

Statistics and Machine Learning Techniques for Real End-User Experience

USE CASE: SAP Traffic

Würth Phoenix is currently investigating methods from machine learning and advanced statistics that can be used to further improve its real end-user experience (RUE) monitoring solution. The theory underlying the use case described below can be found at ([link to theory part](#)).

Let us study the SAP-traffic of a single subnet of a big company network. Intervention on some of the networking infrastructure is planned in the near future so the goal of the study is to monitor performance before, during, and after the intervention to understand the real end-user experience during all three stages of interest.

A glimpse at classical statistics at query level of all three stages of interest is taken as first step. A traditional mean-based statistical analysis of the registered data reveals that each state of the subnet is characterized by the following values of the performance metrics respectively:

Application latency	Server latency	Client latency	Load time	Bytes	Throughput
148.44	13.01	34.30	234.40	3599.56	20539.07
75.66	12.90	33.97	578.11	3794.66	23601.90
59.47	12.89	33.97	129.68	2928.53	23024.72
961.97	13.70	36.36	2209.59	4769.44	26815.23
239.47	14.69	37.41	930.45	4350.94	25021.24
93.37	17.85	21.77	144.56	4488.91	39712.91
171.90	16.04	15.83	256.73	4629.83	44892.86
160.63	15.41	18.91	889.75	5177.67	43444.65
87.16	15.85	20.46	206.65	5143.68	49078.20
72.01	16.35	23.22	187.96	4642.77	39795.40
639.28	15.78	13.58	826.89	5238.55	47976.28
114.17	15.43	13.00	284.48	5996.52	49225.55
133.00	15.33	12.50	199.31	6722.69	65179.35
254.07	16.05	11.82	647.30	4705.36	47730.86
172.29	17.68	10.45	297.68	4807.53	48943.31

Table 1: Each value is the mean of the respective performance metric of 100 consecutive queries. There is no overlap for consecutive table entries. We limit us to report the means of 500 queries before intervention started (top), 500 queries during intervention (center), and 500 queries after intervention (bottom).

It is hard if not impossible to draw conclusions about the real end-user experience by the study of tables full of values for each performance metric. Let us therefore consider the graphical visualization of a selection of them (see Illustration 1).

The application latency seems to remain constant during all three stages of interest. It does not suffer during intervention neither change its mean. The mean of all queries is detected around 200 ms. Given that in the period under investigation the only changes are those to network hardware and not to applications that run on the particular subnet such a result can be accepted.

The server latency is found to have its mean around 16 ms and mostly lies in a small range [10 ms; 20 ms]. This is important to note, because with this kind of constant server latency changes to the overall network latency can be expected to be merely caused by client latency, where we expect the impact of the intervention to be strongest.

Client latency has a significantly decreasing trend during the complete period of analysis. The initial state has its mean around 34 ms, the final one around 14 ms. The reader is referred to the next paragraph for more details on the client latency.

The throughput has a perceptible ascending trend during the period of interest and mostly lies between 20 kB/s and 50 kB/s.

