

Predictive Analytics: A Shortcut to Dependable Computing

Mirosław Malek^(✉) 

Advanced Learning and Research Institute (ALaRI),
Università della Svizzera italiana, Lugano, Switzerland
malekm@usi.ch

Abstract. The paper lists three major issues: complexity, time and uncertainty, and identifies dependability as the permanent challenge. In order to enhance dependability, the paradigm shift is proposed where focus is on failure prediction and early malware detection. Failure prediction methodology, including modeling and failure mitigation, is presented and two case studies (failure prediction for computer servers and early malware detection) are described in detail. The proposed approach, using predictive analytics, may increase system availability by an order of magnitude or so.

Keywords: Failure prediction · Feature selection · Malware detection · Modelling · Predictive analytics · Proactive Fault Management

1 Introduction: Three Tyrants and the Permanent Challenge

With ever-growing system complexity, ever more stringent timeliness requirements and the uncertainty on the rise due to failures and cyber-attacks one should consider three tyrants¹ that impact not only computer and communication systems operation but also our lives. They are:

1.1 Complexity

The growth of complexity cannot be stopped in practice due to permanent strive for new features and applications and continuously growing number of users and objects (things). In fact, the Internet of Things (IoT) is turning into the Internet of Everything where virtually trillions of devices will be connected ranging from self-driving vehicles to coffee machines. The continuous strive for improved properties such as higher performance, low power, better security, higher dependability and others will continue with further requirements for system openness, fading cries for privacy and flow of incredible volumes of data, yes, big data. This situation seems to be beyond control and is simply part of our civilization.

¹ Inspired by a quote from Johann Gottfried von Herder (1744-1803): “Die zwei größten Tyrannen der Erde: der Zufall und die Zeit” (Two biggest tyrants on Earth are: the chance and the time).

1.2 Time

Many individuals, especially our professional friends and colleagues, have an illusion that the time can be controlled and manipulated. There are two major problems destroying this illusion:

- a. Time can neither be stopped nor regained
- b. Disparity between physical and logical time is evident and in many applications cannot be reconciled.

This creates an insurmountable challenge as creating a real-time IoT is simply beyond our current capabilities.

1.3 Uncertainty

Since occurrence of faults is frequently unpredictable, we have to deal with uncertainty which can be controlled to a limited extent but, all in all, we have “to cope” with it. Furthermore, the new failure modes, new environmental conditions and new attacks further increase uncertainty.

In view of ever-increasing complexity, ever more demanding timing constraints and growing uncertainty due to new cyber-attacks and new failure modes, the dependability is and will remain a permanent challenge. There is no hope that one day it will go away. In fact, it will become ever more pervasive and significant as impact of faults or attacks may range from minor inconvenience to loss of lives and economic disasters.

2 Failure Prediction and the Paradigm Shift

Knowing the future fascinated and employed millions throughout the centuries. From fortune tellers and weather forecasters to stock analysts and medical doctors, the main goal seems to be the same: learn from history, assess the current state and predict the future.

Analysing the historical record on our ability to predict, we may admit that in long term predictions we have miserably failed. Stellar examples include cars, phones, mainframes, personal computers, radios. There are some notable exceptions such as Moore’s Law on processor performance but, in principle, especially with respect to breakthrough technologies we were more often wrong than right.

Since a long term future is so difficult to predict, we focus on short, 1–5 min predictions which, as practice shows, have much higher probability of success. In fact, a spirit of this paper can be succinctly summarized by a Greek poet, C. P. Cavafy (1863–1933) who wrote: “Ordinary mortals know what’s happening now, the gods know what the future holds because they alone are totally enlightened. Wise men are aware of future things just about to happen.”

We have demonstrated that, in fact, such short term predictions may be very effective in enhancing computer systems dependability regardless of the root cause of failure, be it software or hardware.

Not surprisingly, a plethora of methods for short term prediction has been developed [1], and many of them perform very well. Take a method based on the Universal Basis Function [3], for example. We have developed a data-driven, dynamic system approach using modified radial basis function which is able to predict performance failures for telecommunication applications in 83% of cases.

Knowing even a short term future may help us to avoid a disaster, better prepare for an imminent failure or increase the system performance.

Predictive analytics, is the processing of algorithms and data which result in prediction of user's behaviour, system status or environmental change. Knowing the future system state or user's behaviour can significantly increase performance, availability and security.

With predictive analytics a drastic paradigm change is in the making. While in the past, we have mainly analysed a system behaviour, and currently, in most cases, we observe the status and behaviour to react to changes in the system, the next big trend is to predict the system behaviour and construct the future (see Fig. 1).

