Performance Analytics by Means of the M5P Machine Learning Algorithm

Markus Fiedler
Blekinge Institute of Technology
Dept. of Technology and Aesthetics
Karlshamn, Sweden
markus.fiedler@bth.se

***Abstract*—Machine Learning (ML) has shown its capability to analyse, classify, and make predictions based on, large data sets, amongst others by means of decision trees. Network performance analysis and evaluation, on the other hand, focuses on finding and expressing qualitative, quantitative and preferably formal relationships between performance parameters. Due to the potential complexity of the latter, approximations that highlight main contributions and their orders of magnitude are of specific interest. Thereby, different parameter sub-spaces may imply different kinds of dependencies, sensitivities and approximation formulae, e.g. saturation, asymptotic and disruptive behaviours. Given such challenges, this work demonstrates the ability of the ML algorithm M5P to perform performance analytics by identifying approximations, together the applicable parameter sub-spaces, in a robust one-strike approach, where the detailed investigation of the obtained model trees provides valuable insights. We present, investigate and discuss a set of examples, spanning from impacts of parameters on user ratings, via modelling of user ratings over time, to post-analysis of analytically obtained performance results for approximations and asymptotic behaviour.**

***Index Terms*—Performance evaluation; model tree; approximation formulae; multivariate analysis; stochastic fluid flow model; Quality of Experience (QoE); Mean Opinion Score (MOS); user ratings**

I. Introduction

The International Teletraffic Congress (ITC) "witnessed the evolution of communications and networking: the influence of computer science on telecommunication", "documented this evolution with contemporary measurement studies, performance analyses of new technologies ..." and "contributed to the methodological toolbox of network scientists" [1]. Performance models, expressing mathematical, quantitative and formal relationships between technical and user-oriented system parameters, have been playing a major role in this evolution.

However, as systems and monitoring data grow in complexity and volume, versatile and yet insightful models (such as the Erlang model for telephony networks) have become increasingly difficult to develop and interpret. Though expressed by closed formulae, many models presented at [1] require a high degree of mathematical expertise to be understood, implemented and interpreted, which may hamper their adoption by less theoretically oriented networking professionals. Indeed, approximations that capture contributions by certain key parameters and related orders of magnitude, which may differ for various parameter sub-spaces, might be beneficial to many dimensioning, evaluation, optimisation and even standardisation tasks at hand. In view of this need, this paper focuses on a corresponding extension of the "methodological toolbox of network scientists" [1] by taking advantage of *Machine Learning* (ML) for producing interpretable approximations, which we henceforth call *performance analytics*.

Indeed, we observe a strong trend to apply Artifical Intelligence (AI) and ML in the area of increasingly complex communication networks and their operations [2], [3], [4], *Big data analytics* – mining and analysing huge amounts of monitoring data – is a key enabler, and is divided into descriptive, predictive and optimization analytics [5]. While the latter two types focus on the discovery of conditions to act upon and making efficient (and preferably optimal) decisions at the right time, *descriptive analytics* addresses past data patterns and probability analysis, with powerful data mining, classification and ML tools at hand. Our adopted notion *performance analytics* – not to be confused with "high performance analytics" [5], [6], classifying analytics performance as such – establishes the link between (classical) performance modeling, analysis and evaluation on one side, and ML tools of benefit for performance studies on the other side. Regarding the latter, we will focus on the specific ML algorithm "M5P" [7], [8] as an analytics tool providing trees of approximative models on the other side.

The contribution of this paper consists in demonstrating the ability of the ML algorithm M5P to perform performance analytics by identifying approximative performance models, together the applicable parameter sub-spaces, in a robust one-strike approach. Thereby, the detailed investigation of the M5P-generated model trees provides valuable insights on dependencies between performance parameters in different parts of the underlying parameter space. We analyse (a) user ratings from the Quality of Experience (QoE) domain from earlier work [9], [10], and (b) performance results from an analytical model in order to reveal the algorithm's capabilities to discover (verifiable) asymptotic behaviours and their underlying parameter sub-spaces, in both uni- and multivariate settings, where the idea to evaluate the algoritm against known cases dates back to [7]. Rather than advancing the areas of performance evaluation or that of ML classification and prediction performance as such, we combine both of them

in order to enhance the "methodological toolbox of network scientists" [1].

Section II provides a short introduction to machine learning in the context of performance evaluation, including related work. Section III focuses on the description, notation, generation and validation of model trees in the context of the task at hand. Section IV reviews recently obtained performance analytics results in the area user QoE ratings. Section V investigates the ability of M5P to track numeric performance results with focus on asymptotic behaviour, and Section VI demonstrates its applicability to multivariate settings. Section VII concludes and provides an outlook to future work.

II. Machine Learning and Performance Evaluation

Performance and its evaluation in the area of ML typically addresses high-performance and real-time analytics [5], [6], focusing on the capabilities of the algorithms themselves. In the application area of network-related performance evaluation, ML algorithms have amongst others been used for data mining and classification [11], [12], [13], quality prediction and management [9], [14], [15] and tuning of performance models [16]. In the latter case, deep learning was used to develop a heuristic for the bounds for deterministic network calculus. Borchert et al. [14], for instance, compare several approaches for correlating QoE and technical parameters with focus on the classification performance. Torres and Liotta [15] discuss the feasibility of un-/supervised learning methods for cognitive no-reference video quality models.

However, besides of [9], [10], [17], we are not aware of any works that explicitly show details of performance models that are *generated* by ML. Before discussing them, we provide a short introduction into various kinds of *decision trees* that represent structured regression models that predict the value of a target based on one or several input variables, thereby implementing a tree of decisions. Leaves represent class labels, and nodes or branches represent conjunctions of features that lead to class labels. Though a commonly used tool in data mining, decision trees are widely used in ML for classification purposes [5]. Decision trees are learned during training phases and evaluated during test phases.

Random Forests (RF) denote a collection of decision trees, over which averaging is performed in order to reduce the risk of overfitting [18]. The corresponding algorithm has amongst other been applied successfully in the area of QoE modeling. The ITU-T Recommendation P.1203.3 [17] contains a machine-learned RF model in combination with a traditional parametric model.

