# A Novel Methodology for the Automated Detection and Classification of Networking Anomalies

Mohamed Moulay*†, Rafael Garcia Leiva*, Pablo J. Rojo Maroni‡,
Javier Lazaro‡, Vincenzo Mancuso*, Antonio Fernandez Anta*
*IMDEA Networks Institute, Madrid, Spain
†University Carlos III of Madrid, Spain
‡Nokia Networks, Madrid, Spain

***Abstract*—The active growth and dynamic nature of cellular networks makes challenging accommodating end-users with flawless quality of service. Identification of network problems leveraging on machine learning has gained a lot of visibility in the past few years, resulting in dramatically improved cellular network services. In this paper, we present a novel methodology to automate the fault identification process in a cellular network and to classify network anomalies, which combines supervised and unsupervised machine learning algorithms. Our experiments using real data from operational commercial mobile networks show that our method can automatically identify and classify networking anomalies, so to enable timely and precise troubleshooting actions.**

***Index Terms*—Network anomalies, feature selection, clustering, decision trees, machine learning.**

## I. Introduction

There has been a remarkable evolution in cellular networks during the recent years. With 4G networks, and even more with the upcoming 5G revolution, network services have gained a large degree of intelligence, and involve intensive access to both data communication and computing resources. With the evolution of cellular networks, it has also come an increase in structural complexity and heterogeneity of services, which requires the constant monitoring of the communication system. Indeed, the early detection and correction of operational issues and malfunctioning components in the network is needed to provide network customers with flawless quality of service (QoS) [1]. However, the development and deployment of monitoring subsystems have to face the fast increase in technical complexity of networks [2], and a steady increase in the number and capabilities of mobile devices, hence in the number and complexity of service instances requested to the network [3]. To deal with these phenomena, operators are investing resources into the automation of the maintenance and troubleshooting tasks through self-healing functionalities within the scope of self-organizing network operation tools. Self-healing network mechanisms are accountable for detecting, identifying, and making decisions on recovery actions [1].

There exist various proposals for making fault detection and self-healing systems effective in mobile networks [4]. However, while traditional approaches lack flexibility and do not scale, newly-defined approaches based on machine learning (ML) lack *interpretability* of results, which hinders the triggering of proper and effective troubleshooting actions when a system fault is detected.

In this work, we join the ML research stream while focusing on the automated detection and classification of possible network performance anomalies. For training and model evaluation, we use real operational network data collected for cellular service auditing purposes by Nokia in various European countries. Differently from existing proposals, we develop a methodology around an interpretable, cost efficient, scalable, and accessible combination of supervised and unsupervised ML algorithms.

### A. Related Work

Early works on fault detection suggested the use of time series *regression* methods and *Bayesian networks*. For instance, Khanafer *et al.* [5] proposed a method based on Bayesian networks to detect faults in UMTS systems, in which they apply different algorithms to discrete KPIs. Other works, such as [6], rely on a scoring-based system, in which the authors build the fault detection subsystem around labeled fault cases. These cases were previously identified by *experts*, using a scoring system to determine how well a specific case matches each diagnosis target. The work presented in [7] is based on a supervised genetic fuzzy algorithm that learns a fuzzy rule base and, as such, relies on the existence of labeled training sets. Indeed, most of the techniques proposed in the literature focus on using supervised machine learning algorithms [5]–[7]. In this paper we show that it may be convenient to combine different supervised and unsupervised techniques, to achieve interpretability of results, among other features.

Other works make use of advanced mathematical and statistical tools. For instance, Ciocarlie *et al.* [8] address the problem of checking the effect of network changes via monitoring the state of the network, and determining if the changes resulted in degradation. Their fault detection mechanism uses *Markov logic networks* to create probabilistic rules that distinguish between different causes of problems. A framework for network monitoring and fault detection is introduced in [9], using *principal component analysis* (PCA) for dimension reduction, and kernel-based semi-supervised fuzzy clustering with an adaptive kernel parameter. To evaluate the algorithms, they use data generated by means of an LTE system-level simulator. The authors claim that this framework proactively detects network anomalies associated with various fault classes. These methods lack the flexibility of ML-based ones and, differently from our proposal, cannot be fully automated for a generic network context.

Notably, observing the literature, it is worth to mention that most of the proposals are evaluated only through network simulators, and that existing proposals help to detect network issues but they do not help to interpret the network behavior. In contrast, in our work we use data collected in real operational networks, and propose a fully automated ML-based methodology that leads to an easy interpretation of network behaviors. This includes identifying not only the occurrence of problems, but also their root cause.