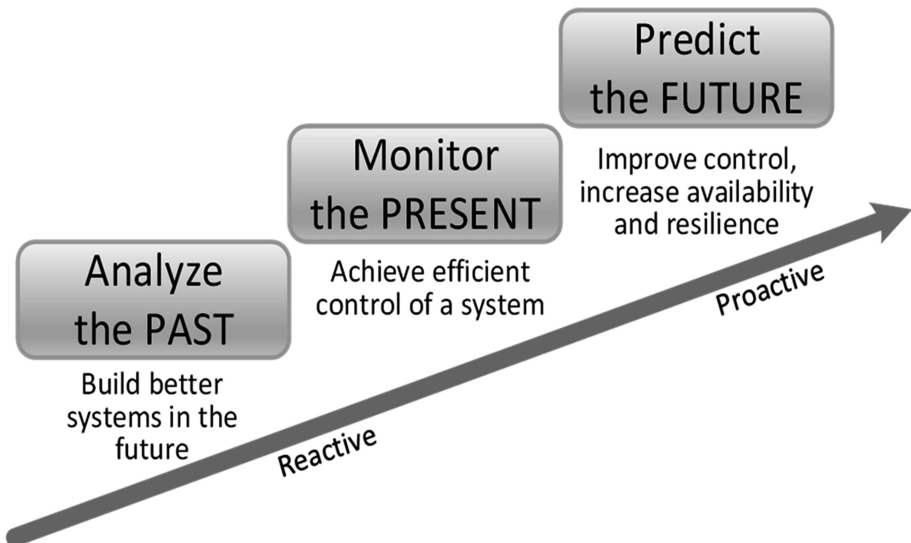


Fig. 1. The paradigm shift.

This leads us to Proactive Fault Management which uses algorithms and methods of failure prediction in order to avoid a failure (e.g., by failover) or minimize its impact [1, 2]. The area is maturing and has already entered the industrial practice, especially in the area of preventive maintenance. It seems to be already evident that the Industry 4.0 will use it as a major design paradigm. The potential of such approach is immense as it may improve a system availability by an order of magnitude or more.

This philosophy requires a change of the mindset: don't wait for a failure, anticipate and avoid it or at least minimize the potential damage. I call it a shortcut to dependable computing as we do not wait for a failure, we act on it before its occurrence.

3 Modelling for Prediction in a Nutshell

“The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work.”

John von Neumann (1903 - 1957)

The key to good predictive capabilities is a development of an appropriate model and selection of the most indicative features (also called variables, events or parameters by different research communities). Unfortunately, models are just an approximate reflection of reality and their distance to reality varies. Since, obviously, it is a challenge to capture all properties of such a complex system as, for example, computer cloud, the typical approach has been to focus on specific properties or functionalities of a system. In Fig. 2 we show the methodology of model development and its application. The interesting part is that the choice of model is not as important as selection of the most indicative features which have decisive impact on the quality of prediction [3].

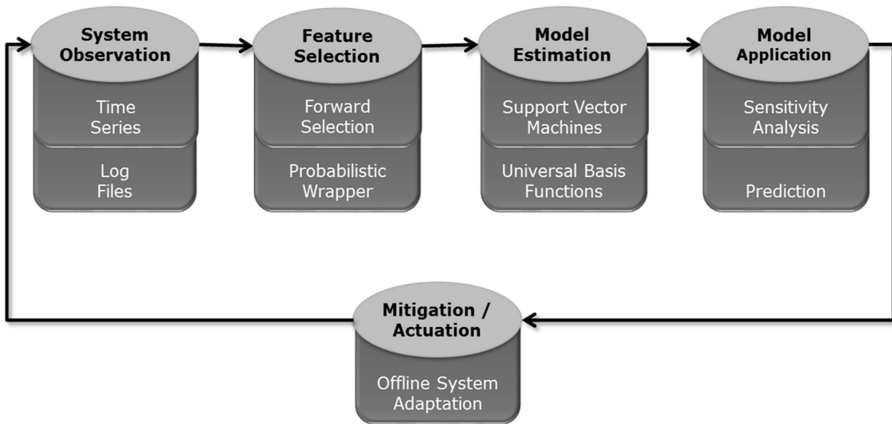


Fig. 2. Building blocks for modelling and failure prediction in complex systems either during runtime or during off-line preparation and testing. System observations include numerical time series data and/or categorical log files. The feature selection process is frequently handled implicitly by system expert’s ad-hoc methods or gut feeling, rigorous procedures are applied infrequently. In recent studies attention has been focused on the model estimation process. Univariate and multivariate linear regression techniques have been used but currently nonlinear regression techniques such as universal basis functions or support vector machines are applied as well. While prediction has received a substantial amount of attention, sensitivity analysis of system models has been largely marginalized. Closing the control loop is still a challenge. Choosing the right mitigation/actuation scheme as a function of quality of service and cost is nontrivial [2, 6].