Model trees are specific decision trees with linear regressions at the leaf nodes, instead of constant values as in *regression trees* [7]. The M5P algorithm [8] is an extended version based on the M5 algorithm that was originally developed by R. Quinlan [7]. They have shown to be able to handle large data sets with many dimensions and partially missing data, and have been applied successfully in a range of areas, e.g. road traffic [19], localisation issues [20], [21] or supply challenges [22], [23]. In [9], the M5P algorithm demonstrated its superiority to other ML algorithms while providing insights into the set of parameters affecting video playout performance. Our own work [10] revealed the potential of M5P to generate descriptive analytics of instantaneous QoE over time, which even allowed for confirming earlier modeling assumptions. The latter two approaches will be detailed in Sections IV-A and IV-B.

III. Model Trees

This section introduces the description of model trees to be used in the sequel, discusses the use of the M5P algorithm, and comments on the validation of the obtained results.

A. Notation

The *decision rules* that span the model tree define a set of spaces (in the sense of multi-dimensional intervals) $\mathcal{X}_i$, stretching from a (usually exclusive) lower bound

$$\vec{x}_i^{\,\mathrm{lo}} = \inf\{\vec{x} \in \mathcal{X}_i\} \tag{1}$$

to an (inclusive) upper bound

$$\vec{x}_i^{\,\mathrm{up}} = \max\{\vec{x} \in \mathcal{X}_i\} \tag{2}$$

This notation implies the possibility for several variables, a.k.a. *features*, to change within the same sub-space $\mathcal{X}_i$ of the parameter or feature space, over which a set of *linear models* is constructed by the M5P algorithm:

$$\begin{aligned} y_i(\vec{x}) &= a_i + \vec{b_i}^{\mathbf{T}}\,\vec{x} \quad \forall \vec{x} = [x_1 \dots x_k] \in \mathcal{X}_i \\ &= a_i + \sum_{j=1}^{n} b_{i,j}\, x_j \end{aligned} \tag{3}$$

As input, the M5P algorithm needs a matrix with n columns containing the features x_j, and an additional column with the results y to be modeled. Experience in the context of [10] has shown that it is highly important to exclude any unwanted features x_j, such as the user ID when the task at hand is to model Mean Opinion Scores (MOS), in order to prevent the algorithm of taking them into account.

B. Data Preparation and Transformations

In order to capture different types of effects by parameter values or by their orders of magnitude, we can extend the M5P-supported linear approximations through transformations on abscissae (feature value/order of magnitude) and/or ordinate (resulting value/order of magnitude) as follows[1]:

- *Logarithmic* (order of magnitude w.r.t. base B on abscissae; additive relationship)

$$y_i(\vec{x}) = a_i + \sum_{j=1}^{k} b_{i,j}\, \log_B(b(x_j)) \tag{4}$$

[1]The following expressions apply to any base B. While for some relationships the natural base $B = e$ works best, other contexts may benefit from relating to the order of magnitude, i.e. $B = 10$.

- *Exponential* (order of magnitude w.r.t. base B on ordinate; multiplicative relationship)

$$\log_B(y_i(\vec{x})) = a_i + \sum_{j=1}^{k} b_{i,j}\, x_j \quad (5)$$

$$y_i(\vec{x}) = B^{a_i} \prod_{j=1}^{k} B^{b_{i,j}\, x_j} \quad (6)$$

- *Power* (order of magnitude w.r.t. base B on all axes; multiplicative relationship)

$$\log_B(y_i(\vec{x})) = a_i + \sum_{j=1}^{k} b_{i,j}\, \log_B(x_j) \quad (7)$$

$$y_i(\vec{x}) = B^{a_i} \prod_{j=1}^{k} {x_j}^{b_{i,j}} \quad (8)$$

The in-data for M5P (features on the abscissae, results to be approximated on the ordinate) needs to be transformed, and the results from (3) to (8) need to be interpreted accordingly, which will be demonstrated in Sections IV to VI.

C. Implementation

We obtain the model tree from applying the Weka [24] M5P implementation, with the following settings:

- minimal number of instances per leaf = 1
- number of decimal places = 8
- batch size = 100
- all other options (useUnshoothed, unpruned, saveInstances, doNotCheckCapabilities, debug, buildRegressionTree) = false

This M5P implementation uses the whole data set as training set for constructing the reported model tree, which is natural to the descriptive analytics problem at hand, for which overfitting is not an issue. However, for purposes of validation, different training and test sets may apply, which will be discussed in the following subsection.

D. Validation

Besides the numbers of instances and rules, as well as the run time of the algorithm, Weka reports a set of statistics:

- correlation coefficient R
- mean absolute error (MAE)
- root mean square error (RMSE)
- relative absolute error (RAE)
- root relative squared error (RRSE)

With our application in mind, we focus on the squared error measures (RMSE, RRSE) besides of the correlation coefficient R.

Above statistics depend on the selected *validation method*. For our application, we are facing a set of alternatives.

1) Evaluation on the *training set*: The statistics relate to the whole data set, which seems to be a natural choice for our kind of matching task. However, it does not provide indications on how the algorithm would perform with different data sets.
2) Evaluation using a *percentage split*: The available data is split into a training set and a test set, e.g. applying a 50:50 ratio.
3) Evaluation using *k-fold cross-validation*: In this case, the data is split in k sets. In each of the resulting k cases, one set serves as test set for the tree constructed using the remaining $k-1$ sets. The k sets of statistics are then averaged in order to obtain a less overfitting-prone view on the performance of the data set.

The original papers [7], [8] recommend $k = 10$, which has also been used by Casas and Wassermann [9]. Thus, 10-fold cross-validation will be applied in the sequel, if not stated otherwise. Indeed, the study [10] has revealed rather small differences between the different validation methods for the matching task at hand.

IV. Analytics of Measurement Results

In this section, two applications of the M5P from the recent literature in the QoE area are presented and discussed in short.

