Real-time prediction of manufacturing lead times in complex production environments

Ádám Szaller[a] (*szaller.adam@sztaki.mta.hu*)
[a]*Institute for Computer Science and Control, Hungarian Academy of Sciences*

Ferenc Béres[a]

Éva Piller[a]

Dávid Gyulai[a]

András Pfeiffer[a]

András Benczúr[a]

Abstract

In recent, dynamically changing production environments, accurate prediction of manufacturing lead times is more complicated than ever before, with traditional methods not always applicable. Given the large amount of data that can be gathered from the processes, it is a natural idea to deploy machine learning for lead time prediction. We show that linear regression and in particular boosted trees achieve accurate lead time predictions in near real-time. The efficiency of the method is presented by experimental results, obtained from a simulation-based test case.

Keywords: machine learning, manufacturing lead time, data analytics

Introduction

Recent global trends and their impact on manufacturing industry

A recent global trend is the continuously increasing turbulence of markets, resulting in the need of rapid responses from the industry. On the customers' side, high service level is expected, manifesting in the need of customized products that need to be delivered within a narrow time-window. These requirements create complex problems on the side of the manufacturing industry, as a high variety of products have to be produced, moreover, planning and control decisions must be made quickly and efficiently.

The complexity of manufacturing systems is higher than ever before, as products in a high variety typically require a wide range of technologies and equipment. Besides the increasing complexity, of course, manufacturing systems and technology have also advanced a lot in several aspects during the past years. Cyber-physical systems are part of the everyday's practice, and they make it possible to collect detailed data about the products, processes and resources near real-time (Monostori et al., 2016). This leads to a massive set of detailed data that accompanies the products through their way in the value chain, from the design stage to their use by the customers. This data is called the digital thread, and enables decision makers to increase the efficiency of the processes related to the creation of products (Helu et al., 2017).

Lead time prediction in complex environments

In order to utilize the digital thread efficiently, decision making processes and advanced analytics techniques need to be applied. Considering lead time prediction as a typical problem in production control, one can identify that traditional methods are not always able to support decision making efficiently. Manufacturing lead time is the time that a product spends in the manufacturing system in between two selected processes. It characterizes the dynamics of the system, and also forms the basis of the most important control decisions, e.g. prioritization of jobs, selection of routings and setting of due dates.

As for the conventional lead time prediction methods, most fundamental ones are based on Little's law (Little, 2011), however, they are typically applied in stationary environments, simple process chains and a limited variety of product range. In order to predict lead times, a data analytics and machine learning based approach is proposed in the paper that relies on the actual state of the production environment utilizing the digital thread. The method is aimed at capturing all aspects that influence production lead times in a complex, changing manufacturing environment.

Problem statement

Manufacturing lead time prediction

Lead time is the span between that a given job spends in the system, starting with its release until its completion. We consider the task of predicting the lead time of a job before its release. In a complex manufacturing environment, several different products share the same resources. In addition, technological and process parameters are typically product type dependent. In such cases, it is complicated to calculate the expected lead times, due to the dynamics of the system and the interdependencies of various factors.

In the paper, lead time prediction is performed by knowing the actual state of the system at any given point of time, along with the parameters of the jobs whose lead time is in question.

The input data of the prediction is provided by the Manufacturing Execution System (MES) of the production environment in quasi real time. This data includes log entries about process completions, therefore, status of jobs and their location within the system are always known. The proposed prediction method and the corresponding data analytics environment were elaborated by considering a realistic flow-shop manufacturing environment. The tests and algorithm fine tuning were performed by applying a discrete-event simulation model, which represents the manufacturing environment together with the process dynamics, stochasticity of the parameters. Moreover, the model is also capable of simulating the MES system operation by streaming production log data in quasi-real-time. The details of the simulation model and the process parameters are provided in the following section.

