou, 2010). In general, Gradient Descent entails the following: derive the first order derivative of the log-likelihood function and set it equal to zero. Then, iteratively update the parameter estimates until convergence is reached. To illustrate Gradient Descent, we provide the example of logistic regression. The log-likelihood function of logistic regression is

$$\ell = \sum_{i=1}^{n} Y_i \log\left(\frac{\exp(X_i\beta)}{1+\exp(X_i\beta)}\right) + (1-Y_i)\log\left(1-\frac{\exp(X_i\beta)}{1+\exp(X_i\beta)}\right) \quad (8)$$

with the first order derivative,

$$\frac{\delta\ell}{\delta\beta} = \sum_{i=1}^{n}\left(Y_i - \frac{\exp(X_i\beta)}{1+\exp(X_i\beta)}\right)X_i \quad (9)$$

Eq. 9 is a summation of the contributions of each of the rows in the dataset to the derivative. This summative nature can be exploited by instead of summing over the entire dataset at once, take intermediate steps towards a more likely solution after adding the contribution of each data point:

$$\hat{\beta} := \hat{\beta} + \lambda\left(Y_n - \frac{\exp(X_i\beta)}{1+\exp(X_i\beta)}\right)X_n \quad (10)$$

where λ is the learning rate giving weight to new observations. Similar to the Eq. 2, the learning rate can be decided upon by the researcher.

The parameters estimated in this section, whether they are obtained through approximations or in closed form solution, all have in common that the parameters are estimated using data from which we assume the data rows are independent of each other. This implies that there is no correlation between data points. However, in social science practice, we often deal with situations where this assumption is violated due to the fact that we have repeated observations of individuals or other kinds of groupings such as employees nested within companies, children in classrooms within schools or citizens in countries. In the next section, we detail how to fit a multilevel model using online learning.

4. Streaming Expectation Maximization Approximation

In this section, we focus on the online estimation with dependent observations. Commonly, dependent observations are analyzed with multilevel models. For the online estimation of these models, we introduce the Streaming Expectation Maximization Approximation (SEMA) algorithm, an online learning algorithm based on the EM algorithm (Ippel et al., 2019, 2016b). Multilevel models have several advantages such as better out-of-sample predictions than models which assume a fixed effect, they are also easier to interpret as the model only exist of three types of parameters (i.e., regression coefficients, variance parameters, and residual variance) (Raudenbush and Bryk, 2002; Skrondal and Rabe-Hesketh, 2004). However, the downside of these models is that they rely on iterations to fit the model, similar to the logistic regression. When data are either large (i.e., long) or augmented with new observations, the estimation time of such a multilevel model quickly becomes infeasible, as well as the required computational power to do the series of matrix inversions which are necessary to estimate the model parameters. While we assume data to enter over time, using SEMA for model estimation can still be beneficial in the case of stationary data. While in a data stream, SEMA will not revisit previously seen observations that is not to say that it is impossible. In a stationary dataset, SEMA can be used to iterate over the dataset more efficiently than the offline method, using less iterations in order to converge (Ippel et al., 2016b). In this section, we first detail the multilevel model and highlight one of the commonly used estimation algorithms to fit the model, i.e., EM algorithm. We, then, continue with the discussion of SEMA.

4.1 Multilevel model

When the assumption of independent observations is violated, social scientists often resort to multilevel models to account for these dependencies (Raudenbush and Bryk, 2002; Skrondal and Rabe-Hesketh, 2004). For instance, assuming that a school effect on student performance is normally distributed and within a school the children's performances are also normally distributed, we can estimated a 'normal × normal' model, however other distributions that fall within the framework of the exponential family (e.g., beta binomial or negative binomial) can be accounted for similarly. In multilevel modeling, we refer to level 1 as the lower level, e.g., observations, and level 2 to the higher level, e.g., individuals. Instead of assuming a fixed effect, which is the same for each individual, in this model we estimate individual effects. These individual effects are not directly observable as these are coming from (a) latent variable(s). Assuming normally distributed individual effects, and normally distributed errors; the model formulation is then as follows:

$$y_{ij} = x_{ij}\beta + z_{ij}b_j + \varepsilon_{ij} \quad (11)$$

where y_{ij} is observation i of person j, x_{ij} is a vector of p fixed effect covariates, z_{ij} is a vector of r random or individual effect covariates, β is the fixed effect regression coefficient, b_j are individual effects and $b_j \sim N(0, \tau^2)$, $\varepsilon_{ij}$is the error term per observation and $\varepsilon_{ij} \sim N(0, \sigma^2)$, where $b_j \perp \varepsilon_{ij}$. Additionally, let J be the number of individuals, n the number of observations, and n_j the number of observations from one individual.