A. Example 1: Mobile Video QoE Prediction

Casas and Wassermann [9] address ML-based mobile video QoE prediction. They identified a set of features, out of which M5P picked the following subset:

1) relative stalling time
2) number of stallings
3) relative initial delay
4) average stalling duration
5) elapsed time since the end of the last stalling event till the end of the video
6) video frames per second

M5P produced a small model tree, consisting of one decision regarding the number of stalls (smaller or larger than the threshold value 1.5) and two corresponding linear models, cf. (3), with different, unfortunately unspecified coefficients. Instead, the paper highlights that M5P outperformed the other ten approaches, amongst others Support Vector Machines (VM), RF, bagging tree and pace regression.

B. Example 2: Modeling of Instantaneous QoE

Our own predecessor study [10] investigated M5P's capability of modeling instantaneous QoE over time, yielding approximations of Mean Opinion Scores in face of quality variations over time. To this end, we used the "LIVE Mobile Stall Video Database II" [25], containing user ratings by 54 subjects and 176 videos with 26 stalling pattern distortions. Every test video trace consists of continuous time subjective user ratings per frame of 27 users. The subjects provided 30 *opinion scores* (OS) per second via a rating device.

For sake of illustration, Figure 1 shows a set of user ratings (OS) of one particular video, and Figure 2 shows the corresponding piecewise linear matchings according to (3):

$$y(x) = a_i + b_i\, x \quad \forall x \in \mathcal{X}_i \quad (9)$$

where x represents time, in comparison to the (instantaneous) MOS. Obviously, both are in very good agreement. M5P

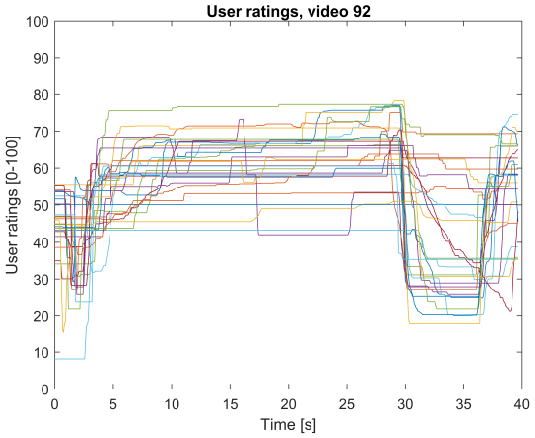

Fig. 1: Illustrative example of ratings of video 92 by 27 users, from [10].

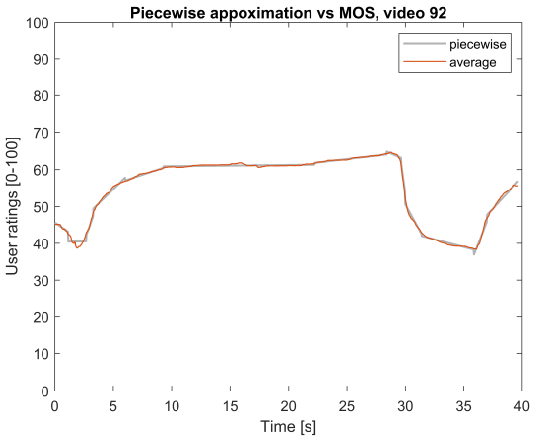

Fig. 2: Piecewise linear instantaneous QoE models, together with MOS values, in case of video 92, from [10].

actually managed to catch the *moments of change*, which are seen from locally extreme values of the b_i coefficient in Table I, with some short delay of about one second [10].

It has to be noted that the exponential transformation (6) cannot be used straight away, as there are non-vanishing y offsets. However, [10] shows that, after removal of the offsets, even the piecewise linear approximations follow exponential trends. In case of the first rise, the slope is matched by $\exp(-0.31t/\mathrm{s})$ from $t \simeq 2.7\,\mathrm{s}$ on with a correlation coefficient $R^2 \simeq 0.93$, while the first fall exhibits a slope of $\exp(-0.53t/\mathrm{s})$ from $t \simeq 29.7\,\mathrm{s}$ on with a correlation coefficient $R^2 \simeq 0.96$. Thus, the piecewise linear approximations allow for further analysis and confirm earlier findings [26].

TABLE I: Excerpt of the model tree for video 92, based on 32481 instances, with 13 rules [10].

i	x_i^{lo}	x_i^{up}	a_i	b_i	Remark
:	:	:	:	:	:
2	1.139 s	2.689 s	40.4038	0.0400/s	
3	**2.689 s**	3.284 s	18.3345	**8.8662/s**	First rise
4	3.284 s	6.022 s	39.3054	3.0826/s	
:	:	:	:	:	:
8	28.396 s	29.651 s	103.6067	–1.3667/s	
9	**29.651 s**	29.980 s	737.4112	**–22.7949/s**	First fall
10	29.980 s	31.399 s	224.5945	–5.8006/s	
:	:	:	:	:	:

C. Insights

When modeling data stemming from measurements (in our case assessment of user ratings of QoE), the M5P has shown to be able to pick the features of greatest importance [9] and to provide matching decent matching results, which includes the identification of suitable intervals as well as corresponding piecewise linear models, even in view of rather noisy data [10]. Furthermore, the obtained piecewise linear models allow for further analysis steps [10]. We can conclude that the M5P is a tool that is well capable of analysing *measurement* results for dependencies and parameter sub-spaces in a one-strike approach.

V. Analytics of Performance Results

We are now turning towards the analytics of results provided by a performance model, in the sense of understanding trends and asymptotic behaviours. We will examine the corresponding capabilities of M5P, both in a univariate (in this section) and a multivariate scenario (in Section V).

To this end, we use results from an analytic performance model, as reference values are available (cf. the approach in [7]). We provide details such that the underlying results can be re-generated in a straightforward manner for validation purposes. While describing the model in the next subsection, we have included the underlying performance analysis steps in the Appendix, in order to facilitate the calculation of any reference values.