4 Predictive Analytics and Its Applications

Predictive analytics applies a set of methods/algorithms or classifiers that use historical and current data to forecast future activity, behaviour and trends.

Predictive analytics involves creating predictive models and applying statistical analysis, function approximation, machine learning and other methods to determine the likelihood of a particular event taking place.

In two case studies, we demonstrate the power of prediction or early detection to:

- (1) Failure prediction in computer servers
- (2) Early detection of malware under Android operating system.

The approaches are general and are applicable to other domains such as disturbance prediction in smart grids [4] and predictive maintenance [5].

5 Dependability Economics

Dependability economics concerns the risk and cost/benefit analysis of IT infrastructure investments in an enterprise caused by planned or unplanned downtime as a result of scheduled maintenance, upgrades, updates, failures, disasters and cyber attacks. Research community shied away from this question with only a few notable exceptions (e.g. D. Patterson, UC-Berkeley), yet from industrial perspective the problem is fundamental. Providers and users want to know what will be the Return-On-Investment (ROI) when they invest in dependability improvement.

Furthermore, we are able to assess what benefits, with respect to dependability, we can get by improving the prediction quality measured in terms of precision and recall [6].

6 Failure Prediction Methodology

The goal of online failure prediction is to identify, at runtime, whether a failure will occur in the near future based on an assessment of the monitored current system state and the analysis of past events. The output of a failure predictor is the probability of a failure imminence in the near future. A failure predictor should predict as many failures as possible while minimizing the number of false alarms. Numerous prediction methods are already successfully used for the enterprise computer systems for online, short-term prediction of failures [1]. The prediction quality is identified as a critical part of the entire predict-mitigate approach.

A design of a failure predictor should be conducted in three phases depicted in Fig. 3 [4]. In the first phase a model of the system should be conceived. The model should clearly identify parts of the system where, based on historical records, failures are most frequent and establish a relation between system parts in terms of fault and error propagation.

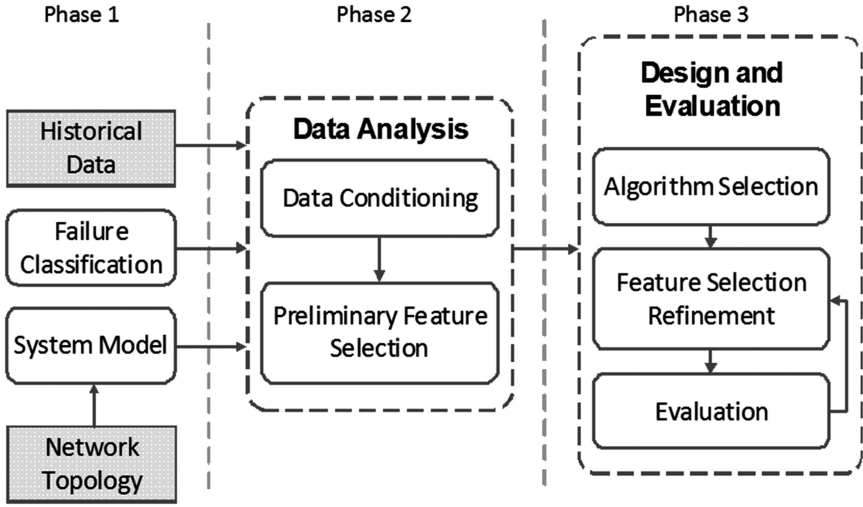


Fig. 3. Failure predictor design stages.

The main usage of the model is in preliminary selection of the most indicative features. Historical data, that are necessary to train and evaluate a predictor, may be obtained from measurement logs.

In the second phase, the obtained dataset should be analyzed. Data conditioning includes extraction of the features and structuring the data in a form that may be used as input for the prediction algorithm. In particular, each data set in the stream, that describes one system state, should be associated with a failure type or marked as failure-free. A preliminary feature selection should be conducted while taking into account system model. Feature selection is the process of selecting the most relevant features (and examples for algorithm training) and combining them in order to maximize predictors’ performance; discard redundant and noisy data; obtain faster and more cost-effective algorithm training and online prediction; and better interpret the data relations (data simplification for better human understanding). Feature selection methods may be classified as filters, wrappers and embedded methods. A widely used filter method is Principal Component Analysis (PCA). PCA converts a set of correlated features into a set of linearly uncorrelated features (principal components) using orthogonal transformation. The procedure is independent with respect to the type of the prediction algorithm that will be used and thus very appropriate for preliminary selection of features. Numerous software packages are available for feature selection, including those that are a part of popular tools for statistical analysis (e.g. Matlab/Octave, Python and R). A good overview of feature selection methods is given in [7, 8].