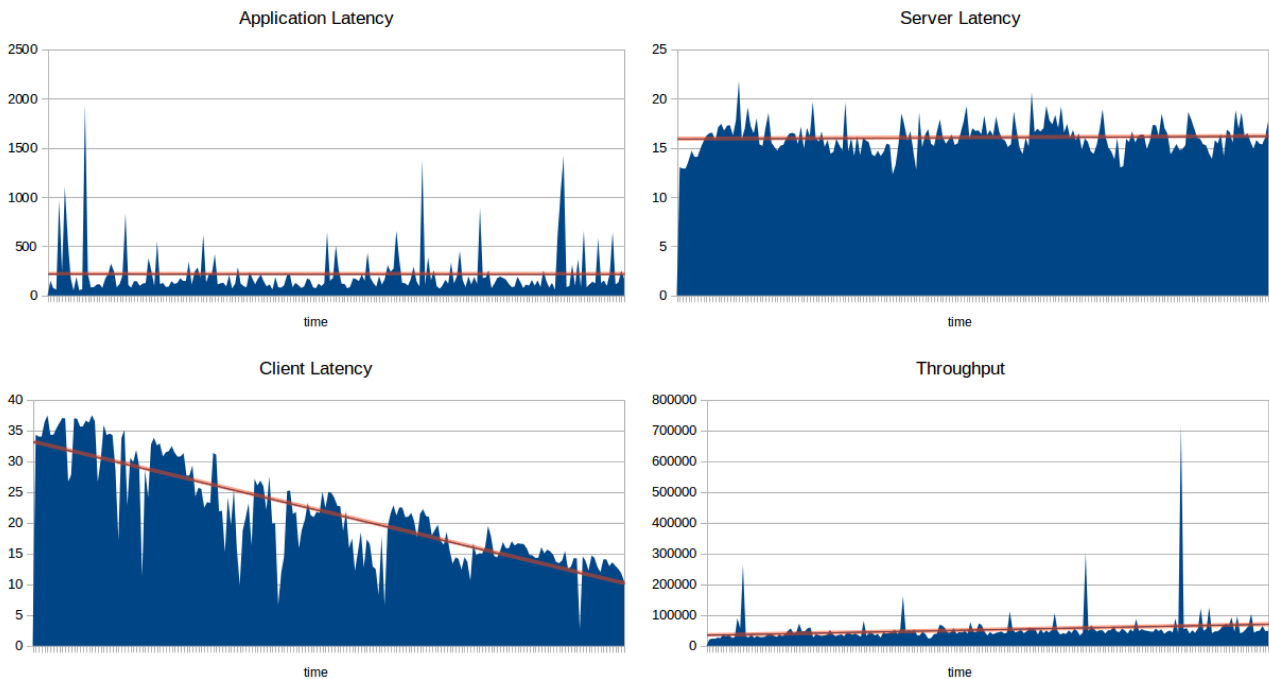


Illustration 1: Standard mean-based statistics of complete traffic

Standard methods allow the detection of trends, but it is hard to get an idea of the bigger picture of what is actually happening. For example let us split the graph of the client latency into the three stages of interest (see Illustration 2). Data from the initial and final stage seem almost homogeneously distributed around the mean. The transition stage between both states instead does not follow the linear trend line calculated from data. There is much movement within the complete range between the initial and the final state. One might think, that this is mainly due to a lack of precision and select to produce more values by averaging fewer queries. The outcome of this analysis might be found on the right of Illustration 2. The mean in the transition stage seems to be less stable and to have a bigger standard deviation.

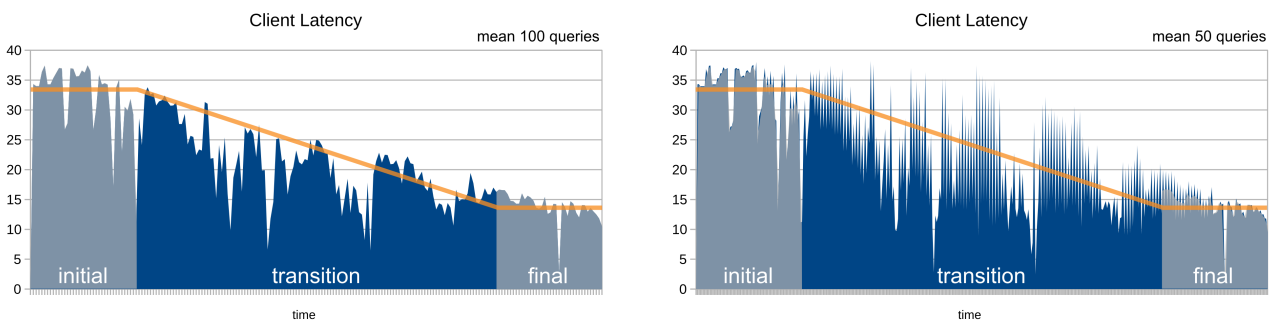


Illustration 2: Client latency - different precision for mean values

It is not possible to inspect in detail what is happening to the subnet under investigation as long as one stays with the mean-based standard statistical analysis. A single value, such as the mean, taken at several time steps is not able to catch a behavior as the one present in the presented example. A solution to this problem can be found in density based statistics (the interested reader is referred to the related [white paper](#) for more detailed theory). In the following we are going to show how techniques from the field of machine learning and advanced statistics can help to go one step further in cases where state-of-the-art analysis is stressed to its limits.