A. Stochastic Fluid Flow Model

We assume a queuing system with a constant inflow (e.g. video) and a variable channel rate that is typical for wireless systems. Once the sender-side buffer fills up, data is lost and cannot be displayed at the receiver side, which in case of video implies freezing. We furthermore assume that the data flows in such a voluminous and regular manner that a stochastic fluid flow model can be used. Using Kendall-like notation, we describe the system as an **RP/**Π**IRP/**K system:

1) Source with Constant-**R**ate **P**rocess, rate $R = r$.
2) Channel with variable capacity $C = c > r$ during its on-phase, and $C = 0$ during its off phase(s). Its capacity dynamics are modelled by two independent/superposed **I**nterrupted **R**ate **P**rocesses (a.k.a. on-off processes):
   a) One slow *shadowing process* with exponential state durations [27], parameterised by average on time $\mathbf{E}T_{\mathrm{on},1}$ and average off time $\mathbf{E}T_{\mathrm{off},1}$.
   b) One fast *automatic repeat request (ARQ) process* with exponential state durations [28], parameterised by average on time $\mathbf{E}T_{\mathrm{on},2}$ and average off time $\mathbf{E}T_{\mathrm{off},2}$.

TABLE II: Feature and parameter list for the RP/ΠIRP/K system

Feature	Parameter	Min.	Max.	Step	Note
x_1	$\mathbf{E}T_{\mathrm{on},1}$		1 min		
x_2	$\mathbf{E}T_{\mathrm{off},1}$		60 ms		Shadowing
x_3	$\mathbf{E}T_{\mathrm{on},2}$		1 ms		
x_4	$\mathbf{E}T_{\mathrm{off},2}$		1 ms		ARQ
x_5	k	0 ms	240 ms	1 ms	241 values
y	P_{L}		as calculated		

TABLE III: Application of M5P to the RP/ΠIRP/K system with different underlying transformations

Type	x_i	y	Instances	Rules	R	RMSE	RRSE
linear	lin	lin	241	3	0.2721	0.0335	95.6 %
exp.	lin	$\log_{10}$	241	4	0.9611	0.1823	23.1 %
log.	$\log_{10}$	lin	240	3	0.6508	0.0104	76.9 %
power	$\log_{10}$	$\log_{10}$	240	11	0.9907	0.083	13.5 %

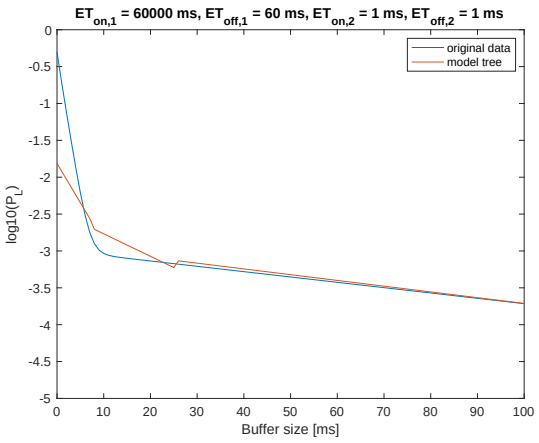

Fig. 3: Loss probability versus buffer size.

The rates of these server-side processes multiply ("Π") each other [29] in the way that only if both IRPs are in the corresponding on phase, the channel is available ($C = c$). As soon as one of the sub-IRPs is off, the channel is unavailable ($C = 0$), which entails buffering at the sender side.

3) Buffer of limited size K, entailing loss in case of overflow.

Two-state RP/IRP(/K) systems have been addressed before, a.o. in [30], [31]. Here, we are extending the analysis towards two processes, yielding four states, in order to be able to investigate two asymptotes, stemming from different dynamics.

The fluid system scales in both data units and time units, which means that suitable units can be chosen. We henceforth use one millisecond (ms) as time unit, and scale the data unit such that $r = 1$ data unit/ms. For sake of convenience, we specify the buffer size in time units:

$$k = \frac{K}{r} \tag{10}$$

From the fluid flow analysis, which is found in Appendix A for the four-state RP/ΠIRP/K system and in Appendix B for the two-state RP/IRP/K system, we obtain the loss probability P_{L} from to (23) and (30), respectively.

B. Analytics of an RP/ΠIRP/K System

We assume the following parameters as shown in Table II. Naturally the shadowing process acts on a much slower time scale than the ARQ process. We assume a capacity of $c = 10r$. Due to the chosen ARQ parameters, the effective channel capacity is upper-bounded to $c_{\mathrm{eff}} < 5r$) We use the four transformations described in Section III-B with $B = 10$, in order to highlight the switch from values to orders of magnitudes, and to investigate both additive and multiplicative relationships between parameters as shown in (3), (4), (6) and (8). This provides us with four datasets in total, to which we apply the M5P algorithm.

We first compare the overall performance, shown in Table III. In the *linear case*, the model tree is rather small (three rules), and the correlation coefficient is low. In the *exponential* case, the tree is slightly larger (four rules), with a high correlation coefficient and decent error values. In the *logarithmic* case (where $k = 0$ ms has to be excluded), the tree is of the same size as in the linear case (three rules), however with a higher correlation and reduced error values. Finally, in the *power* case (dito), the tree contains eleven rules and provides the best correlation and error performance. In this case, M5P manages to track the curve well, however at the expense of a rather detailed model tree.

TABLE IV: Model tree based on an exponential transformation (5) with $B = 10$

i	k_i^{lo}	k_i^{up}	a_i	b_i
1	0	7.5	-1.8129	-0.1066/ms
2	7.5	25.5	-2.4625	-0.0305/ms
3	25.5	126.5	-2.9316	-0.0078/ms
4	126.5	240	-2.9697	-0.0074/ms

Yet, it is interesting to see the combination of a small tree with a decent correlation and error performance in the exponential case (5), which is not surprising when considering the structure of (19) and the curve in Figure 3. Looking at the performance data, we can clearly distinguish two asymptotic behaviours, one in the area of small buffers towards the ordinate, and the other one for large buffer sizes. From Figure 3, we can see that the M5P-generated model tree shown in Table IV has some difficulties to capture the shape of the sharply bended curve belonging to the original data in the first two intervals. Still, it indicates a significant decline of the loss probability (about one order of magnitude per 10 ms) in the first interval, which reflects the dominating, but quickly decreasing impact of the ARQ. This is followed by a transition area (interval 2) with intermediate coefficients, stretching over the sharp bend, and two further approximations with very similar coefficients ($a_3 \simeq a_4$, $b_3 \simeq b_4$), which point at asymptotic behaviour with a rather flat decline as compared to intervals 1 and 2. This latter part is due to the shadowing that appears seldom –the approximations would

TABLE V: Feature and parameter list for the RP/IRP/K system modeling the shadowing

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\mathrm{on},1}$	1 min			
x_2	$\mathbf{E}T_{\mathrm{off},1}$	60 ms			Shadowing
x_3	k	0.01 ms	1000 ms	$10^{0.1}$	51 values
y	$\ln(P_{\mathrm{L}})$	as calculated			

hit a non-logarithmic ordinate at $10^{-2.9316} = 0.00117$ and $10^{-2.9697} = 0.00107$ – but has a long after-effect due to the rather long average off time of 60 ms.