In the final stage, the predictor is adapted and evaluated. In fact, an ensemble of predictors may be used to improve quality of prediction. Having in mind a large number of existing failure prediction algorithms, the most viable solution is to select and to adopt one of them. A comprehensive survey of failure prediction algorithms is

given in [1]. Three main approaches used for prediction are: failure tracking, symptom monitoring and detected error reporting. Failure tracking draws conclusions about upcoming failures from the occurrence of the previous ones. These methods either aim at predicting the time of the next occurrence of a failure or at estimating the probability of failures co-occurrence. Symptoms are defined as side effects of looming faults that not necessarily manifest themselves as errors. Symptom-monitoring based predictions analyze the system features in order to identify those that indicate an upcoming failure. Several methodologies for the estimation were proposed in the past, including function approximation, machine-learning techniques, system models, graph models, and time series analysis. Finally, the methods based on detected error reporting, such as the rule-based, the co-concurrence-based and the pattern recognition methods, analyze the error reports to predict if a new failure is about to happen.

In order to speed up the prediction, the set of selected features should be refined by extracting the most indicative ones. To facilitate the process, heuristics (such as the ones presented in [7, 8]) may be employed. After each iteration, the quality of prediction has to be evaluated. The process terminates when a sufficient quality of prediction is reached so that resilience and availability are improved.

7 Failure Mitigation

Once the prediction mechanisms anticipate a failure, corrective actions that will mitigate it should be scheduled and activated. The mitigation is composed of three phases [4]: diagnosis, decision on countermeasures and implementation of countermeasures. In the diagnosis phase, the output of the prediction is analyzed. Additional algorithms may be employed to better identify the location of the anticipated failure. In the second phase of mitigation, a decision on a countermeasure is taken. This decision should take into account the probability of a failure (provided by the predictor), the cost of the measure (e.g. maintenance cost or the cost in terms of the number of customers affected), the probability of a successful mitigation and the overall effect on resilience (for example the effect on steady-state availability). Finally, the implementation of the countermeasure has to be performed.

The ultimate goal is to fully avoid the failure (e.g. by failing over an application to another server). If that is not possible, then the effect of a failure should be minimized or confined (e.g. by preventive load shedding) or a preparation of repair actions may be triggered to minimize the repair time (e.g. by checkpointing or saving critical files). Some of these techniques can lead to a system performance degradation. For example, if a failure predictor was wrong, unnecessary preventive load shedding may be conducted affecting a subset of customers. The entire process may be implemented as fully automated or it may require the involvement of an operator for decision-making. This may depend on the type of mitigation and its cost. When more than one failure is anticipated, a coordinated management of mitigation is required.

8 Case Study 1: Telecommunication System

The system we consider is an industrial telecommunication platform which handles mobile originated calls (MOC) and value adding services such as short message services (SMS) and multimedia messaging services (MMS) [3]. It operates with the Global System for Mobile Communication (GSM) and General Packet Radio Service (GPRS). The system architecture follows strict design guidelines considering reliability, fault tolerance, performance, efficiency and compatibility issues. We focus on one specific system which, at the time we took our measurements, consisted of somewhat more than 1.6 million lines of code, approximately 200 components² and 2000 classes³. It is designed to be operated distributed over two to eight nodes for performance and fault tolerance reasons. We focused on modeling and predicting system events (i.e. calls) which take longer time to be processed than some guaranteed threshold value. We call these events failures or target events (see Fig. 4).

The data we used to build and verify our models consists of

- a. equidistant-time-triggered continuous features and
- b. time-stamped, event-driven log file entries.

We gathered numeric values of 46 system features once per minute and per node. This yields 92 features in a time series describing the evolution of the internal states of the system. In a 24-hour period we collected a total of 132.480 readings. In total we collected roughly 1.3 million system variable observations.

Please note that in special purpose systems such as telecommunication systems probability of correctly predicting a failure is much higher than in a general purpose system when an arbitrary application may be invoked at any time.