Description of the simulation model

In the analyzed case, the digital thread is composed of data about the execution of the manufacturing processes, the flow of materials in the system and the state of the resources. The operation of the system is represented by its discrete-event simulation model (implementing a digital twin (Rosen et al., 2015)), which is the most common, and efficient way of analyzing complex production and logistics processes. The simulation model is linked to the data analytics toolset, as described later. The model is implemented in Siemens Tecnomatix Plant Simulation, and it is capable of streaming production data real-time towards the lead time prediction model. Important to note that applying a simulation model as a testbed supports the model development, fine tuning and commissioning, however, the main objective is to apply the final, fledged version of the models in a real production environment.

The simulation model represents a realistic flow-shop production environment, including 7x4 machines. This means that the system consists of seven processing stages, with four parallel, alternative resources in each stage. In front of each machine, a buffer is placed with a capacity of 10 parts. After the necessary manufacturing processes has been performed, products are tested to identify the functionally failed products that are transferred to a buffer dedicated to items to be reworked. As for the product failures and reject rates, two different cases were investigated (in both cases, a certain product can be reworked only one time - after that, it is sure that the product is appropriate):

1. In total, 10% of the jobs are randomly (with a uniform distribution) marked as failed items that need be reworked. The station which actually does the rework is chosen randomly from the routing of the job, simulating random failure root causes.
2. Machine availabilities are imperfect, as they can break down for a certain time. Machine breakdowns randomly happen in between 3-5 hours, following a uniform distribution. The duration of a breakdown is also a random timespan with uniform distribution between 45-90 minutes. When a machine is failed, it continuously produces failed products, which have to be reworked on the same workstation.

The difference between the production stages is their manufacturing time that are stochastic by nature, following a normal distribution predefined by parameters (Table 1). In the second case, if machines are considered as elements of a matrix, and a row index of the machines are denoted by x, then the manufacturing time of a certain machine is the following (in seconds):

Table 1 – Cycle times of the machines in the two investigated cases

	Expected value	**Deviation**	**Lower bound**	**Upper bound**
First case	60 sec	30 sec	20 sec	100 sec
Second case	x*60 sec	x*60/4 sec	x*60-x*15 sec	x*60+x*15 sec

In the analyzed production system, four product types (A,B,C,D) are produced. Neither product, nor job routings are definite, but they are described by probability functions. For example, on product B, three manufacturing steps are performed: the first step can be done only on SingleProc7, the second can be done on SingleProc15 or on SingleProc22 with 50%-50% probability, and the third step is performed by SingleProc20 or SingleProc27 with 80% and 20% probability. The material flow of product B is visualized with a Sankey diagram on Figure 1. All product types have similar, characteristic routings with different number of manufacturing steps.

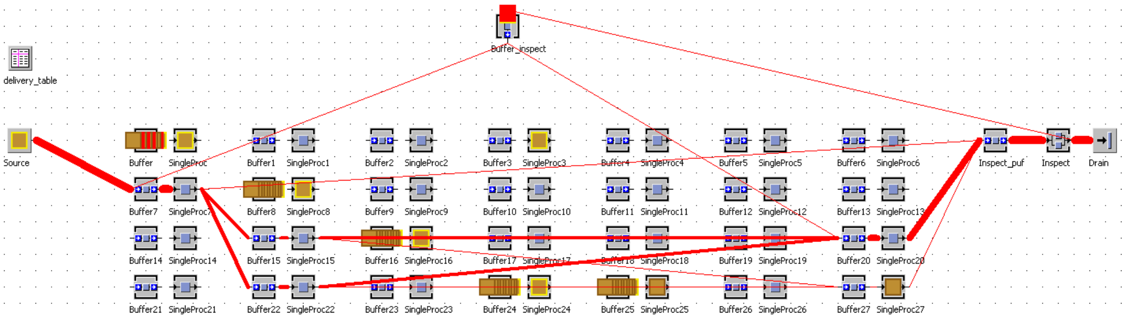

Figure 1 – Material flow of product B visualized on a Sankey diagram

Lead time prediction

For lead time prediction, we implemented a data analytics toolset, composed of a NoSQL database, a feature calculator and a lead time predictor with a dashboard (Figure 2). This framework can provide almost real time predictions for jobs in case of streaming data sources (e.g. the simulation model).