The rather mediocre performance in the first two intervals can be attributed to a rather sparse data grid. For instance, increasing the resolution in k by one order of magnitude gives a much better matching behaviour. Varying k from 0 ms to 120 ms in steps of 0.1 ms yields a rather detailed model tree with 19 rules, $R = 0.9975$, RMSE $= 0.0468$ and RRSE $= 9.0\,\%$, which is not shown for sake of brevity.

C. Analytics of the Asymptotes

We now investigate the asymptotes each for itself, i.e. two RP/IRP/K subsystems. In order to be able to compare directly with (27), we switch the base of (6) to $B = e$.

1) Shadowing Subsystem: We start with the subsystem that is modeling the shadowing behaviour with parameters according to Table VI. The model tree obtained from M5P has only one rule:

$$\ln(P_{\mathrm{L}}) = -6.9088 - 0.0167/\mathrm{ms} \cdot k \quad (11)$$

$$P_{\mathrm{L}} \simeq 0.001 \exp(-0.0167/\mathrm{ms} \cdot k) \quad (12)$$

with $R \simeq 1$, RMSE $= 0$ and RRSE $= 0.0009\,\%$, indicating a close-to-perfect fit. Thereby, the value $b_1 = -0.0167/\mathrm{ms}$ matches the non-vanishing (dominant) Eigenvalue z_{dom}, taking the normalisation (10) with $r = 1$ data unit/ms into account.

2) ARQ Subsystem: We now turn our attention towards the subsystem modeling the ARQ, with parameters according to Table VI. In this case, the M5P produces three rules, as shown in Table VII, with $R \simeq 1$, RMSE $= 0.0388$ and a RRSE $= 0.187\,\%$. The second and third rule again point at the existence of an asymptote from $k \simeq 5.6$ ms on towards larger buffer sizes, with coefficients b_2 and b_3 close to the non-vanishing Eigenvalue $z_{\mathrm{dom}} \simeq -0.8889/$data unit, while there is a different behaviour seen for the first interval, with a slightly steeper gradient.

TABLE VI: Feature and parameter list for the RP/IRP/K system modeling the ARQ

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\mathrm{on},2}$	1 ms			
x_2	$\mathbf{E}T_{\mathrm{off},2}$	1 ms			ARQ
x_3	k	0.01 ms	100 ms	$10^{0.1}$	41 values
y	$\ln(P_{\mathrm{L}})$	as calculated			

TABLE VII: Model tree for the ARQ subsystem, based on the exponential transformation (5) with $B = e$

i	k_i^{lo}	k_i^{up}	a_i	b_i
1	0.010	5.661	-0.7226	-0.9072/ms
2	5.661	35.717	-0.7818	-0.8882/ms
3	35.717	100.000	-0.7688	-0.8881/ms

D. Insights

From above cases, we can see that, given a sufficient amount of data and the right kind of transformation employed, the M5P is able to analyse numerical results for asymptotic behaviour, including the identification of areas for which the asymptotic behaviour applies. It is recommended to test different transformations and resolutions in the feature space, with both additive and multiplicative spacings of values.

VI. Analysis of Multivariate Cases

So far, we have been investigating the dependency of a single parameter, the buffer size k. Indeed, the M5P algorithm is not limited to one single dependency. Thus, this subsection, we study cases in which several parameters are varied. We confine ourselves to the RP/IRP/K case.

A. Variation of Average On Time and Buffer Size

We revert to the base $B = 10$ (in order to obtain orders of magnitude) and vary the parameters as stated in Table VIII. Based on the corresponding data, the M5P reports the model tree shown in Table IX, with $R = 0.9281$, RMSE $\simeq 2.85$ and RRSE $\simeq 37.2\,\%$, i.e. larger errors as seen so far. Looking at the model tree shown in Table IX, we observe that there is a certain dependency on the average on time until the buffer gets about one order of magnitude larger than the average off time. In case of an even larger buffer, the average on time looses its influences on both decision and approximation, i.e. the asymptotic behaviour is purely dominated by the buffer size (as seen from rule 3).

TABLE VIII: Feature and parameter list for an RP/IRP/K system with varied average on time and buffer size

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\mathrm{on}}$	1 ms	10 s	$10^{0.1}$	41 values
x_2	$\mathbf{E}T_{\mathrm{off}}$	1 ms			
x_3	k	0.01 ms	100 ms	$10^{0.1}$	41 values
y	$\log_{10}(P_{\mathrm{L}})$	as calculated			1681 values

TABLE IX: Model tree for varied average on time and buffer size, based on the exponential transformation (5) with $B = 10$ ($b_{i,2} = 0$)

i	Bounds for $\mathbf{E}T_{\mathrm{on}}$	Bounds for k	a_i	$\frac{b_{i,1}}{\mathrm{ms}^{-1}}$	$\frac{b_{i,3}}{\mathrm{ms}^{-1}}$
1	≤ 112.946 ms	≤ 11.295 ms	-0.7221	-0.0168	-0.4233
2	> 112.946 ms	≤ 11.295 ms	-2.5871	-0.0002	-0.4326
3	any	> 11.295 ms	-6.2873	$\simeq 0$	-0.2447

TABLE X: Feature and parameter list for an RP/IRP/K system with varied average off time and buffer size

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\text{on}}$		100 ms		
x_2	$\mathbf{E}T_{\text{off}}$	1 ms	100 ms	$10^{0.1}$	21 values
x_3	k	1 ms	100 ms	$10^{0.1}$	21 values
y	$\log_{10}(P_{\text{L}})$	as calculated			441 values

B. Variation of Average Off Time and Buffer Size

We assume a constant average on time of 100 ms and the remaining parameters as shown in Table X. The application of the M5P results in a model tree with 19 rules, with $R = 0.9767$, RMSE $\simeq 0.93$ and RRSE $\simeq 22.3\,\%$. The model tree, which is omitted for sake of brevity, consists of approximation formulae and the decision rules with both $\mathbf{E}T_{\text{off}}$ and k involved, from which no regular behaviour can be seen.