When making predictions about the system's future state we must take into account true positive (TP), false positive (FP), true negative (TN) and false negative (FN) classifications. Focusing on TP alone may substantially bias a model. A metric which takes all four prediction outcomes into account is precision P and recall R.

$$P = \frac{TP}{TP + FN} \quad (1)$$

$$R = \frac{FP}{FP + FN} \quad (2)$$

Using a combination of forward selection and backward elimination we have identified two most indicative features, namely the number of semaphores (exceptions) per second and the growth rate in kernel memory. These two features were expressed as a linear combination of nonlinear kernel functions (Gauss/sigmoid) supported by

² A system element offering a predefined service and able to communicate with other components.

³ Classes are used to group related features and functions.

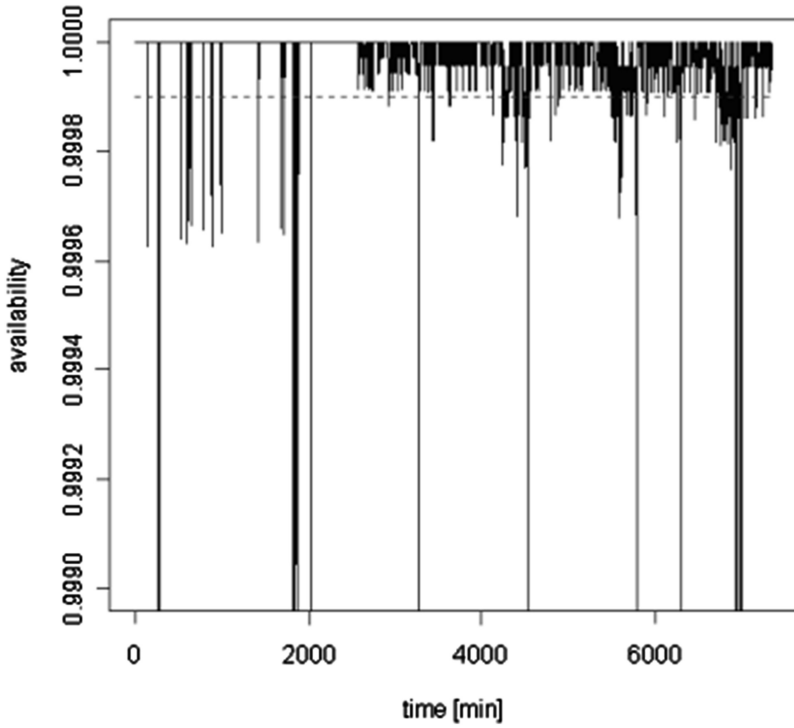


Fig. 4. The target is the system's interval call availability (A) of 0.9999. The dotted line indicates a 0.9999 availability limit. Any drop below that threshold is defined as a failure. Our objective is to model and predict the timely appearance of these failures. The system's interval call availability is reported in consecutive five minute intervals and is calculated as the number of successful calls over the total number of calls in this interval.

training with evolutionary algorithm formed, what we called, a Universal Basis Function (UBF) which gave us excellent results of $P = 0.83$ and $R = 0.78$, outperforming all other methods.

An example of correlating the two features and devising the UBF is given in Fig. 5.

Failure prediction quality deteriorates with increase of a lead time of the prediction. Figure 6 shows the prediction quality measure for the UBF Universal Basis Function, the Area Under Curve (AUC), deteriorates over extended lead time in comparison to methods based on Radial Basis Functions and Multivariate Linear models, including non-linear variations of UBF and RBF.