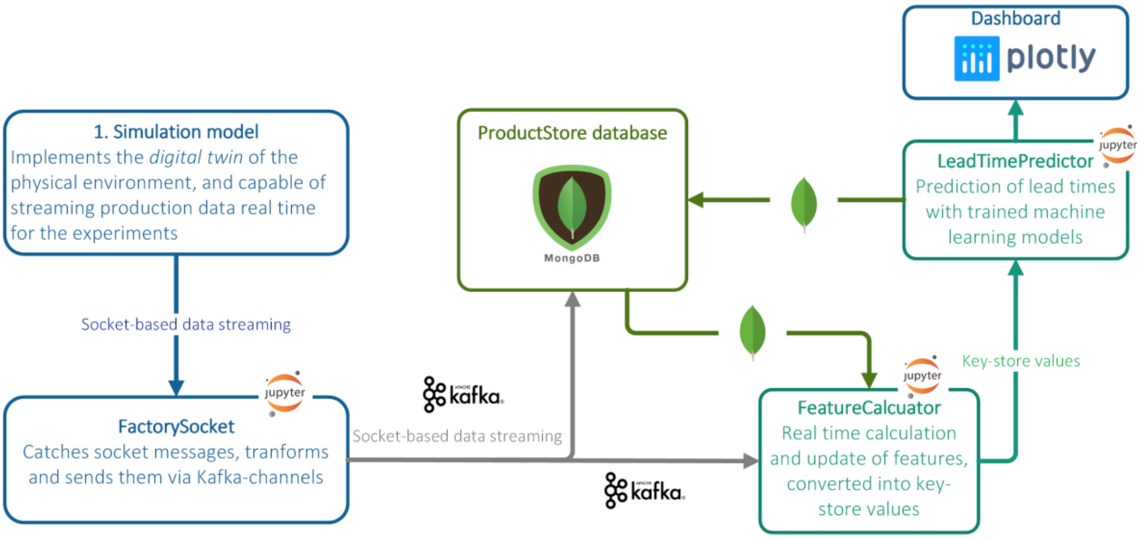
Dashboard
plotly
1. Simulation model
Implements the *digital twin* of the physical environment, and capable of streaming production data real time for the experiments
ProductStore database
MongoDB
LeadTimePredictor
Prediction of lead times with trained machine learning models
jupyter
Socket-based data streaming
kafka
Key-store values
FactorySocket
Catches socket messages, tranforms and sends them via Kafka-channels
Socket-based data streaming
FeatureCalcuator
Real time calculation and update of features, converted into key-store values

Figure 2 - Architecture for real time prediction of lead times

In what follows, we will describe the design and evaluation of the feature calculator and lead time predictor modules.

Little's law

Little's law (Little, 2011) states that the long-term average number L of products in a stationary system is equal to the long-term average effective arrival rate λ multiplied by the average lead time W. Hence $W = L/\lambda$, which we will use both as a direct baseline lead time prediction and a predictive variable in our experiments. Note that it is a key assumption in Little's law that the system is stationary, hence we cannot expect to hold under our highly non-stationary setting. If the system is not stationary, we measure different L and λ at different time intervals. We will also experiment with the time window for calculating L and λ to give lead time prediction at a given point in time.

Machine Learning

Machine learning, predictive analytics, or regression analysis are all different names of producing qualitative predictions derived from existing data. Traditionally, predictive methods belong to statistics, which was already using decision trees (Safavian et al., 1998) and regression (Cox, 1958) from early times. Recently, boosting became a preferred technique used in most data analysis challenge solution projects (Chen et al., 2016) due to its high accuracy and good model interpretability. Next, we describe these methods, which we will use in our experiments.