C. Variation of Average On and Off Times

We assume a constant buffer size of 100 ms and the remaining parameters as shown in Table XI. The model tree shown in Table XII, with $R = 0.9005$, RMSE $= 4.1$ and RRSE $= 43.5\,\%$, consists of four rules, with a similar matching performance as in the first multivariate case. The decisions in the tree only depend on the average off time, while the average on time hardly contributes to the approximations. Thus, the performance is dominated by the average off time and its relation to the (constant) buffer size. The smaller the average off time, the more effect the buffer has, the less the loss and the steeper the decline becomes.

D. Variation of Average On, Off Time and Buffer Size

Our final example addresses the variation of all three features, which is reported in Table XIII. The model tree shown in Table XIV, with $R = 0.954$, RMSE $\simeq 1.7$ and RRSE $\simeq 30\,\%$, consists of 74 intervals. When investigating that model tree, it becomes obvious that quite a lot of neighbouring intervals share the same coefficients $b_{i,1}$ and $b_{i,2}$. This occurs when rather short average off times coincide with rather large buffer sizes, which is the typical asymptotic regime of this kind of system.

TABLE XI: Feature and parameter list for an RP/IRP/K system with varied average on time and average off time

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\text{on}}$	100 ms	10 s	$10^{0.1}$	21 values
x_2	$\mathbf{E}T_{\text{off}}$	1 ms	100 ms	$10^{0.1}$	21 values
x_3	k		100 ms		
y	$\log_{10}(P_{\text{L}})$	as calculated			441 values

TABLE XII: Model tree for varied average on and off times, based on the exponential transformation (5) with $B = 10$ ($b_{i,3} = 0$)

i	Bounds for $\mathbf{E}T_{\text{off}}$	a_i	$\frac{b_{i,1}}{\text{ms}^{-1}}$	$\frac{b_{i,2}}{\text{ms}^{-1}}$
1	$[1\,\text{ms}, 7.126\,\text{ms}]$	-28.0964	0	3.3202
2	$]7.126\,\text{ms}, 11.295\,\text{ms}]$	-10.6264	-0.0002	0.4440
3	$]11.295\,\text{ms}, 22.536\,\text{ms}]$	-7.788	-0.0002	0.1967
4	$]22.536\,\text{ms}, 100\,\text{ms}]$	-4.0067	-0.0002	0.0361

TABLE XIII: Feature and parameter list for an RP/IRP/K system with varied average on time, off time and buffer size

Feature	Parameter	Min.	Max.	Factor	Note
x_1	$\mathbf{E}T_{\text{on}}$	1 ms	100 ms	$10^{0.1}$	21 values
x_2	$\mathbf{E}T_{\text{off}}$	1 ms	100 ms	$10^{0.1}$	21 values
x_3	k 10 ms	100 ms	$10^{0.1}$	11 values	
y	$\log_{10}(P_{\text{L}})$	as calculated			4851 values

TABLE XIV: Model tree for varied average on and off times and buffer size, based on the exponential transformation (5) with $B = 10$, a_i and $b_{i,1}$ not shown (for sake of brevity)

i int.	Bounds for $\mathbf{E}T_{\text{off}}$ [ms]	Bounds for k [ms]	$\frac{b_{i,2}}{\text{ms}^{-1}}$	$\frac{b_{i,3}}{\text{ms}^{-1}}$
3–6	$[1, 1.422]$	$]14.219, 17.901]$	3.0003	-0.1337
9–12	$[1\,1.129]$	$]17.901, 28.371]$	3.3962	-0.2561
13–16	$]1.129, 1.422]$	$]17.901, 28.371]$	3.3962	-0.2374
17–20	$[1, 1.422]$	$]28.371, 35.717]$	5.5299	-0.1337
21–24	$]1.422, 1.790]$	$]17.901, 28.371]$	1.8721	-0.1793
27–30	$]1.422, 1.790]$	$]28.371, 35.717]$	2.7299	-0.1033
52–57	$]2.837, 3.572]$	$]71.264, 100]$	1.1246	-0.0762

E. Insights

From the model trees of the multivariate cases, we observe that the average off time $\mathbf{E}T_{\text{off}}$ often appears in conjunction with the buffer size k, where $k/\mathbf{E}T_{\text{off}} \gg 1$ fosters asymptotic behaviour. On the other hand, it also clarifies that the average on time $\mathbf{E}T_{\text{on}}$ plays a minor role in the corresponding decisions and approximation formulae. In many cases, that feature is simply suppressed from the corresponding formulae. Altogether, the M5P is able to discover and re-confirm earlier knowledge from the domain of fluid flow on-off scenarios.

Especially when comparably large feature spaces spanning over several orders of magnitude are to be conquered, multiplicative spacing are highly recommended. In our work, we have made good experience with using 10 multiplicative steps per decade, i.e. a factor of $10^{0.1} \simeq 1.26$.

VII. Conclusions and Outlook

In this paper, we have investigated and demonstrated the capabilities of the M5P Machine Learning algorithm to assist with the analysis of performance measurement and evaluation results, which we call *performance analytics*. Hereby, the M5P algorithm is able to extract approximation formulae from a set of features, together with applicable sub-spaces, in a one-strike approach.

We first reviewed earlier work on analysing user ratings, which showed the capability of M5P to pick most important parameters out of a set of candidates on the one hand, and to follow and average over a noisy data set on the other hand. The latter included the discovery of points or moments of change from which on the approximation formula changes to a significant extent.