### B. Original Contribution of the Work

By means of studying the behavior of real networks with respect to TCP performance, the main contribution of this article is a comprehensive ML-based complex methodology that $(i)$ identifies if a network is behaving as expected or is under-performing, and $(ii)$ automatically determines with high accuracy the root causes that lead to performance issues. In addition, we provide an open source implementation of our methodology, which is based on the use of the Scikit-learn library for Python [10]. Besides, we use valuable real data from commercial networks to test our proposal.

### C. Organization of the Paper

The structure of the rest of the paper is as follows. Section II presents background on the ML algorithms used to develop the methodology. Section III takes a detailed look into the methodology we propose. Section IV illustrates the implementation of our methodology and analyzes the results it achieves when using real data. We summarize and conclude the paper in Section V.

## II. Background on Machine Learning

Recent advances in ML are showing its potential use in many different application areas [11]. Networking is one of these areas that could greatly benefit from ML, for instance for the detection and classification of anomalies, or the prediction of its future performance. In this paper we use two well-known and solid ML techniques. The first technique is decision trees, which is used for supervised classification (i.e., it requires a properly labelled dataset from which it learns). The second technique is $k$-means, which is used for unsupervised clustering. Instead of using these techniques to blindly classify and cluster data, we use them in tandem, to make it possible to understand the features of the available dataset (containing time series of network and TCP attributes), and so that the models and results they obtain are interpretable by domain experts.

### A. Decision Trees

A decision tree is a classification approach that assigns a class to a data item by comparing its attributes with certain splitting values for these attributes [12]. This is done in the form of a tree, so each internal node of the tree has an attribute and a splitting value, which are used to send a data item to one of the two children of the node. Each leaf of the tree is associated to a class, so a data item is assigned the class of the leaf it reaches after traversing the tree starting at the root. A nice property of decision trees is that they are interpretable: it is clear from the path followed why a given data item has been assigned a given class.

A decision tree is built from a pre-classified training dataset, which is used to choose the attributes and splitting values of internal nodes, and the classes assigned to the leaves. There are several ways of choosing attributes and splits at each point of the tree construction. In this paper we use the Classification And Regression Tree (CART) algorithm [13] for this, which uses the *Gini impurity* metric [11] to choose the best attribute and split at each point. The Gini impurity of a set of items belonging to $k$ classes is computed as

$$\gamma = \sum_{i=1}^{k} p_i(1-p_i),$$

where $p_i$ is the fraction of items in the set that belong to class $i$. The Gini impurity ranges between 0 and 1, where 0 indicates that all items belong to a single class.

### B. k-means

$k$-means is an unsupervised method for finding compact clusters (and their corresponding cluster centroids) in a set of unclassified data items [11]. The parameter $k$ of the method is the number of clusters to be found. The $k$-means algorithm partitions the data into $k$ subsets so that the distances between the data items and the centroid of their subset is minimized.

Mathematically speaking, this algorithm aims at finding the partition $\{S_1, \ldots, S_k\}$ of the dataset that minimizes the following sum-squared error function:

$$C = \sum_{i=1}^{k} \sum_{x \in S_i} \|x - c_i\|^2,$$

where $\|x - c_i\|$ is the Eucledian distance between item $x$ and $c_i$, and $c_i$ is the centroid of all the items in $S_i$.

## III. Methodology

The methodology proposed to detect and classify network problems is based on the analysis of highly correlated key performance indicators (KPIs), the characterization of the anomalies found, and the derivation of an interpretable model for the identification of new problems. In this paper we have analyzed a collection of experiments in which a file is downloaded, and two KPIs measured and compared: Throughput Data Rate (TDR) and Round Trip Time (RTT). In case of TCP protocol, the TDR of a file download is almost directly derived from the observed RTT. Hence, anomalous data items (downloads with unexpeceted combinations of TDR and RTT) are classified depending on the cause of their anomaly (see Fig. 1). In particular, we will consider in the next section anomalies caused by radio and TCP problems. Finally, these

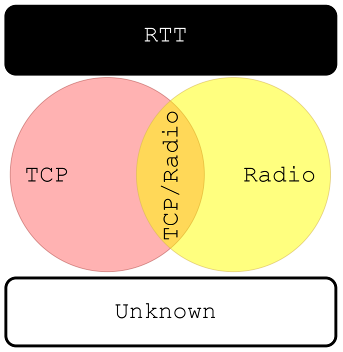

Fig. 1. Possible causes of a performance anomaly, starting with the RTT as the first choice and moving downwards to identify other factors that cause anomalies in the TDR observed (when using TCP for downloading files). The conclusion may be that the cause of the observed anomalies is not identifiable given the current model.

classification results are used to build a model for the detection and automated classification of future anomalies.