Simply stated, a **linear regression** model is a monotonic transformation of the linear combination of the variables (Cox, 1958). The advantage of the linear model is that it explains the importance of the variables by the coefficients as weights. Linear regression has a drawback: it is prone to overfitting correlating variables. Two improved versions that we test in our experiments is **LASSO** (least absolute shrinkage and selection operator) that performs both variable selection and regularization (Tibshirani, 1996) and Huber's regression (Rousseeuw and Leroy, 2005) that is robust to outliers in the data.

Regression trees (Safavian et al., 1998) are inductively built by splitting the actual set of events along the variable that reduces the variance the most after the split. The disadvantage of regression trees is that they tend to overfit deeper down in the tree, since the decisions are made based on decreasingly fewer events.

Boosting (Freund et al., 1997) uses the idea to train simple classifiers, for example, small decision trees, by gradually improving the prediction quality in iteration cycles. The main advantage compared to large decision trees is that boosting obtains training gained over the entire data and not just a subset in all the iterations. We will also use boosting for **feature selection**: The first shallow decision tree, by construction, involves the variables with strongest predictive power. The next tree is trained on the difference of the predicted and the actual value (the residual), hence the variables in the next tree have the strongest predictive power *independently of the previous set*, after removing the effects of the previously selected variables.

Evaluation metrics

The prediction of manufacturing lead times is a regression problem, as a numerical value is to be predicted in the know of various features. **Symmetric mean absolute percentage error** (SMAPE) is used to describe the prediction error:

Equation (1):

$$SMAPE\ =\frac{1}{n}\sum_{i=1}^{n}\frac{|F_i-A_i|}{(|F_i|+|A_i|)\ /\ 2},$$

where F_i is the forecasted and A_i is the actual lead time of the i^{th} job. In this section conventional techniques and machine learning models are compared based on SMAPE.

Feature engineering and selection

We extracted several variables from manufacturing time series that describe the exact state of the system when a new job enters. The number of jobs in workstations or buffers, product arrival rate and various other statistics were also calculated based on the recent or long-term historical production data to improve the proposed lead time predictors, including:

- Product type and number of job entering in last 1 and 24 hours, by type.
- Work in progress (WIP, number of jobs in the system), actual average lead time, and lead time estimated by Little’s law. All three figures broken down by product type and routings, including failure routings. All figures measured at present, and in the lasts 100, 500, 800 and 1000 seconds.
- Time elapsed since the last job entered the factory for each product type, routing, or failed product.
- Number of failed stations.

The purpose of feature engineering is to provide a sufficiently rich attribute table for the machine learning methods. However, linear models are known to produce suboptimal results when trained over a large number of correlating features. We propose and evaluate the following feature selection method in conjunction with the linear models. We compute a Gradient Boosted Tree regressor and for each feature, we used the average gain of splits by that feature as an importance measure (Table 2). We selected the eight most important features, which are listed in Table 3. The table also shows regression coefficients and the Spearman correlation with lead time.

Table 2 – Prediction error (SMAPE) is better for linear models using boosting based feature selection. Models were trained on the first 30 days of the second simulation.

Model	Without feature selection	With feature selection
Linear Regression	0.2453	0.2100
LASSO	0.2428	0.2095
Huber Regression	0.2440	0.2040

Table 3 – List of the eight most important attribute selected by Gradient Boosted Tree for the second simulation in decreasing order. The coefficients of the linear models trained over the first 30 days. We marked counterintuitive regression coefficients by asterisk (*).