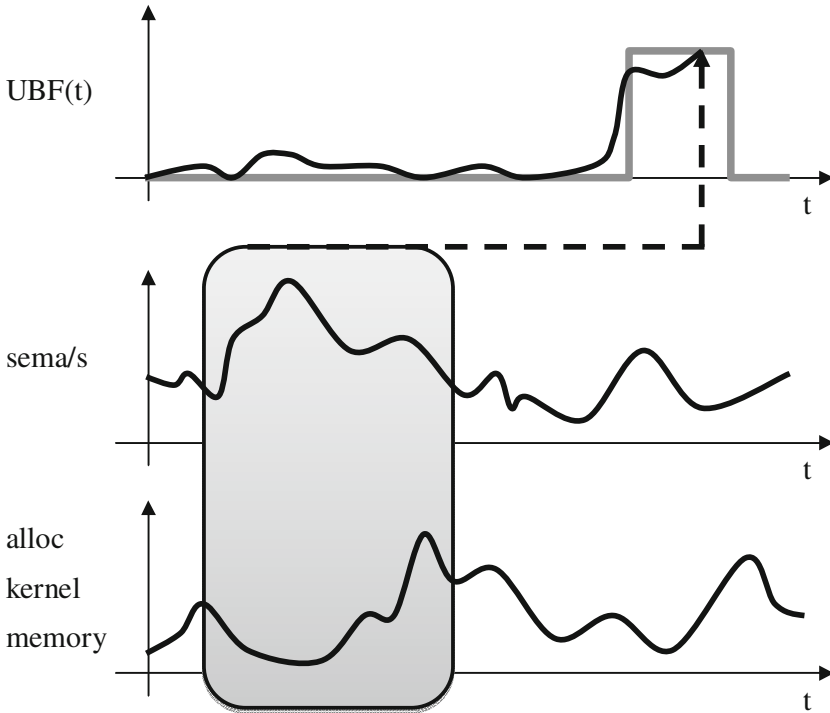


Fig. 5. Time series of the two features and the UBF

9 Case Study 2: Early Malware Detection

With an ever-increasing and ever more aggressive proliferation of malware, its detection is of utmost importance. However, due to the fact that IoT devices are resource-constrained, it is difficult to provide effective solutions.

The main goal of this case study is to demonstrate how prediction methodology can help in finding lightweight techniques for dynamic malware detection. For this purpose, we identify an optimized set of features to be monitored at runtime on mobile devices as well as detection algorithms that are suitable for battery-operated environments. We propose to use a minimal set of most indicative memory and CPU features reflecting malicious behavior.

To enable efficient dynamic detection of mobile malware, we propose the following approach to identify the most indicative features related to memory and CPU to be monitored on mobile devices and the most appropriate classification algorithms to be used afterwards [9]:

1. Collection of malicious samples, representing different families, and of benign samples
2. Execution of samples and collection of the execution traces
3. Extraction of features, from the execution traces, related to memory and CPU

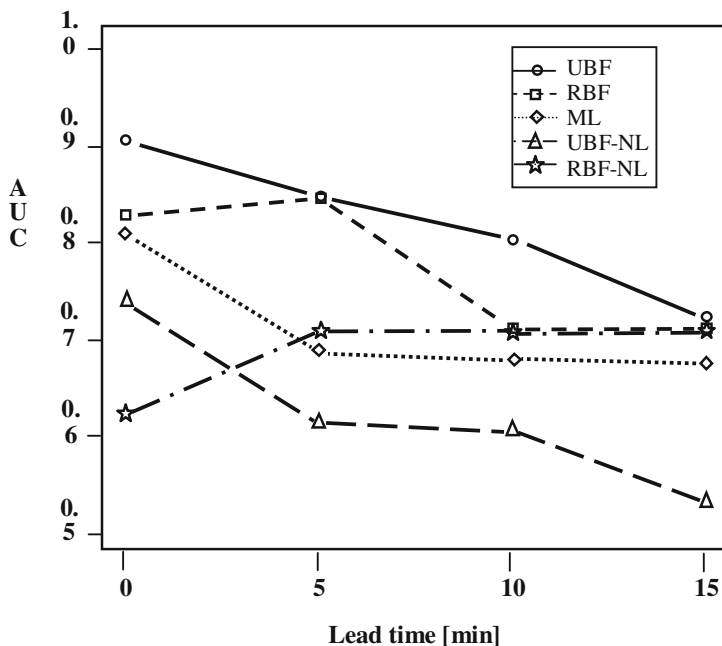


Fig. 6. Failure prediction results: the UBF model (AUC = 0.9024) outperforms the RBF (AUC = 0.8257) and ML (AUC = 0.807) approach with respect to 1, 5, 10 and 15-minutes lead time.

4. Selection of the most indicative features
5. Selection of the most appropriate classification algorithms
6. Quantitative evaluation of the selected features and algorithms

The first step defines the dataset to be used in the remaining parts of the methodology. Namely, it is important to use malicious samples coming from different malware families, so that the diverse behavior of malware is covered to as large extent as possible.

Furthermore, it is needed to set up the execution environment to run malicious samples, so that malicious behavior can be triggered. Additionally, such environment should provide the possibility to execute large number of malicious samples, within reasonable time, so that the obtained results have statistical significance. We have achieved these requirements, first, by using a variety of malware families with broad scope of behavior, second by triggering different events while executing malicious applications and, third, by using an emulation environment that enabled us to execute applications quickly. Since our goal is to discriminate between malicious and benign execution records, we have taken into account also benign samples, and executed them in same conditions used for malicious applications. While usage of an emulator enables us to execute statistically significant number of applications on one hand, on the other hand it is our belief that its usage instead of a real device has a limited or no effect on results, due to the nature of features observed. However, we are aware of the fact that the use of an emulator may prevent the activation of certain sophisticated malicious samples.