## A. *Detecting Anomalies*

As mentioned above, the first step of our methodology is to identify data items (e.g., file downloads) in which the combination of the two correlated KPIs (e.g., TDR and RTT) is inconsistent. To do that, we divide the values that one of the KPIs takes (e.g., the TDR) into different classes (e.g., low, medium, high), and we use all the input data to build a decision tree that classifies the data items into these classes, using only the attributes of the second KPI (e.g., the average/maximum/minimum RTT observed). In order to avoid overfitting, we limit the depth of the decision tree.

As a result of the above process, we have a tree that classifies correctly a large portion of the data items. Intuitively, these are experiments in which the two KPIs are consistent. However, there will be a number of items that are not properly classified by the tree. These are anomalous experiments in which one KPI does not explain the other. For instance, in the next section we will observe file download experiments in which the TDR and the RTT values are inconsistent. These are the data items that we want to explore further, since they may be symptoms of an underlying network problem.

## B. *Clustering Anomalies*

In the second step of our methodology, we restrict our attention to the anomalous data items that were misclassified by the decision tree in the previous step. We have to identify a number $c$ of potential causes for the misclassification, and for each cause, a set of KPIs that can be used to characterize the problem. For instance, in the next section we have identified as potential anomalies *TCP problems* and *radio problems*. Hence, using only the KPIs that correspond to each potential cause, a clustering method (i.e., $k$-means) is used to divide the anomalous data items into two clusters. It is expected (and in fact verified with real data in the next section) that the clustering process will classify the anomalous experiments into two classes: those that are affected by the considered potential cause and those that are not. By applying this process for each

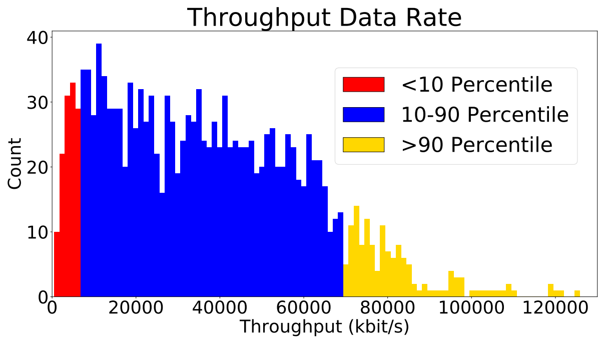

Fig. 2. Distribution of Throughput Data Rate.

of the $c$ potential causes of problems, we obtain a classification of anomalous data items into $2^c$ classes.

## C. *Classifying Anomalies*

In the third step, we build a second decision tree with the full collection of KPIs identified and trained using the items classified in the previous step. The class into which a data item is classified by this second tree reflects what makes it to be anomalous, and it is one of the possible $2^c$ classes identified. For instance, in the next section the tree obtained in this third step determines if a given file download is anomalous because of TCP problems or radio problems (or both, or none). Observe that the outcome may show that there are several causes for the same data item. It is also possible that no cause is assigned to a data item, maybe because it is a false positive or because the actual cause is not among the set of $c$ considered causes.

In our methodology, we leverage on the interpretability of decision trees to find which are the conditions in the KPIs and attributes that make each data item anomalous. These conditions are given by the path in the decision tree from the root to the leaf. This is expected to be useful for the network administrators to identify what is causing the anomaly. This should allow for a fast diagnose and solution of the corresponding network problem.

Observe that this decision tree can also be used in the future to classify other anomalous data items. If the network administrators have been able to identify the issues that made an experiment anomalous in the past, they can then use that experience with new instances, and very possibly fix the problem quickly.

## IV. Results

In this section, we report the results of our methodology, implemented by using the widely adopted scikit-learn library from Python in real data. Scikit-learn provides us with ML algorithms, such as decision trees and $k$-means.

## A. *Experimental Data*

To benchmark quality of service in mobile networks, continuous drive tests with end to end test scenarios are performed every day internally by the network operators (Quality Teams), and externally by third-parties or government regulators. Each of these test campaigns can have up to tens of thousands of