Feature name	Importance	Spearman	Linear Regression	LASSO	Huber
Mean lead time of current product type	104.07	0.83	-0.15*	0.01	0.02
WIP in current routing	49.67	0.91	213.87	198.70	154.01
Mean of lead times for failed products on current routing	41.26	0.85	-0.06*	0.12	-0.04*
Mean of lead times on the current routing	33.77	0.88	0.60	0.42	0.66
Is product type A?	17.35	0.75	3046.41	0.00*	2.24
Index of product in the current batch	11.60	0.14	122.54	107.10	99.32
Time elapsed since last job entered with same product type	10.74	0.14	-0.11	-0.13	-0.14
Is product type C?	8.87	-0.42	125.83	112.77	20.01

Experiments

We compared the prediction error of various prediction methods for both simulations. For model fitting and evaluation, we used the data up to a given point in time for training and the remaining data for testing. We present SMAPE for the testing period. For each job, we predicted the lead time when a job entered the system. We consider mean lead time from the past and the estimation based on Little's law, in addition to the machine learning based predictions.

In general the prediction error was larger for reworked jobs compared to normal products (Figure 3). In the first simulation we have no information about whether the given product needs to be reworked later when it enters the factory. In the second case if we observe the number of reworked products per workstations in the recent history, then we may guess whether the routing of the current job contains any malfunctioning processes at the moment. In this case, the normal lead time will increase by the length of the additional rework process.

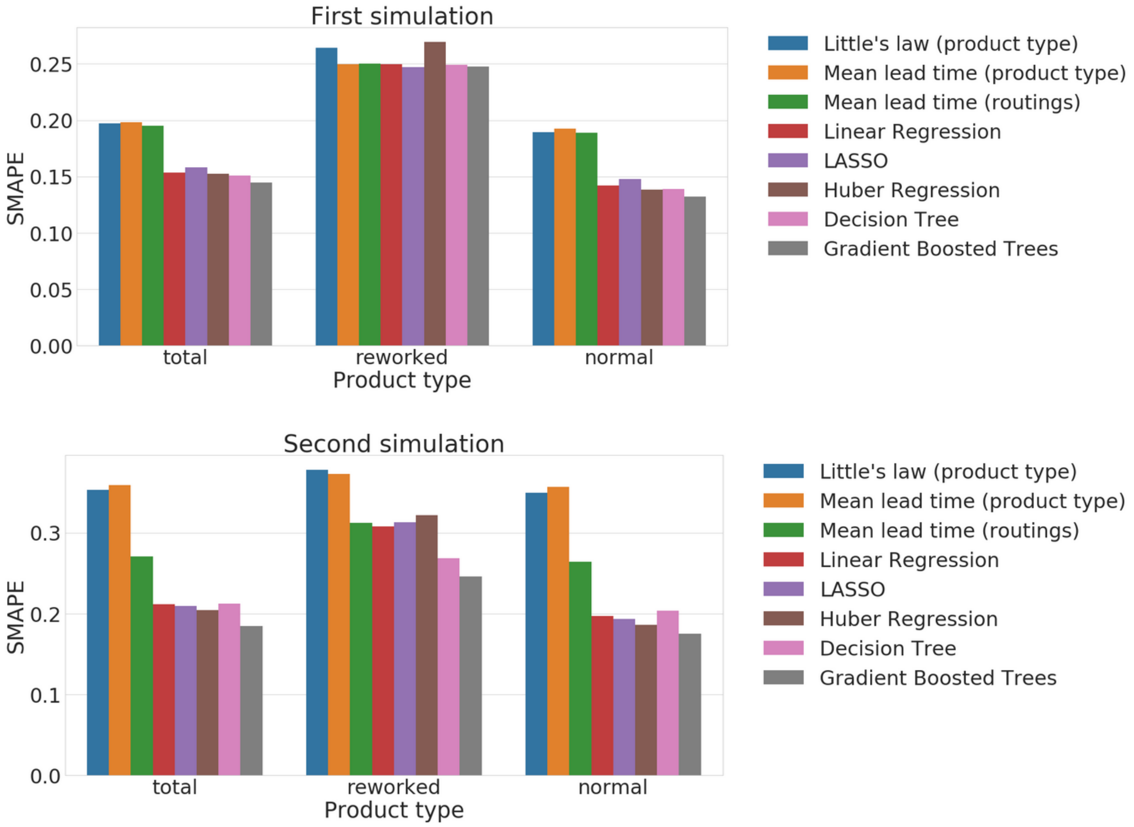

Figure 3 – Predictive performance is evaluated for all products (total), only reworked or only normal jobs. We used the first 30 days of each simulation for model fitting.