Performance Plots

A performance plot of the same time interval as represented in the standard plots reveals more detailed insights and is therefore closer to the truth. The answer to the problem presented above is straight forward. The client latency does not linearly decrease during the transition stage between the initial state and the final state, but one could rather describe the transition as the coexistence of two states for a certain time interval. For this reason the mean is decreasing more or less linearly, but none of the actual values is close to the calculated mean. One might imagine this as the part of the network that is already using the new hardware simply already being faster while the part that is still using the old hardware not yet profiting from any improvement at all. So the mean over a certain range of consecutive queries is misleading, as it proposes that the whole subnet is already faster at a speed lying between the two local maxima.

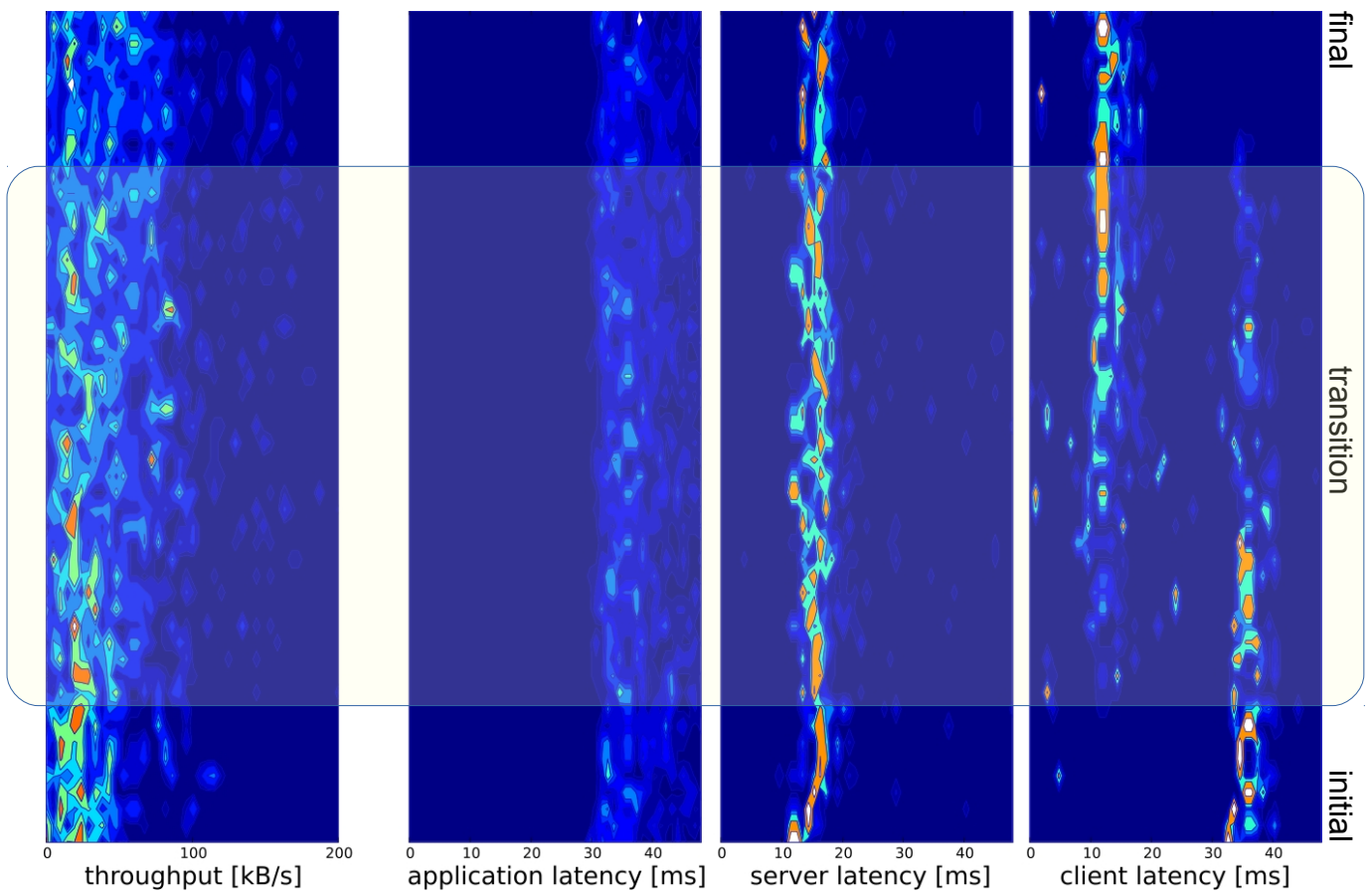


Illustration 3: Performance trend plots

The impact of moving from state-of-the-art analysis towards density based analysis is particularly perceptible when one or more of the performance metrics under investigation have multiple probable states that can occur simultaneously. For example when monitoring the changes of an application and its performance across the network, it could be that different operation systems or device groups have different, specific maxima around which their performance metrics are concentrated. State-of-the-art analysis would simply throw them into the same pot without any possibility to distinguish by calculating the overall mean. The more sophisticated density-based analysis of NetEye RUE monitoring tool can reveal the subgroups and their behavior over time.

Unsupervised learning gives us even more options: the possibility to split traffic data into sparse and dense traffic. It can easily be seen that the density-based algorithm was able to separate more stable “dense” activity from sparse activity. 68% of registered data have been detected to be dense and the remaining 32% sparse. While the classical statistics applied to the dense part of data do not show many changes for server and client latency, it can be seen, that the data mean for the application latency was overestimated when calculated from all available data due to a few, sparse, quite extreme outliers. The truth is that most of the time users experience application latencies round 55 ms while the mean calculated from all the whole data was estimated at 220 ms. A study of the sparse traffic instead reveals that most of the time it is an application latency far from mean of dense standard traffic or extremely high throughput that cause queries that are going to be detected as sparse traffic.