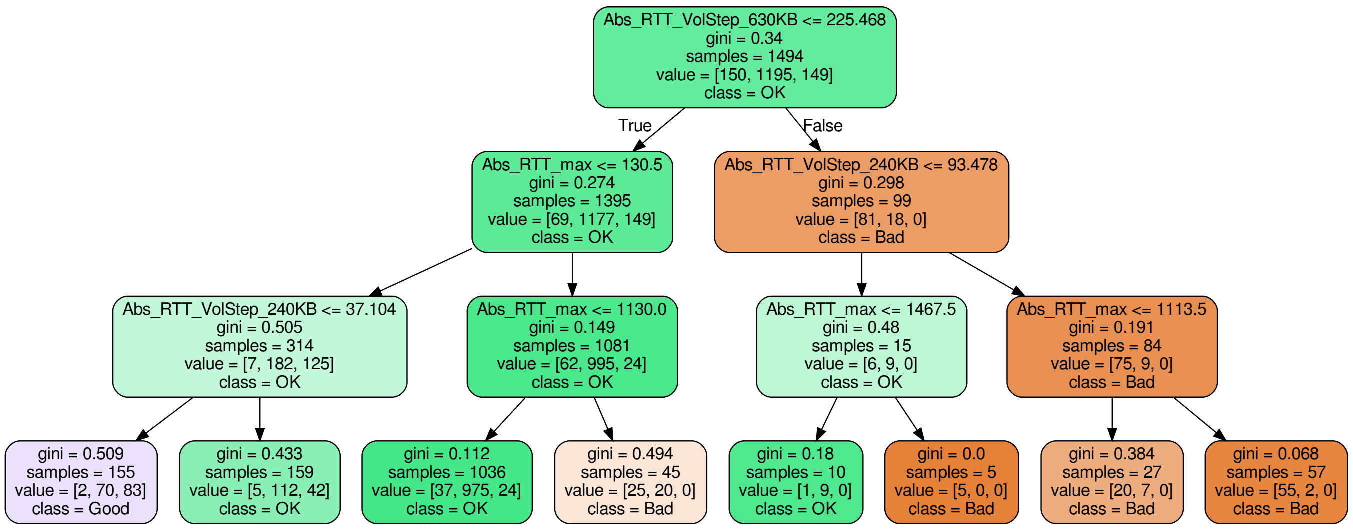

Fig. 3. Decision tree generated with supervised ML using RTT attributes as input and TDR classes inferred from percentiles (see Fig. 2).

individual test cases, from which specific KPIs are calculated. These drive tests are normally executed with off-the-shelf testing equipment (NEMO, TEMS, Swissqual, etc.), capable of running predefined sequences of tests and collecting relevant low level radio and traffic information, as well as application performance statistics. The dataset used for this experimental validation is a real Test Record (TR) used by Nokia for the assessment of various mobile networks in year 2019. The dataset has been generated by processing all the information provided by the testing equipment and aggregating it at test level (count, sum, min, max, average, percentiles, etc.). The resulting dataset has a single row per test and hundreds of columns summarizing all the dimensions (date, time, location, network element information, etc.) and features (radio, TCP/IP, application, etc.) related to that particular test. The data transmitted in the drive tests is synthetic and does not include customer's sensitive information, protected by the European Union General Data Protection Regulation (GDPR) or similar regulations (i.e., the data has been generated by a testing device and not by real users). Finally, potentially sensitive data, such as the identity of the network operator, has been anonymized.

The actual size of the dataset is 2.2 GB in CSV format containing $358,514$ rows of data and $1,164$ attributes. Some of the highlights of the dataset are the ability to filter by test type, infrastructure, operator, vendor, and technology. In this section we have only chosen one value from each of these categories (experiments with other, more complex, choices are left for future work). As a consequence, the data set used corresponds to Hypertext Transfer Protocol (HTTP) file downloads, in cities, over LTE networks, and with one single operator (we anonymize the operator's name for privacy purposes). Filtering the dataset with these parameters produced $1,494$ samples. Henceforth, these are the data items (samples, experiments) to be used and analyzed.

TABLE I
SUMMARY OF THE DECISION TREE RULES OF FIG 3 PRODUCING THE MOST SIGNIFICANT RTT ATTRIBUTES AGAINST THE TDR PERCENTILE CLASS SPLIT SHOWN IN FIG 2

| **Rules** | **Prob.** | **Class** |
|---|---|---|
| If Abs_RTT_VolStep_630KB <= 225.468 | 0.84 | TDR OK |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max <= 130.5 | 0.58 | TDR OK |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max <= 130.5<br>and if Abs_RTT_VolStep_240KB <= 37.104 | 0.54 | TDR Good |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max <= 130.5<br>and if Abs_RTT_VolStep_240KB >37.104 | 0.70 | TDR OK |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max >130.5 | 0.92 | TDR OK |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max >130.5<br>and If Abs_RTT_max <= 1130.0 | 0.94 | TDR OK |
| If Abs_RTT_VolStep_630KB <= 225.468<br>and If Abs_RTT_max >130.5<br>and If Abs_RTT_max >1130.0 | 0.56 | TDR Bad |
| If Abs_RTT_VolStep_630KB >225.468 | 0.82 | TDR Bad |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB <= 93.478 | 0.60 | TDR OK |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB <= 93.478<br>and If Abs_RTT_max <= 1467.5 | 0.9 | TDR OK |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB <= 93.478<br>and If Abs_RTT_max >1467.5 | 1.00 | TDR Bad |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB >93.478 | 0.89 | TDR Bad |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB >93.478<br>and If Abs_RTT_max <= 1113.5 | 0.74 | TDR Bad |
| If Abs_RTT_VolStep_630KB >225.468<br>and If Abs_RTT_VolStep_240KB >93.478<br>and If Abs_RTT_max >1113.5 | 0.96 | TDR Bad |