Before identification of the most indicative symptoms, feature extraction and selection needs to be performed [7, 8].

We have performed feature selection as a separate step, in which we have evaluated features usefulness based on the following techniques: Correlation Attribute Evaluator, CFS Subset Evaluator, Gain Ratio Attribute Evaluator, Information Gain Attribute Evaluator, and OneR Feature Evaluator.

We have chosen feature selection techniques due to their difference in observing usefulness of features as they are based either on statistical importance or information gain measure. Following, the list of feature selection algorithms that we have used:

- Correlation Attribute Evaluator calculates the worth of an attribute by measuring the correlation between it and the class.
- CfsSubsetEval calculates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low intercorrelation are preferred.
- Gain Ratio Attribute Evaluator calculates the worth of an attribute by measuring the gain ratio with respect to the class.
- Information Gain Attribute Evaluator calculates the worth of an attribute by measuring the information gain with respect to the class.
- OneR Feature Evaluation calculates the worth of an attribute by using the OneR classifier, which uses the minimum-error attribute for prediction.

In order to validate the usefulness of selected features we have used the following detection algorithms having different approach to detection: Naive Bayes, Logistic Regression, and J48 Decision Tree. While most of the steps of the proposed approach are executed offline (i.e., on machines equipped with extensive computational resources), the

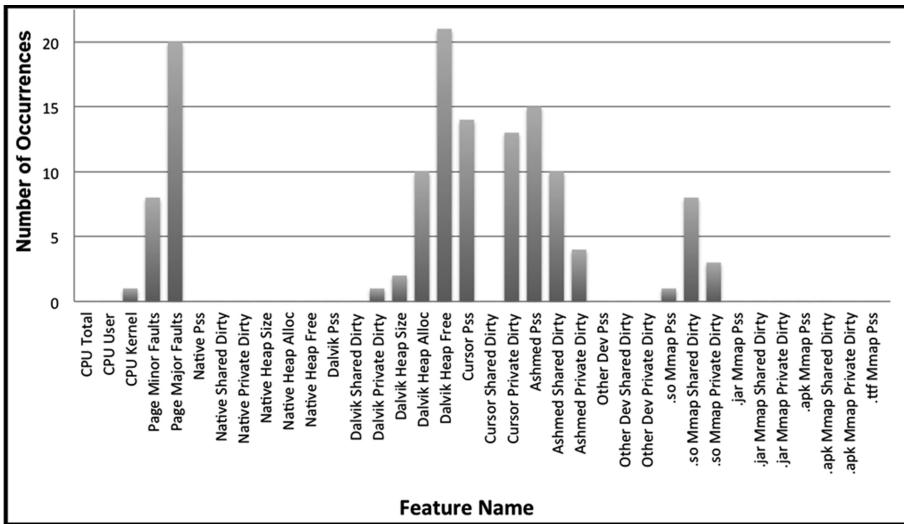


Fig. 7. Frequency of occurrence of features among top 5 of the most indicative features.

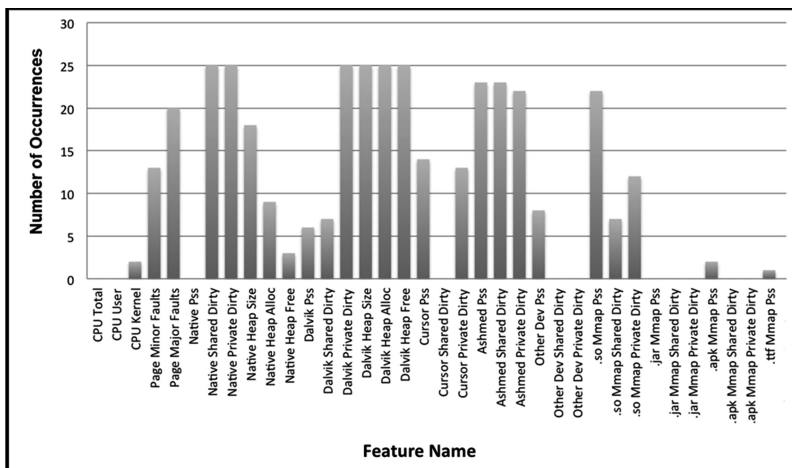


Fig. 8. Frequency of occurrence of features among top 15 of the most indicative features.

classification algorithm will be executed on the mobile devices; thus, it needs to be compatible with their limited resources. This is the reason why we also take into account the complexity of algorithms, and use the ones with low complexity.

Feature selection results are illustrated in Figs. 7 and 8 where the number of occurrences in top 5 or 15 of most indicative features is shown, respectively [10].