For both simulations, machine learning methods performed better than the past average and Little's law. However, for the second simulation, the gain compared to baselines was significantly larger (0.17) than for the first case (0.05). This is due to the non-stationary distribution of the second simulation (see Table 1) where the total manufacturing time for different product routings can greatly vary. Also note that average lead time by routings performs much better than simply by product type in the second simulation (Figure 3), where we may also guess reworked jobs in advance by observing failure rate of workstations.

The results show that the length of the training period (first 7 or 14 days) has only minor impact on the prediction error (Figure 4). However, using just a few days to fit models could introduce many missing values, as we may not know some statistics for all the product routings. During the experiments, we replaced missing values with the average value of known records for each attribute.

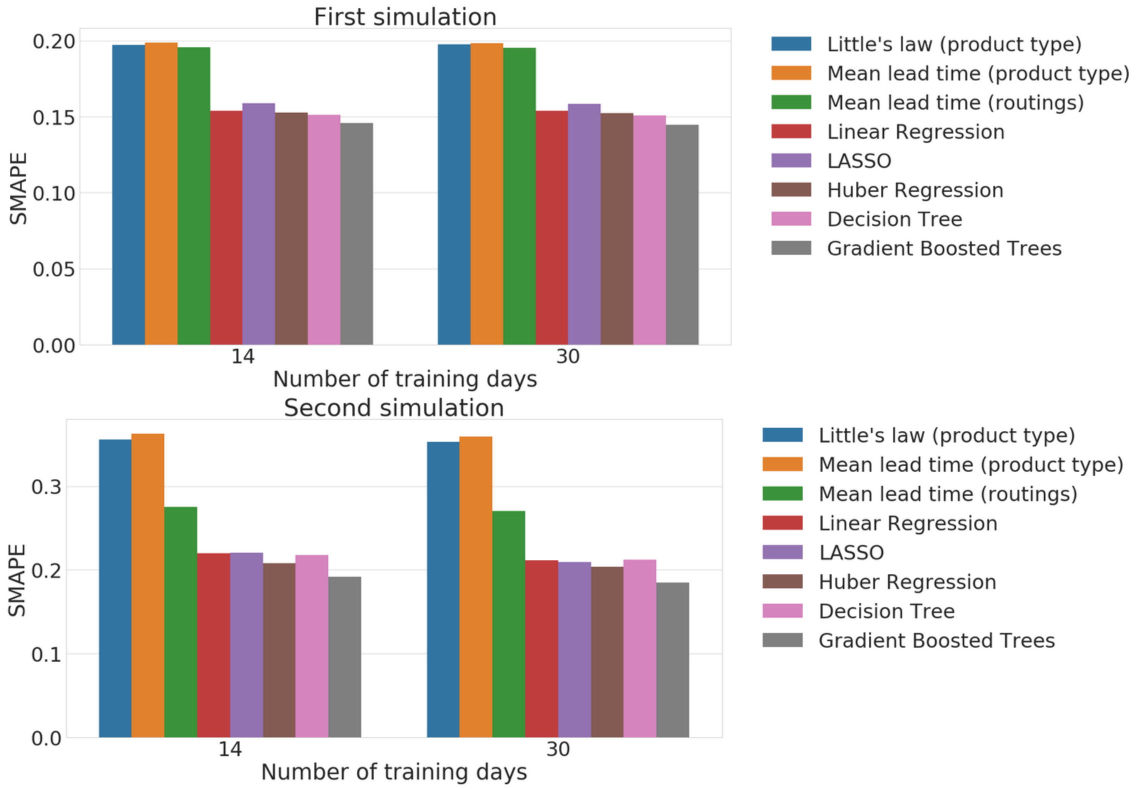

Figure 4 – Prediction error of several baseline and machine learning models for both simulations. Methods were trained on the first 14 or 30 days respectively.