## B. Data Analysis

*1) TDR Characterization:* Fig. 2 depicts the distribution of the TDR values for the dataset of $1,494$ data items used in this paper. Using these TDR values, we split the samples into three groups: those with TDR values above the 90-th percentile (*good* throughput samples), below the 10-th percentile (*bad* samples), and everything else (*OK* samples). Specifically, the value of the 10-th percentile was $7,796$ kbit/s, while the value of the 90-th percentile was $68,493$ kbit/s. This approach based on the use of statistical percentiles is commonly adopted by regulators and (self-) quality assessment teams for the analysis of complex systems, and by the Nokia team involved in the measurements. Other possible percentile thresholds are possible, for instance the 20-th and 80-th percentiles could be used. This choice does not affect the proposed methodology.

*2) Detecting Anomalies:* We use the TDR split of the $1,494$ samples as a target for a supervised ML decision tree with three classes: Bad, OK, and Good. Since it is well known that the TCP throughput is a function of the RTT, in the classification we use only the RTT attributes available in the dataset to feed the CART algorithm. As mentioned, to avoid overfitting we limited the number of internal tree levels to three. Fig. 3 illustrates the resulting decision tree with RTT attributes. For a given sample, at the root of the node, it is decided if the Abs_RTT_VolStep_630KB attribute (the RTT after downloading 630 KB) value of the sample is smaller or equal to $255.468$ seconds. If true, we move to the left side of the tree, and to the right side otherwise. Other information included in each tree node besides the attribute and split value is the actual value of the Gini impurity for the class split at this level of the tree, the total number of samples considered in the node ($1,494$ at the root) and the number of samples for each label [150, 1195, 149], that correspond to the classes [Bad, OK, Good]. The class value matches the predicted class at this level of the tree (the class that has a highest number of samples). Employing a tree traversal, we will reach a leaf node, where there are no more conditions, and the class for the sample is defined. The decision tree depicted in Fig. 3 was able to classify the TDR samples to some degree, with an accuracy score of $85\%$ (number of correct predictions from the TDR classes computed by the tree, in this case $1,269$, over the total number of samples, which is $1,494$). Table I describes the results generated by the decision tree at relevant nodes.

After using the decision tree built with the CART algorithm for the classification of all the samples in the dataset, we still have $225$ samples, which represent $15\%$ of the original dataset, that are not correctly classified (they are visually reported in Fig. 4). More in detail, Table II reports the confusion matrix, which shows how data samples labeled by means of percentile thresholds are subsequently classified by the decision tree, based on RTT attributes only. Understanding the cause of these misclassification is the target of the methodology presented in the previous section. We therefore consider next how to cluster these anomalous samples, i.e., the data samples that are incorrectly modeled and classified with the decision tree.

*3) Clustering Anomalies:* To cluster the misclassified samples obtained in previous step we have chosen $k$-means, due to its suitability for a medium-sized dataset and the ability to cluster data. We fit $k$-means with attributes picked from a homogeneous class of KPIs. In particular, we use either TCP-related KPIs or radio-related ones. I.e., we try to identify if the problem belongs to TCP events (losses, duplicated ACks, etc.,), radio quality events (e.g., changes in signal strength), or a combination of both. Table III displays the KPIs in question.

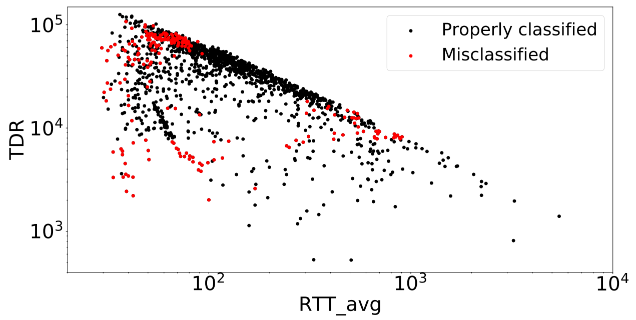

Fig. 4. Graphical plot of the data items properly classified by the decision tree of Fig. 3 in black, and the misclassified data items in red.