4.2 Model estimation using EM algorithm

An option to estimate the coefficients of Eq. 11 is using the 'Expectation-Maximization' algorithm. In short, the algorithm works as follows: In the first step, the Expectation step, the unobserved values (of the latent variable) are predicted, given the current set of parameter estimates. The second step, the Maximization step, then maximizes the (log-)likelihood given these predictions, thereby updating the estimates of the parameters. Alternating between these two steps, EM algorithm will obtain the maximum likelihood estimates.

4.2.1 E step

The E step consists of three equations, one for each type of parameter: β, τ^2, and σ^2 to compute the Complete Data Sufficient Statistics, CDSS. We use the term complete data because we treat the predicted values as if they were observed. We refer to the CDSS as T_1, T_2, and T_3, for respectively β, τ^2, and σ^2. Each will be discussed in turn, starting with T_1:

$$T_{1(k)} = \sum_{j=1}^{J} X'_j Z_j \hat{b}_{j(k)} \quad (12)$$

where X_j is an $n_j \times p$ matrix, Z_j is an $n_j \times r$ matrix, k indexes the current iteration, $T_{1(k)}$ is an $p \times 1$ vector, $\hat{b}_{j(k)}$ is an $r \times 1$ vector and defined as,

$$\hat{b}_{j(k)} = C_{j(k)}^{-1}(Z'_j y_j - Z'_j X_j \hat{\beta}_{(k-1)}), \quad (13)$$

where $C_{j(k)}$ is an $r \times r$ matrix which quantifies the uncertainty of $\hat{b}_j$, and is given by:

$$C_{j(k)} = Z'_j Z_j + \hat{\sigma}^2_{(k-1)} \hat{\tau}^{-1}_{(k-1)} \quad (14)$$

Second, $T_{2(k)}$ is computed as follows

$$T_{2(k)} = \sum_{j=1}^{J} \hat{b}_{j(k)} \hat{b}'_{j(k)} + \hat{\sigma}^{2}_{(k-1)} \sum_{j=1}^{J} C^{-1}_{j(k)}, \tag{15}$$

where, $T_{2(k)}$ is an $r \times r$ matrix. Lastly, $T_{3(k)}$ is given by

$$T_{3(k)} = \sum_{j=1}^{J} u'u + \hat{\sigma}^{2}_{(k-1)} tr\left(\sum_{j=1}^{J} C^{-1}_{j(k)} Z'_j Z_j\right) \tag{16}$$

where $u = y_j - X_j \hat{\beta} - Z_j \hat{b}$, is the residual.

4.2.2 M step

Using the updated CDSS, in the M step the parameter estimates are updated. In iteration k, β,is computed as follows:

$$\hat{\beta}_{(k)} = \left(\sum_{j=1}^{J} X'_j X_j\right)^{-1} \sum_{j=1}^{J} X'_j Y_j - T_{1(k)} \tag{17}$$

The $\hat{\tau}^{2}_{(k)}$ is equal to:

$$\hat{\tau}^{2}_{(k)} = \frac{T_{2(k)}}{J}, \tag{18}$$

Lastly, $\hat{\sigma}^{2}_{(k)}$is given by:

$$\hat{\sigma}^{2}_{(k)} = \frac{T_{3(k)}}{n} \tag{19}$$

4.3 Online model estimation using SEMA

There are several operations presented in the previous sections, which would make the online estimation of the multilevel model infeasible in a growing dataset. For instance, the matrix multiplication and inversion ($(X'X)^{-1}$) is a costly operation, which would have to be computed again every time an up-date is desired. In this section, we will detail the adaptations which allow for online estimation. However, note that, the online estimation will not result in the exact same parameter estimates when only a few observations have been processed as the offline estimation procedure. When more (i.e., tens of thousands) of observations have been processed, the estimates of the parameters will be the same or at least highly similar.