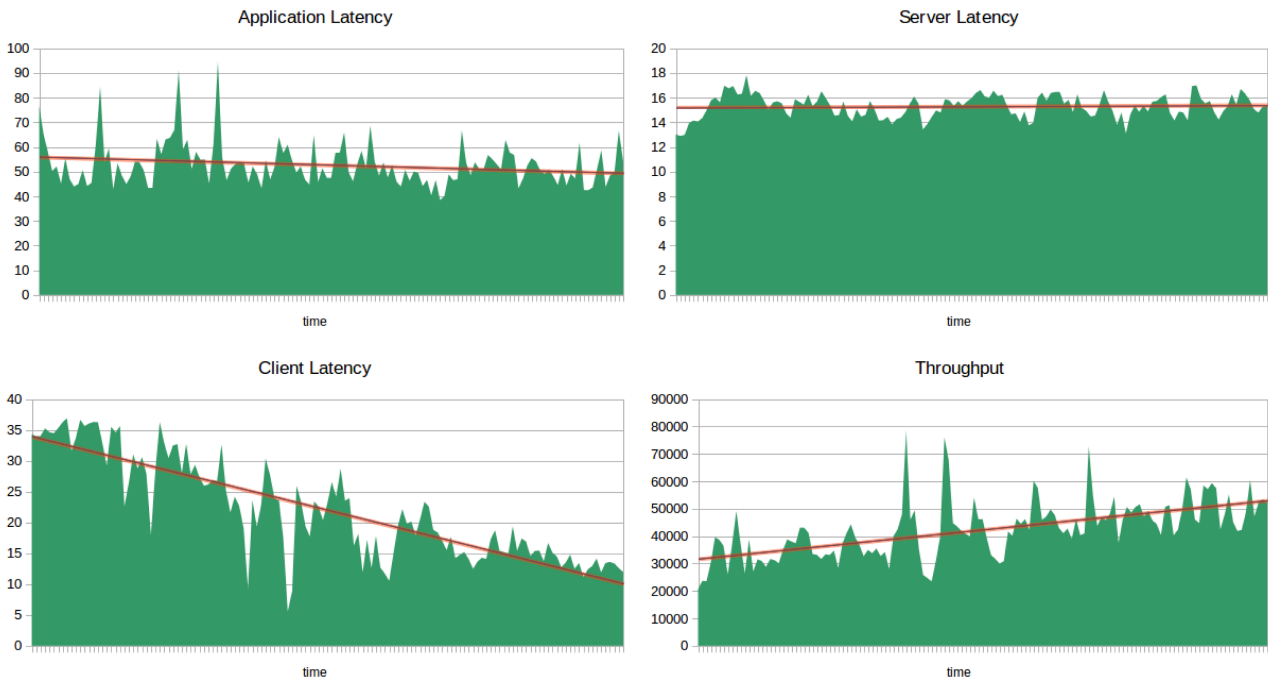


Illustration 4: Standard mean-based statistics of dense traffic (68%)

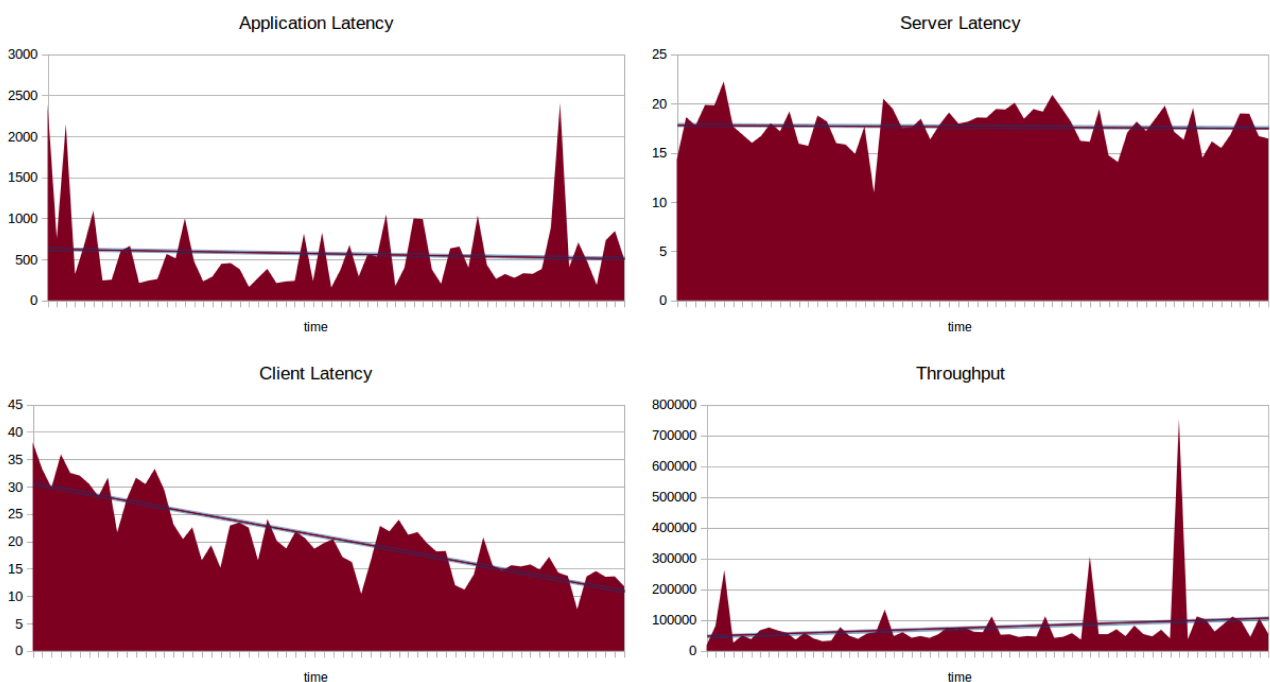


Illustration 5: Standard mean-based statistics of sparse traffic (32%)

Density plots

We perform a 2D density plot using the client latency and the throughput as performance metrics for each of the two sets of queries. The probability density of the queries belonging to dense traffic is shown in green, the probability density of the queries belonging to sparse traffic is shown in red. At the initial state most of the queries are detected around 35 ms and 20 kB/s. The density of the sparse traffic at the same time point shows, that there are also queries with lower latency and/or higher throughput. The second maximum for the client latency appears within the sparse traffic first and is notable before it gets also prominent in the dense traffic. After a period of coexistence of both states all dense traffic lies close to the new maximum. Throughput of the dense traffic lies in the range of 0-100 kB/s while the distribution of throughput of the sparse traffic is much broader.