TABLE II
CONFUSION MATRIX FOR THE SUPERVISED ML CLASSIFIER OF FIG. 3 TRAINED WITH THE CLASSES IDENTIFIED USING FIG. 2

| | **Bad (ML)** | **OK (ML)** | **Good (ML)** |
|---|---|---|---|
| **Bad (percentile)** | 105 | 43 | 2 |
| **OK (percentile)** | 29 | 1096 | 70 |
| **Good (percentile)** | 0 | 66 | 83 |

Although not presented here for lack of space, we have tested various possibilities for the numbers of target clusters, and found out that the highest score of $k$-means was obtained by using only two clusters. Indeed, the use of $k$-means revealed that the data incorrectly classified by the decision tree can be further clustered into two groups according to TCP attributes. Similarly, the best choice is to use two clusters also in case of using radio attributes. Fig. 5 and Fig. 6 show the clusters obtained by using carefully chosen pairs of attributes. The first figure reports an example of TCP attributes and the second depicts an example of radio attributes. In both cases, it is clear that the clusters obtained with $k$-means separate the samples into those that have TCP (resp., radio) issues and those that do not have issues. Therefore, applying $k$-means to TCP (resp., radio) attributes allows to identify whether there exists a problem with the TCP (resp., radio) performance.

*4) Classifying Anomalies:* The last stage of our methodology involves deriving a final model, using again a decision tree, to identify the root cause of the identified anomalies. For our specific case study, we use a decision tree to classify into the cases that the causes of the anomaly are TCP events, radio conditions, or both. This decision tree has been trained with the $k$-means labels previously collected from TCP-based and radio-based clustering, and a new collection of relevant attributes (see Table III). Fig. 8 showcases the resulting decision tree, which not only classifies TCP and radio issues, but also identifies a class of anomalies that cannot be explained by means of TCP and radio attributes (labeled as “failure to identify”). A problem is classified as “unknow” when it is not possible to distinguish between TCP or radio problem, but it is definitely one of these two. An enumeration of the

TABLE III
RADIO AND TCP KPIS USED BY THE UNSUPERVISED ML ALGORITHM ($k$-MEANS, WITH $k = 2$)

| Type | KPI | Description |
|---|---|---|
| Radio | Start.RSSI.dBm | Received signal strength indication initial value. |
| Radio | End.RSSI.dBm | Received signal strength indication (final value). |
| Radio | Start.RSRP.dBm | Reference Signals Received Power (initial value). |
| Radio | End.RSRP.dBm | Reference Signals Received Power (final value). |
| Radio | Start.SINR.dB | Signal-to-Interference-plus-Noise Ratio (initial value). |
| Radio | End.SINR.dB | Signal-to-Interference-plus-Noise Ratio (final value). |
| TCP | Abs_CWIN_avg | Average congestion window size. |
| TCP | Abs_CWIN_max | Maximum congestion Window size. |
| TCP | Abs_RWIN_avg | Average TCP receive window. |
| TCP | Abs_RWIN_max | Maximum TCP receive window. |
| TCP | Abs_PacketLost_sum | PacketLost total value. |
| TCP | Abs_IdleTime_avg | Average gap value between consecutive TCP segments. |
| TCP | triple_dupacks_b2a | Triple duplicate ack value (server to client). |

rules applied by the decision tree can be found in Table IV. Fig. 7 depicts the points that belong to each class in terms of TDR and RTT. Observing the final decision tree, we can distinguish if the problem is due to TCP with a probability of 0.623, or radio with probability of 0.60 (note that events are not mutually exclusive so that their probabilities do not need to sum up to one or less). The overall score of the tree is 70%, which means that 70% of the anomalies are identified, jointly with their root causes. Misclassification is mostly due to lack of data, since after three split levels, the amount of data at some leaf nodes is no longer statistically relevant. Note that the "Failure to Identify" class shown in the decision tree of Fig. 8 and in Table IV is due to a lack of training data over other attributes. For example, the availability of Domain Name System (DNS) and Transport Layer Security (TLS) measurements might help to identify more types of network problems and further complement the analysis in a fully automated way.

With the above simple example we have shown that we have defined an automated anomaly detection system, which is functional to improve the quality of the networking service. Indeed, the system is able to self-identify network performance issues, and so it can be used to alert the right support personnel, either TCP or radio experts in the presented example, which will receive as input the rules from Table IV that raised the alarm. Our methodology is easy to implement, and it can be deployed in production environments.

## V. CONCLUSION AND FUTURE WORK

In this paper, we have presented a methodology that allows to fully automate the process of identifying anomalies in the behavior of a network. The main advantages of the proposed approach are that it can be implemented in a fully automated way and that its results are interpretable. Specifically, our methodology is based on the compound application of well-established supervised and unsupervised ML tools.

We have also provided an application example based on real data from operational cellular networks. In the example, we have shown that when we identify anomalies in TCP throughput with our methodology, we are also able to further investigate the root causes of anomalies. This feature is key to promptly activate precise and effective troubleshooting actions.