We then turned to the post-analysis of performance results from a fluid flow queuing model, in order to investigate to which extent M5P is capable of identifying and marking off

underlying trends such as asymptotes. We found that, with a sufficiently dense data set and upon a suitable transformation of the original feature space, M5P is able to provide indications of asymptotic behaviour, including identification of conditions under which it can be observed and selection of the most important features in a multivariate space. These indications are obtained from careful examinations of the model trees, their structures and related coefficients.

Encouraged by these observations, future work will address the application of M5P to data sets stemming from measurements and simulations (and potentially even from complex performance models) in order to identify underlying trends and their areas of applicability, as a new tool in the "methodological toolbox of network scientists" [1].

VIII. Acknowledgements

Big thanks go to Usha Kiran Chapala and Sridhar Peteti for the joint work on [10], to Niklas Lavesson (Jönköping University), Jörgen Gustafsson and David Lindero (both at Ericsson Research) for fruitful discussions, and to the Swedish KKS Foundation for their support of the "BigData@BTH" project (d-nr 2014-0032), under which this work was done.

References

[1] About ITC, https://itc31.org/en/about-itc.html [online; last seen 2019-04-03].

[2] M. Cooney, "Cisco: How AI and machine learning are going to change your network," Network World, Sept. 2018, https://www.networkworld.com/article/3305327/cisco-how-ai-and-machine-learning-are-going-to-change-your-network.html [online; last seen 2019-04-03].

[3] R. Casellas et al., "Enabling data analytics and Machine Learning for 5G services within disaggregated multi-layer transport networks,", in Proc. 2018 20$^{\text{th}}$ Int. Conf. on Transparent Optical Networks (ITCON), Bucharest, Romania. Sept. 2018.

[4] Y. Fu, S. Wang, C.-X. Wang, X. Hong, and S. McLaughlin, "Artificial Intelligence to manage network traffic of 5G wireless Networks," IEEE Network, vol. 32, issue 6, pp. 58–64, 2018.

[5] A. Osman, M. El-Refaey, and A. Elnaggar, "Towards real-time-analytics in the cloud," in Proc. 2013 IEEE Ninth World Congress on Services, Santa Clara, CA, June/July 2013.

[6] R.C. Maheshwar, D. Haritha, "Survey on high analytics of Bigdata with Apache Spark," in Proc. 2016 Int. Conf. on Advanced Communication Control and Computing Technologies (ICACCCT), pp 721–725, 2016.

[7] R. J. Quinlan, "Learning with continuous classes," in Proc. 5$^{\text{th}}$ Australian Joint Conference on Artificial Intelligence, Singapore, pp. 343–348, 1992.

[8] Y. Wang and I. H. Witten, "Induction of model trees for predicting continuous classes," in Poster Papers of the 9$^{\text{th}}$ European Conference on Machine Learning, 1997.

[9] P. Casas and S. Wassermann, "Improving QoE prediction in mobile video through machine learning," in Proc. 2017 8$^{\text{th}}$ International Conference on the Network of the Future (NOF), pp. 1–7, 2017.

[10] M. Fiedler, U. Chapala, and S. Peteti, "Modeling instantaneous Quality of Experience using machine learning of model trees," to be presented at QoMEX 2019, Berlin, June 2019.

[11] P. Casas, J. Mazel, and P. Owezarski, "MINETRAC: Mining flows for unsupervised & semi-supervised classification," in Proc. 23rd International Teletraffic Congress (ITC), San Francisco, CA, pp. 87–94, Sept. 2011.

[12] A. Morichetta, E. Bocchi, H. Metwalley, and M. Mellia, "CLUE: Clustering for mining web URLs," in Proc. 28th International Teletraffic Congress (ITC), Würzburg, Germany, pp. 286–294, Sept. 2016.

[13] A. Montieri, D. Ciuonzo, G. Aceto, and A. Pescapé, "Anonymity services Tor, I2P, JonDoym: Classifying in the dark," in Proc. 29th International Teletraffic Congress (ITC), Genua, Italy, pp. 81–89, Sept. 2017.

[14] K. Borchert, M. Hirth, T. Zinner, and D. Constantin, "Correlating QoE and technical parameters of an SAP system in an enterprise environment," in Proc. 28th International Teletraffic Congress (ITC), Würzburg, Germany, pp. 34–36, Sept. 2016.

[15] M. Torres Vega and A. Liotta, "Cognitive real-time QoE management in video streaming services," in Proc. 30th International Teletraffic Congress (ITC), Vienna, Austria, pp. 123–128, Sept. 2018.

[16] F. Geyer and G. Carle, "The case for a Network Calculus heuristic: Using insights from data for tighter bounds," in Proc. 30th International Teletraffic Congress (ITC), Vienna, Austria, pp. 43–48, Sept. 2018.

[17] ITU-T Recommendation P.1203.3, 'Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport - Quality integration module," International Telecommunication Union, January 2019, https://www.itu.int/rec/T-REC-P.1203.3/en [online; last seen 2019-04-03].

[18] T. K. Ho, "Random Decision Forests," in Proc. 3rd Int. Conf. on Document Analysis and Recognition, Montreal, QC, pp. 278–282, Aug. 1995.

[19] C. Zhan, A. Gan, and M. Hadi, "Prediction of lane clearance time of freeway incidents using the M5P tree algorithm," IEEE Trans. Intell. Transp. Syst., vol. 12, no. 4, pp. 1549–1557, Dec. 2011.

[20] K. K. Almuzaini and T. A. Gulliver, "Localization in wireless networks using decision trees and k-means clustering," in Proc. 2012 IEEE Vehicular Technology Conference (VTC Fall), pp. 1–5, 2012.

[21] P. Singh and S. Agrawal, "Node localization in Wireless Sensor Networks using the M5P tree and SMOreg algorithms," in Proc. 2013 5th International Conference and Computational Intelligence and Communication Networks, 2013.

[22] L. Wang, L. Tan, C. Yu, and Z. Wu, "Study and application of nonlinear time series prediction in ground source heat pump system," in Proc. 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), pp. 3522–3525, 2012.

[23] A. Cakir, H. Calis, and E. U. Kücüksille, "Data mining approach for supply unbalance detection in induction motor," Expert Systems with Applications, vol. 36, issue 9, pp. 11808–11813, Nov. 2009.