4.4 Online E step

In this section we will use the '∼' to differentiate between the offline and online estimated parameters. We refer to the individual which is generating the most recent data point as j_t, where t indexes the most recent data point. Since all three CDSS are sums over individuals, the online update for each of the CDSS follows this logic:

CDSS := CDSS - previous contribution + updated contribution

The online computation of $\tilde{T}_1$ is as follows:

$$\tilde{T}_1 \coloneqq \tilde{T}_1 - T_{1j_{t(t-1)}} + T_{1j_t} \tag{20}$$

where T_{1j_t} is defined as,

$$T_{1j_t} = X'_j Z_j \hat{b}_j, \tag{21}$$

where the online computation of $X_j^{'} Z_j$ equals

$$X_j^{'} Z_j \coloneqq X_j^{'} Z_j + X_{ij} Z_{ij}^{'}. \tag{22}$$

Second, the CDSS $\tilde{T}_2$ is computed as follows

$$\tilde{T}_2 \coloneqq \tilde{T}_2 - T_{2j_{t(t-1)}} + T_{2j_t} \tag{23}$$

where T_{2j_t} is given by

$$T_{2j_t} = \hat{b}_j \hat{b}_j^{'} + \hat{\sigma}^2 C_j^{-1}, \tag{24}$$

where $\hat{b}_j$ is computed online exactly the same as offline (Eq. 13), where the product of $Z_j^{'} Y_j$ is computed similar online to Eq. 22 and $Z_j^{'} X_j$ is the transpose of that same equation. The online computation of C_j online requires the online matrix multiplication presented earlier in Eq. 22. Lastly, the computation of $\tilde{T}_3$,

$$\tilde{T}_3 \coloneqq \tilde{T}_3 - T_{3j_{t(t-1)}} + T_{3j_t} \tag{25}$$

where the individual contribution is given by

$$T_{3j_t} = Y_j^{'}Y_j + \hat{\beta}^{'} X_j^{'}X_j\hat{\beta} + \hat{b}_j^{'}Z_j^{'}Z_j\hat{b}_j - 2Y_j^{'}X_j\hat{\beta} - 2Y_j^{'}Z_j\hat{b}_j + 2\hat{\beta}^{'} X_j^{'}Z_j\hat{b}_j + \hat{\sigma}^2\, tr(C_j^{-1})$$

(26)

For Equation 26, we have to store several components, to ensure we do not have to redo the computations when new observations come in. For instance, we have to store $X_j^{'}X_j$ matrix, similar to $Z_j^{'}Z_j$, $Y_j^{'}X_j$, $Y_j^{'}Z_j$, and lastly $X_j^{'}Z_j$. All these matrix and vector products are updated like Eq. 22. When the new contribution is computed, these matrices are multiplied with the relevant parameters.

4.5 Online M step

In the case of fitting a multilevel model where both the random effects as well as the error terms are assumed to be normally distributed, the M step of EM algorithm is computationally simple. Adapting this step to fit in an online learning algorithm is therefore rather straightforward: the maximization of τ^2 and σ^2 remain exactly the same. The estimation of the fixed effect regression coefficients, β, is altered slightly to fit the online framework. The main adaptation is in the matrix inversion of $X'X$. While the inverse of C_j matrix has to be computed for the most recent individual to update the model, the $(X'X)^{-1}$ can be updated directly (Ippel et al., 2016a, 2019; Sherman and Morrison, 1950; Plackett, 1950). The reason of this difference lies in the fact that the computation of $C_j$depends on continuously changing estimates of the model parameters and the values of the latent variables, while $X'X$ only dependents on the values of the observed covariates. Once $X'X$ is invertible, the inverse of this matrix can be updated using the formulation presented in Equation 6. You can find an R package with the SEMA algorithm *https://github.com/L-Ippel/SEMA*

5. Discussion

This chapter introduces several approaches to analyzing data streams with independent and dependent observations. Four approaches were suggested in how to process large and/or streaming data. We introduced online learning estimation procedure for models commonly used by social scientists, such as correlations and linear regression. However, for many models online

learning approaches have not yet been developed. For instance, commonly used Machine Learning algorithms, e.g., random forest, and Neural networks are challenging to estimate in an online learning manner. The research field of analyzing data streams is growing vastly.