Based on these results, we have selected seven features out of 53 that practically give equal or higher probability of malware detection in terms of precision, recall and F-measure (harmonic mean of precision and recall) than the entire set of features (see Table 1). Furthermore, effectiveness of three different classifiers has been compared

Table 1. Performance of the classifiers when different number of features are considered.

		Model		
		Initial	Max. F-measure	Optimized
Naive Bayes	Precision	0.79	0.84	0.84
	Recall	0.76	0.83	0.83
	F-measure	0.77	0.83	0.83
	No. of features	53	7	7
Logistic Regression	Precision	0.84	0.86	0.84
	Recall	0.84	0.86	0.84
	F-measure	0.83	0.86	0.84
	No. of features	53	38	7
J48 Decision Tree	Precision	-	-	0.83
	Recall	-	-	0.83
	F-measure	-	-	0.82
	No. of features	-	-	6

indicating that Logistic Regression yields best results. The approach was validated by using the features related to memory and CPU during execution of 1080 mobile malware samples belonging to 25 malware families.

10 Concluding Remarks

We have outlined an accelerated and more economical method of improving dependability by an order of magnitude or so by using predictive analytics. The main message of this paper is: Do not wait for a failure but predict it and then you have better chance to avoid it or minimize its impact.

To tame three tyrants (complexity, time and uncertainty), we need radically new approaches to keep systems running, simply because current modeling methods and software are not able to handle ever-increasing complexity and ever-growing demand for timeliness. We also need to learn how to cope with uncertainty.

The proposed methodology based on predictive analytics provides an effective, efficient and economical approach to improve dependability, real time performance and security, highly needed, especially in the IoT environments where massive redundancy is usually too expensive and impractical.

With big data and machine learning, predictive analytics is charting a new paradigm shift where application of prediction methods will turn out to be successful in all aspects of computer and communication systems operation, be it performance, security, dependability, real time and others.

In addition to failure prediction and mitigation methodology, we have also presented two case studies on: (1) failure prediction in computer servers and (2) early malware detection where effectiveness of prediction methods has been demonstrated.

In the nutshell, applying the **AMP** principle: **A**nalyze the past, **M**onitor and control the present and **P**redict the future may significantly enhance dependability and other system properties.

Acknowledgement. I would like to acknowledge valuable contributions of my students Günther Hoffmann, Igor Kaitovic and Felix Salfner to the methodology and the case study on failure prediction. Alberto Ferrante and Jelena Milosevic contributed to the malware detection methodology and experiments.

References

1. Salfner, F., Lenk, M., Malek, M.: A survey of online failure prediction methods. *ACM Comput. Surv. (CSUR)* **42**, 10:1–10:42 (2010)
2. Hoffmann, G.A., Trivedi, K.S., Malek, M.: The best practice guide to resource forecasting for computing systems. *IEEE Trans. Reliab.* **56**(4), 615–628 (2007)
3. Hoffmann, G.A., Malek, M.: Call availability prediction in a telecommunication system: a data driven empirical approach, In: 25th IEEE Symposium on Reliable Distributed Systems (SRDS 2006), Leeds, UK (2006)

4. Kaitovic, I., Lukovic, S., Malek, M.: Proactive failure management in smart grids for improved resilience: a methodology for failure prediction and mitigation. In: IEEE GLOBECOM Workshops (SmartGrid Workshop), San Diego, USA, pp. 1–6 (2015)
5. Garcia, M.C., Sanz-Bobi, M.A., del Pico, J.: SIMAP: Intelligent System for Predictive Maintenance: Application to the health condition monitoring of a windturbine gearbox. *Comput. Ind.* **57**, 552–568 (2006)
6. Kaitovic, I., Malek, M.: Optimizing failure prediction to maximize availability, In: 13th International Conference on Autonomic Computing, Würzburg, Germany (2016)
7. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *ACM J. Mach. Learn. Res.* **3**, 1157–1182 (2003)
8. Liu, H., Yu, L.: Toward integrating feature selection algorithms. *IEEE Trans. Knowl. Data Eng.* **17**(4), 491–502 (2005)
9. Milosevic, J., Malek, M., Ferrante, A.: A friend or a foe? detecting malware using memory and CPU features. In: 13th International Conference on Security and Cryptography (SECRYPT 2016), Lisbon, Portugal, pp. 73–84 (2016)
10. Milosevic, J., Ferrante, A., Malek, M., What Does the Memory Say? Towards the most indicative features for efficient malware detection, In: 13th Annual IEEE Consumer Communications and Networking Conference (CCNC 2016), Las Vegas, NV, USA. IEEE Communication Society (2016)