[24] E. Frank, M. A. Hall, and I. H. Witten, "The WEKA Workbench. Online appendix for 'Data Mining: Practical Machine Learning Tools and Techniques,' " Morgan Kaufmann, Fourth Edition, 2016.

[25] D. Ghadiyaram, J. Pan, and A. C. Bovik, "A subjective and objective study of stalling events in mobile streaming Videos," IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 1, pp. 183–197, 2019.

[26] F. Guyard and S. Beker, "Real-time anomalies monitoring for QoE indicators," in Proc. 18$^{\text{th}}$ ITC Specialist Seminar on Quality of Experience, pp. 114–123, May 2008.

[27] S. Akin and M. Fidler, "Backlog and delay reasoning in HARQ system," in Proc. 27th International Teletraffic Congress (ITC), Ghent, Belgium, pp. 185–193, Sept. 2015.

[28] I. K. Jain, R. Kumar, and S. Pawar, "Driven by capacity or blockage? A millimeter wave blockage analysis," in Proc. 30th International Teletraffic Congress (ITC), Vienna, Austria, pp. 153–159, Sept. 2018.

[29] M. Fiedler, P. Carlsson, and A. Nilsson, "Voice and multi-fractal data in the Internet,", in Proc. 26th Annual Conf. on Local Computer Networks (LCN), Tampa, FL, Nov. 2001.

[30] J. W. Bosman, R. D. van der Mei, and R. Nunez-Queija, "A fluid model analysis of streaming media in the presence of time-varying bandwidth," in Proc. 24th International Teletraffic Congress (ITC), Cracow, Poland, Sept. 2012.

[31] M. Fiedler, A. Popescu, and Y. Yao, "QoE-aware Sustainable Throughput for energy-efficient video streaming," in Proc. 2016 IEEE Int. Conf. on Sustainable Computing and Communications (SustainCom), Atlanta, GA, pp. 493–500, Oct. 2016.

APPENDIX

A. Fluid Flow Analysis of the Four-State System

For the resulting four-state system, we obtain a state space as follows:

$$S \in \left\{ \begin{array}{l} s=1 : \text{Channel on} \\ s=2 : \text{Channel off (ARQ)} \\ s=3 : \text{Channel off (Shadowing)} \\ s=4 : \text{Channel off (ARQ and shadowing)} \end{array} \right\} \quad (13)$$

The corresponding drift values d_s represent the rates at which the buffer changes unless the buffer gets full (in case of a positive drift $d_s = r$ when the channel is off) or empty (in case of a negative drift $d_s = r - c$ when the channel is on). The drift values are arranged in the *drift matrix* as follows:

$$\mathbf{D} = \begin{bmatrix} r-c & 0 & 0 & 0 \\ 0 & r & 0 & 0 \\ 0 & 0 & r & 0 \\ 0 & 0 & 0 & r \end{bmatrix} \quad (14)$$

With the transition rates

$$\lambda_i = 1/\mathbf{E}T_{\mathrm{on,i}} \quad (15)$$

$$\mu_i = 1/\mathbf{E}T_{\mathrm{off,i}} \quad (16)$$

the *generator matrix* is obtained as

$$\mathbf{M} = \begin{bmatrix} -\lambda_1-\lambda_2 & \mu_2 & \mu_1 & 0 \\ \lambda_2 & -\lambda_1-\mu_2 & 0 & \mu_1 \\ \lambda_1 & 0 & -\lambda_2-\mu_1 & \mu_2 \\ 0 & \lambda_1 & \lambda_2 & -\mu_1-\mu_2 \end{bmatrix} \quad (17)$$

The vector of the *compound probability distribution* $\vec{F}(x)$ of the buffer content X with components

$$F_s(x) = \Pr\{X \leq x \wedge S = s\} \quad (18)$$

is given by

$$\vec{F}(x) = \sum_{q=0}^{3} a_q \vec{\varphi}_q \exp(z_q x) \quad (19)$$

where the *Eigensystem* $\{z_q, \vec{\varphi}_q\}$ is obtained from

$$z_q \, \mathbf{D}\vec{\varphi}_q = \mathbf{M}\vec{\varphi}_q \quad (20)$$

At this point, we assume that there is no vanishing drift (i.e. $d_s \neq 0 \, \forall s$). The *coefficients* a_q are obtained from the boundary conditions that the buffer is never full when the channel is on, i.e.

$$F_1(K) = \Pr\{S = 1\} \quad (21)$$

and never empty when the channel is off, i.e.

$$F_{\{2/3/4\}}(0) = 0 \quad (22)$$

Finally, the *loss probability* is obtained as

$$P_{\mathrm{L}} = \sum_{s=2}^{4} \left(\Pr\{S = s\} - F_s(K) \right) \quad (23)$$

B. Fluid Flow Analysis of the Two-State System

The two-state system only has one process that governs the rates of the channel; the state space is given by

$$S \in \left\{ \begin{array}{l} s=1 : \text{Channel on} \\ s=2 : \text{Channel off} \end{array} \right\} \quad (24)$$

The *drift matrix* is obtained as

$$\mathbf{D} = \begin{bmatrix} r-c & 0 \\ 0 & r \end{bmatrix} \quad (25)$$

and the *generator matrix* as

$$\mathbf{M} = \begin{bmatrix} -\lambda & \mu \\ \lambda & -\mu \end{bmatrix} \quad (26)$$

The compound probability distribution $\vec{F}(x)$ is given by

$$\vec{F}(x) = \sum_{q=0}^{1} a_q \vec{\varphi}_q \exp(z_q x) \quad (27)$$

with the Eigensystem $\{z_q, \vec{\varphi}_q\}$ similar to (20). The non-vanishing Eigenvalue is a.k.a. the dominant Eigenvalue z_{dom}. Again, we assume that there is no vanishing drift (i.e. $d_s \neq 0 \, \forall s$). The coefficients a_q are obtained from the boundary conditions

$$F_1(K) = \Pr\{S = 1\} \quad (28)$$

$$F_2(0) = 0 \quad (29)$$

Finally, the loss probability is obtained as

$$P_{\mathrm{L}} = \Pr\{S = 2\} - F_2(K) \quad (30)$$