In addition to exploring new methods to analyze data streams, methodological issues regarding analyzing data streams is a pioneering research field. Currently, open questions are, for instance, the treatment of (temporarily) missing data. To illustrate, it is yet unclear how one should handle data streams where not all covariates are observed at once, resulting in missing values. While randomly missing values can cause an increase of the standard error, the problem becomes even more challenging when values are missing due to attrition: a particular subgroup of observations drops out of the stream. This will likely lead to biased estimates.

Related to the issue of systematic dropout is concept drift (e.g., Zliobaite, 2009), where the data generating model fluctuates over the data collecting period. There are several approaches to handle such fluctuations over time. One branch of research is focused on the auto-correlation models where previous observations are taken into account for new predictions (e.g., Cappé, 2011). Another branch of research focuses on forgetting factors also known as learning rates. This learning or forgetting parameter determines the weight of the newly observed data compared to the weight of the historical data. A simple example is the computation of the sample mean: $\frac{1}{\min(n;1000)}\sum_{i=1}^{n} x_i$. Computing the sample mean like this, gives equal weight to all observations, until $n = 1000$. When additional observations augment the dataset, these observations will influence the sample mean more than the historic data allowing the mean to fluctuate more with the recent data.

An additional complication in analyzing data streams with fluctuations over time arises when observations from an individual are collected at highly skewed time intervals. Observations closer in time are more strongly correlated than observations which are spread out over a longer time interval. These differences in time intervals might, therefore, cause bias in the individual predictions as dependencies between close-in-time observations compared to distance-in-time observations might not be picked up adequately by the model.

Lastly, analyzing data streams, similar to analyzing static datasets, requires a well-designed research plan. This plan should entail which method and model that will be used and which variables will be collected. Moreover, it should also contain which tests will be done at which point in time, to prevent type 1 error inflation. In data streams, this research plan is even more important than in the case of static data analysis, since data that was not stored, is lost. It also means that prior to the data collection, one has to consider the purpose of the study, e.g.,

different strategies apply for a prediction whether someone will click on an advertisement versus understanding the influence of sentiment after a match of a national soccer match on stock market behavior, and how long or how many observations will be collected.

Using data streams for the understanding of social behavior is an exciting new research area. It allows novel research questions to be asked using innovative research methods. More and more tools are becoming available for the interested researcher such as Rapidminer (Hofmann and Klinkenberg, 2013) or Massive Online Analysis (Bifet et al., 2010) which allow mining and learning from these data streams.

References

Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences, 116* (32), 15849–15854. doi: 10.1073/ pnas.1903070116

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive Online Analysis. *The Journal of Machine Learning Research, 11*, 1601–1604. Retrieved from http://dl.acm.org/citation.cfm?id=1756006.1859903{\%}5Cnpapers3://publication/ uuid/3EE94251-4014-4FD0-AD94-606713F1845F

Böse, J.-h., & H¨ogqvist, M. (2010). Beyond Online Aggregation: Parallel and Incremental Data Mining with Online Map-Reduce.

Bottou, L. (2010). Large-Scale Machine Learning with Stochastic Gradient Descent. In *Proceedings of the 19th international conference on computational statistics (compstat 2010)* pp. 177–187.

Cappé, O. (2011). Online Expectation-Maximisation. In K. Mengersen, M. Titterington, C. Robert, & P. Robert (Eds.), *Mixtures: Estimation and applications* pp. 1–20. Wiley.