So far, we observed lead time prediction in different simulations with predefined training and testing intervals. Additionally, we also would like to compare the predictive power of the previously described methods when the production environment changes unexpectedly. We model this concept drift by combining 10-10 days from the first then from the second simulation. We present each model in two different settings:

A. **Static:** training only on the first 7 days (Day 1-7)

B. **Adaptive:** daily re-training of the model, always using the previous 7 days

The prediction error (SMAPE) is evaluated for each day from Day 8 to Day 20. On the one hand, after the concept drift on Day 11, static models become completely ineffective (Figure 5). On the other hand, tree based methods (Decision Tree, Gradient Boosted Trees) quickly adapt to the new circumstances, even faster than the linear models (Figure 6). Linear models stabilize only after Day 18, when the training data no longer includes jobs from the first simulation.

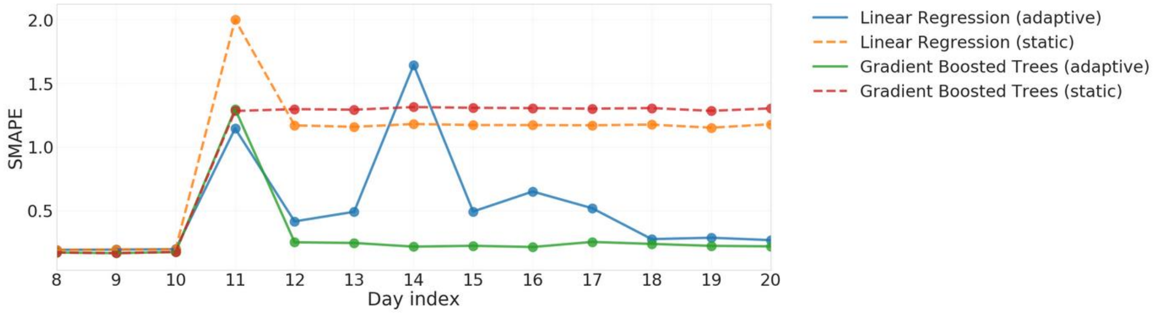

Figure 5 – Comparison of static and adaptive model behavior after the concept drift on Day 11.

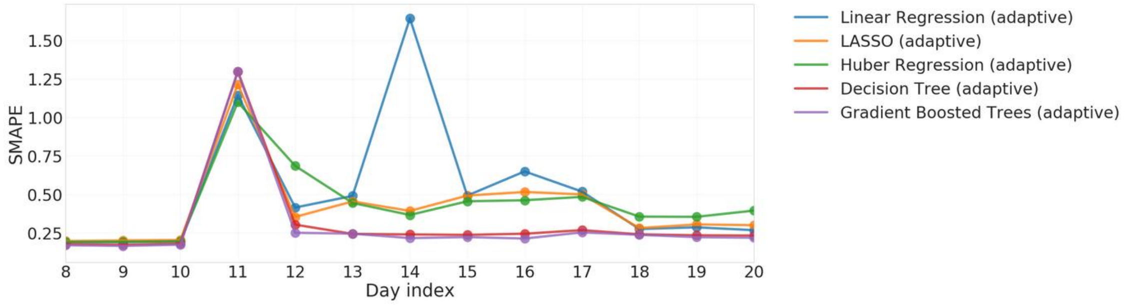

Figure 6 – Prediction error of adaptive models after the concept drift on Day 11.