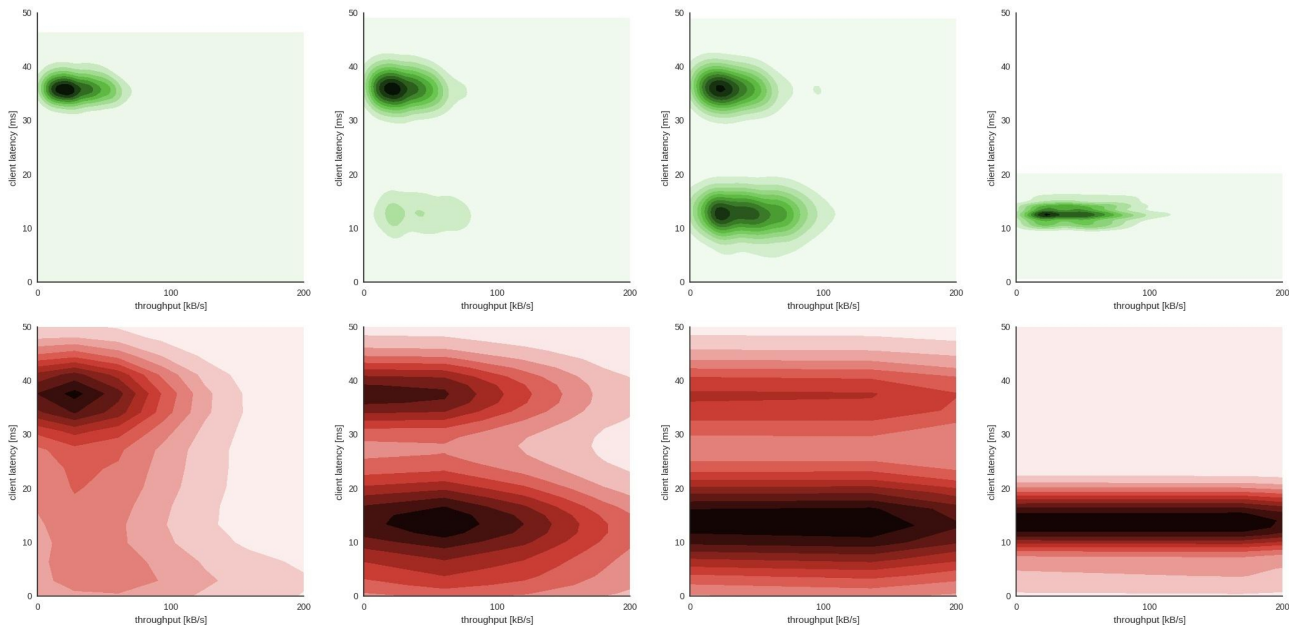


Illustration 6: Density plot of dense and sparse traffic at different time steps. The first couple of plots depicts the initial state. The dense, regular traffic (green) falls within a range with clear borders. The sparse traffic (red) has a broader distribution including throughput and client latencies far away from that of the dense traffic. The traffic around the second local maximum due to the intervention is visible as sparse traffic at first (few queries). Dense traffic remains in clear ranges over time (green row), sparse traffic is much more variable over time (red row). The respective video might be found in our [Blog](#).

Conclusion

The new features involving density-based statistics and unsupervised learning are a useful addition to existing common practice. They are particularly effective in settings where state-of-the-art methods have no way of addressing the bigger picture, because they are limited by a single value (commonly the mean value) for each performance metric that characterizes the current state of a network or application. By replacing this single value by a density, data and their distribution are characterized in much more detail.

The combination of density based statistics and unsupervised learning makes it possible to investigate abnormal traffic first. It furthermore furnishes information about the percentage of queries that is creating stable, constant traffic and those queries that are falling into regions of sparse traffic instead. The latter ones correspond to a potentially worse real end-user experience. Regions of sparse or high density are defined in the multi-dimensional space containing all performance metrics. This is an advantage as one does not only control whether a single performance metric is within range but whether the combination of all performance metrics is probable or not.