Chu, C., Kim, S. K., Lin, Y., & Ng, A. Y. (2007). Map-Reduce for Machine Learning on Multicore. In B. Schölkopf, J. C. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems* (19th ed., Vol. 19, p. 281). Massachusetts Institute of Technology. doi: 10.1234/12345678

Elwell, R., & Polikar, R. (2011). Incremental learning of concept drift in nonstationary environments. I*EEE Transactions on Neural Networks, 22* (10), 1517-1531. doi: 10.1109/TNN.2011.2160459

Gaber, M. M. (2012). Advances in data stream mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2* (1), 79–85. doi: 10 .1002/widm.52

Gaber, M. M., Zaslavsky, A., & Krishnaswamy, S. (2005). Mining Data Streams: A Review. *SIGMOD, 34* (2), 18–26.

Heinze, G. (2006). A comparative investigation of methods for logistic regression with separated or nearly separated data. *Statistics in Medicine, 25* (24), 4216-4226. doi: 10.1002/sim.2687

Hilbe, J. M. (2009). *Logistic regression models*. Chapman and Hall/CRC. doi: 10.1201/9781420075779

Hofmann, M., & Klinkenberg, R. (2013). *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Boca Raton, Florida, USA: Chapman & Hall/CRC.

Ippel, L., Kaptein, M.C, & Vermunt, J.K. (2016a). Dealing with data streams: An online, row-by-row, estimation tutorial. *Methodology, 12* (4). doi: 10.1027/1614-2241/a000116

Ippel, L., Kaptein, M. C., & Vermunt, J. K. (2016b). Estimating RandomIntercept Models on Data Streams. *Computational Statistics & Data Analysis*, 169–182. doi: 10.1016/j.csda.2016.06.008

Ippel, L., Kaptein, M. C., & Vermunt, J. K. (2019, Mar 01). Estimating multilevel models on data streams. *Psychometrika, 84* (1), 41–64. doi: 10.1007/s11336-018-09656-z

L'Heureux, A., Grolinger, K., Elyamany, H. F., & Capretz, M. A. M. (2017). Machine learning with big data: Challenges and approaches. *IEEE Access, 5* , 7776-7797. doi: 10.1109/ACCESS.2017.2696365

Pébay, P. (2008). Formulas for Robust, One-Pass Parallel Computation of Covariances and Arbitrary-Order Statistical Moments. *Sandia Report, SAND2008-6* (September), 1–18. Retrieved from http://www.ntis.gov/search/product.aspx?ABBR=DE20111028931%5Cninfoserve.sandia.gov/sand_doc/2008/086212.pdf

Plackett, R. (1950). Some Theorems in Least Squares. *Biometrika, 37*, 149– 157.

Raudenbush, S., & Bryk, A. (2002). *Hierarchical Linear Models: applications and Data Analysis Methods* (2nd ed.; J. de Leeuw, Ed.). Thousand Oaks, California, USA: Sage Pulication.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence, 1* (5), 206–215. doi: 10.1038/s42256-019-0048-x

Shalev-Shwartz, S. (2011). Online Learning and Online Convex Optimization. *Foundations and TrendsR in Machine Learning, 4* (2), 107–194. doi: 10 .1561/2200000018

Sherman, J., & Morrison, W. J. (1950). Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix. *The Annals of Mathematical Statistics, 21* (1), 124–127. doi: 10.1214/aoms/1177729893

Skrondal, A., & Rabe-Hesketh, S. (2004). *Generalized latent variable models: multilevel, longitudinal, and structural equation models* (Vol. 17). doi: 10.1007/BF02295939

Wang, H., Yang, M., & Stufken, J. (2019). Information-based optimal subdata selection for big data linear regression. J*ournal of the American Statistical Association, 114* (525), 393-405. doi: 10.1080/01621459.2017.1408468

Wang, H., Zhu, R., & Ma, P. (2018). Optimal subsampling for large sample logistic regression. J*ournal of the American Statistical Association, 113* (522), 829-844. (PMID: 30078922) doi: 10.1080/01621459.2017.1292914

Yang, C., Huang, Q., Li, Z., Liu, K., & Hu, F. (2017). Big data and cloud computing: innovation opportunities and challenges. I*nternational Journal of Digital Earth, 10* (1), 13-53. doi: 10.1080/17538947.2016.1239771

Zliobaite, I. (2009). Learning under Concept Drift: an Overview. Training, abs/1010.4:1–36.