Furthermore, our methodology is generic and can be used to automatically detect several types of networking problems, and examine several classes of potential anomaly causes, without losing the interpretation of the results. As such, it

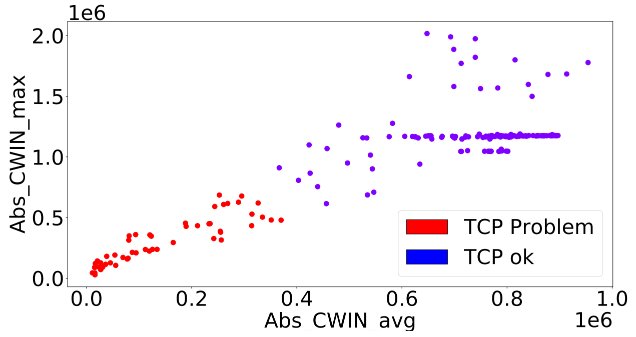

Fig. 5. The two $k$-means clusters obtained with TCP attributes with respect to the Congestion Window average and maximum value attributes.

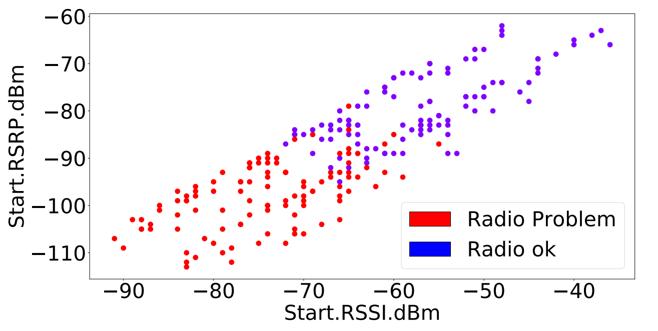

Fig. 6. The two $k$-means clusters obtained with radio attributes with respect to the RSSI and RSRP attributes.

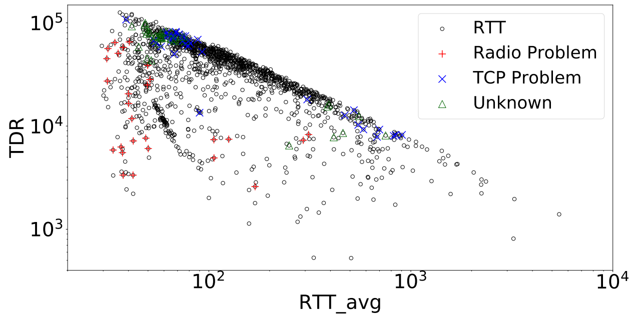

Fig. 7. The outcome of the misclassified points using a combination of unsupervised and supervised ML in Fig. 8 and properly classified points with the supervised ML classifier in Fig. 3.

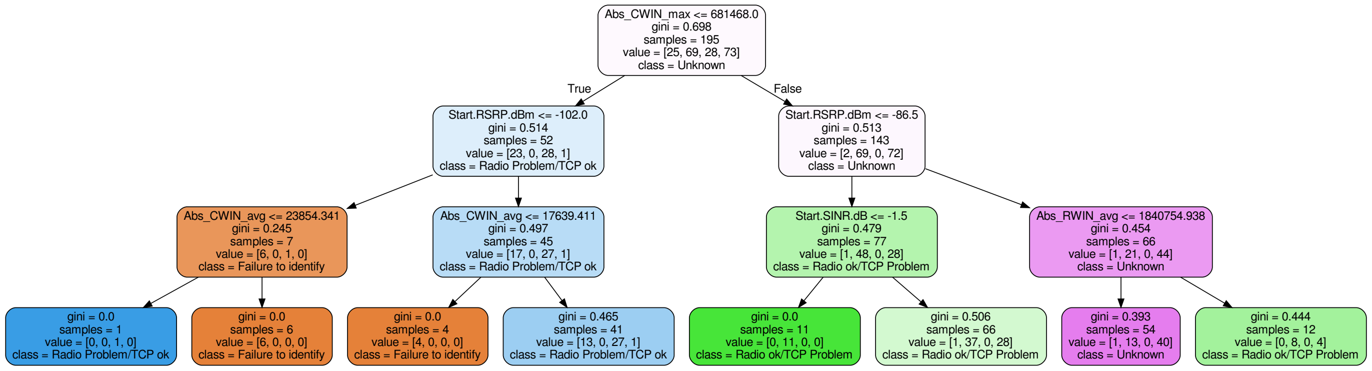

Fig. 8. Anomaly detection decision tree with TCP, radio KPIs from Table III, and $k$-means labeled clusters as classes.