Application of the method in production management decisions

Accurate prediction of lead times in production management and control is of crucial importance in increasing overall effectiveness, as several different decisions relate to it, including allocated human manpower, or assigned due dates. Therefore, in order to prevent late job completions, and idle, underutilized capacities, decision makers in production control need to apply efficient tools for prediction, as dynamics of recent production processes is higher than ever before. However, it is a great advantage of today's production systems that they are capable of providing detailed data about the processes, enabling to apply novel lead time prediction tools, as introduced in the paper.

The proposed method is developed so as so apply it in a daily production control decisions, as a complementary, add-on tool of business intelligence or MES systems. Although the tool directly supports lower level control decisions, it is also capable of providing high level, managerial decisions, with data about lead time related statistics as important elements corporate KPIs, e.g. service level, work-in-progress, or OEE related metrics like performance. As the tool utilizes latest production data, it can be applied to evaluate different control logics, job prioritization modes and dispatching rules that have significant impact on KPIs. In this way, production managers can always obtain an up-to-date and comprehensive picture of the efficiency production control decisions.

Conclusions

In the paper, a novel, data analytics and machine learning based method was proposed to predict lead times in a dynamic production environment. We demonstrated that the method is capable of making accurate predictions in near real time, even in the case of a highly non-stationary system. We tested several linear regression and tree based methods and found that boosted trees give lowest error while also capable of interpreting the results and selecting the most relevant features that influence the lead time. Our experiments were conducted on simulation-generated data, however, the method can be applied in any flow-shop environment that is capable of streaming online production log data. As stated, the method can be applied directly to support production control decisions on a daily basis, moreover, managerial implications can be also derived by obtaining a comprehensive picture about the effect of control decisions on the change of higher level KPIs. As for the future work, we plan to evaluate the results in a production environment, deploying the proposed models and integrating them with the corporate and shop-floor IT systems.

References

Chen, Tianqi and Guestrin, Carlos (2016), “Xgboost: A scalable tree boosting system”, *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 785-794.

Cox, DR (1958). "The regression analysis of binary sequences (with discussion)". *Journal of the Royal Statistical Society: Series B*, Vol 20, pp. 215–242.

Freund, Y., Schapire, R. E. (2017), “A decision-theoretic generalization of on-line learning and an application to boosting”, *Journal of Computer and System Sciences*, Vol 55.1 pp. 119–139.

Helu, M., Hedberg, T., Barnard F. and Allison B. F. (2017), “Reference architecture to integrate heterogeneous manufacturing systems for the digital thread”, *CIRP Journal of Manufacturing Science and Technology*, Vol. 19, pp. 191–195.

Little, J.D. (2011), “OR Forum-Little's law as viewed on its 50th anniversary”, *Operations research*, Vol. 59, No. 3, pp. 536-549.

Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., Sauer, O., Schuh, G., Sihn, W. and Ueda, K. (2016), “Cyber-physical systems in manufacturing”, *CIRP Annals - Manufacturing Technology*, Vol 65, No. 2, pp. 621-641.

Rosen, R., Wichert, G., Lo, G. and Bettenhausen Kurt D. (2015), “About The Importance of Autonomy and Digital Twins for the Future of Manufacturing”, *IFAC-PapersOnLine*, Vol. 48, No. 3, pp. 567–572.

Rousseeuw, Peter J., and Annick M. Leroy (2005), *Robust regression and outlier detection*, John Wiley & Sons, Vol. 589.

Safavian, S. R. and D. Landgrebe (1998), “A Survey of Decision Tree Classifier Methodology”, *IEEE Transactions on Systems, Man and Cybernetics*, Vol 22, pp. 660–674.

Tibshirani, Robert (1996). “Regression Shrinkage and Selection via the lasso”. *Journal of the Royal Statistical Society*. *Series B (methodological)*, Vol. 58. No. 1, pp. 267-288.