TABLE IV
HIGHLIGHTS OF THE DECISION TREE RULES FOR DETECTION OF ANOMALIES IN FIG. 8 SHOWCASING THE DOMINANT KPIS FROM TABLE III AND CLASSES FROM FIG. 8

| **Rules** | **Probability** | **Class** |
|---|---|---|
| If Abs_CWIN_max $<=$ 681468.0 | 0.54 | Radio Problem |
| If Abs_CWIN_max $<=$ 681468.0<br>and if Start.RSRP.dBm $<=$ -102.0 | x | Failure to identify<br>(lack of data) |
| If Abs_CWIN_max $<=$ 681468.0<br>and if Start.RSRP.dBm $>$ -102.0 | 0.60 | Radio Problem |
| If Abs_CWIN_max $<=$ 681468.0<br>and if Start.RSRP.dBm $>$ -102.0<br>and If Abs_CWIN_avg $>$17639.411 | 0.66 | Radio Problem |
| If Abs_CWIN_max $>$681468.0 | x | Unknown problem!<br>(investigate further) |
| If Abs_CWIN_max $>$681468.0<br>and if Start.RSRP.dBm $<=$ -86.5 | 0.62 | TCP Problem |
| If Abs_CWIN_max $>$681468.0<br>and if Start.RSRP.dBm $>$ -86.5 | 0.67 | Unknow problem!<br>(investigate further) |

can be stationed by current cellular networks and used for future cellular monitoring infrastructures. Future works will involve combining different ML algorithms and study more KPI classes. Most importantly, we will study the design of more complex automatic interpreters for the results of the analysis.

## ACKNOWLEDGMENTS

This research was supported by the Region of Madrid through the TAPIR-CM project (S2018/TCS-4496). The work of V. Mancuso was supported by the Ramon y Cajal grant RYC-2014-16285 from the Spanish Ministry of Economy and Competitiveness.

## REFERENCES

[1] J. P. Santos, R. Alheiro, L. Andrade, A. L. Valdivieso-Caraguay, L. I. Barona-López, M. A. Sotelo-Monge, L. J. Garcia-Villalba, W. Jiang, H. Schotten, J. M. Alcaraz-Calero, Q. Wang, and M. J. Barros, "SELFNET framework self-healing capabilities for 5G mobile networks," *Trans. Emerg. Telecommun. Technol.*, vol. 27, no. 9, p. 1225–1232, Sep. 2016. [Online]. Available: https://doi.org/10.1002/ett.3049

[2] P. Yang, Y. Xiao, M. Xiao, and S. Li, "6G wireless communications: Vision and potential techniques," *IEEE Network*, vol. 33, no. 4, pp. 70–75, July 2019.

[3] FCC, "2013 Measuring Broadband America February Report," FCC's Office of Engineering and Technology and Consumer and Governmental Affairs Bureau, Tech. Rep., 2013.

[4] A. Asghar, H. Farooq, and A. Imran, "Self-healing in emerging cellular networks: Review, challenges, and research directions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1682–1709, 2018.

[5] R. M. Khanafer, B. Solana, J. Triola, R. Barco, L. Moltsen, Z. Altman, and P. Lazaro, "Automated diagnosis for UMTS networks using Bayesian network approach," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 4, pp. 2451–2461, July 2008.

[6] S. Rezaei, H. Radmanesh, P. Alavizadeh, H. Nikoofar, and F. Lahouti, "Automatic fault detection and diagnosis in cellular networks using operations support systems data," in *2016 IEEE/IFIP Network Operations and Management Symposium (NOMS 2016)*, April 2016, pp. 468–473.

[7] E. J. Khatib, R. Barco, A. Gómez-Andrades, and I. Serrano, "Diagnosis based on genetic fuzzy algorithms for LTE self-healing," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1639–1651, March 2016.

[8] G. Ciocarlie, U. Lindqvist, K. Nitz, S. Nováczki, and H. Sanneck, "On the feasibility of deploying cell anomaly detection in operational cellular networks," in *2014 IEEE Network Operations and Management Symposium (NOMS 2014)*, May 2014, pp. 1–6.

[9] Q. Liao and S. Stanczak, "Network state awareness and proactive anomaly detection in self-organizing networks," in *2015 IEEE Globecom Workshops (GC Wkshps)*, Dec 2015, pp. 1–6.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. Springer, 2009. [Online]. Available: http://www-stat.stanford.edu/ tibs/ElemStatLearn/

[12] W.-Y. Loh, "Fifty years of classification and regression trees," *International Statistical Review*, vol. 82, no. 3, pp. 329–348, 2014.

[13] L. Breiman, J. Friedman, R. Olshen, and C. Stone, "Classification and regression trees." *Kluwer Academic Publishers, New York*, 1984.