# Applying Classification Methods to Model Standby Power Consumption in the Internet of Things

Leandro Andrade*, Ricardo Rios*†, Tatiane Nogueira*† and Cássio Prazeres*
* WISER @ DCC, Federal University of Bahia, Salvador, Bahia, Brazil
† CInO @ DCC, Federal University of Bahia, Salvador, Bahia, Brazil
Email: leandrojsa@dcc.ufba.br,{ricardoar,tatiane.nogueira,prazeres}@ufba.br

***Abstract*—This paper presents an approach that combines Internet of Things (IoT) technologies and classification methods to improve efficient usage of power consumption. We focused on energy use of electronic devices on standby mode, which represent from 5 to 26% of power consumption in a home. The proposed approach aims at predicting situation in which devices on standby can be turned off, reducing power consumption. In summary, our approach uses motion and current sensors connected to an IoT infrastructure to build a profile about the presence of people at home. Results obtained from our approach present a reduction of the electric energy consumption by applying Machine Learning methods on Internet of Things scenarios.**

## I. Introduction

The number of devices currently connected to the Internet is greater than world's population and, in 2020, this number will increase to about 50 billion [1]. Due to this huge amount of connected devices, a new research area, referred to as Internet of Things (IoT) [2], has attracted the attention of Industry and Academy. In general, IoT aims at coordinating devices connected to the Internet without requiring any human intervention [3], [4]. Concepts and technologies inspired by IoT have been developed in several real-world applications as, for instance: ambient assisted living, efficient management of energy, management of smart environments, smart traffic management, and environmental monitoring.

In the context of efficient management of energy, IoT can be adopted, for instance, to control the energy usage when devices are on standby mode [5], [6]. According to Ross and Meier [7], devices on standby mode represent 5-26% of energy consumption in residences with an average of 67 kWh for a month. Therefore, by dealing with devices on standby mode, one can efficiently use energy and, consequently, save money.

Generally, there are two types of standby mode [8]: passive and active. In the passive mode, the energy is only used to turn on devices by remote control (e.g. TV, DVD player, and Microwave oven). On the other hand, the active mode is used to keep running some services in background (e.g. answering machines and alarm clock).

In relation to the passive mode, IoT can be used to monitor devices and act in power consumption by controlling turn on/off operations. In this sense, sensors and actuators are connected to devices for measuring, for example, electric current, frequency and voltage.



Some works already published have addressed the usage of IoT for management of standby mode of devices. For example, Lee et al. [9] present a proposal to control standby power consumption of home appliances using IoT. Their solution is based on sensor networks to control the power consumption and state of home appliances, and based on a software application (App) designed to run on mobile devices (e.g. smartphone), which collects information about the presence of people at home. This solution requires a complex infrastructure to control the appliances. In addition, the usage of an App may limit its adoption to controlled environment as, for instance, residential one. In commercial or industrial scenario, it is more difficult to control people that have access to the environment.

Zhang et al. [10] developed a system to control power consumption in residences as well. In a similar work, Han and Lim [11] proposed a hierarchical control of home appliances, in which, for example, the user can switch on/off all appliances of a room. However, both systems do not provide an automatic solution to take a decision for switching on/off standby power. These proposals have graphical interfaces and the users can use it to turn on/off home appliances and monitor power consumption.

By analyzing such researches, it is possible to notice they present static solutions without adapting users' needs along time. Aiming at overcoming this drawback, this paper presents an approach to provide efficient consumption of energy in residences using techniques of IoT and Machine Learning (ML).

In our approach, an automatic system continuously monitors and controls electronic devices, collecting information to be later analyzed by classification methods. These methods are used to model users' profile along time, in order to turn off/on the power standby of home appliances or electronic devices. The results presented in this paper emphasize the importance of using models obtained from classification methods to predict user's activities, reducing the energy consumption and saving money. Moreover, the experiments shown in our approach can adapt itself according to the changes in users' daily behavior without any human intervention.

The remainder of this paper is organized as follows. Section II presents the scenario designed to assess the proposed approach. Section III details the experimental setup, emphasizing how the data was preprocessed and the classification methods were used to build the energy consumption model. Section IV shows the results obtained to automatically control appliances on standby mode. Finally, Section V we draw conclusions and discuss future work.

## II. Power Consumption Scenario

The proposed scenario was deployed in a home, where we installed part of an IoT infrastructure to get data about presence of people and power consumption. The experimental analysis was performed in the main room of a house, which has the highest number of appliances to be monitored in the home. The proposed experimental scenario (illustrated in Figure 1) is composed of: two motion sensors, which get data about human presence in a room (positioned at an specific angle to avoid capture of movement of animals and appliances on the floor); a current sensor connected in the same circuit of the home appliances; and a relay to handle the power switch (on/off) of such home appliances. Theses devices are connected to an Arduino[1] board, that is a microcontroller board to prototype with sensors and actuators. In turn, the Arduino board is connected to an internal network over the WiFi protocol. In order to realize the experiment, we used four home appliances, which are: Digital TV, DVD player, Home Theater and Satellite TV decoder.

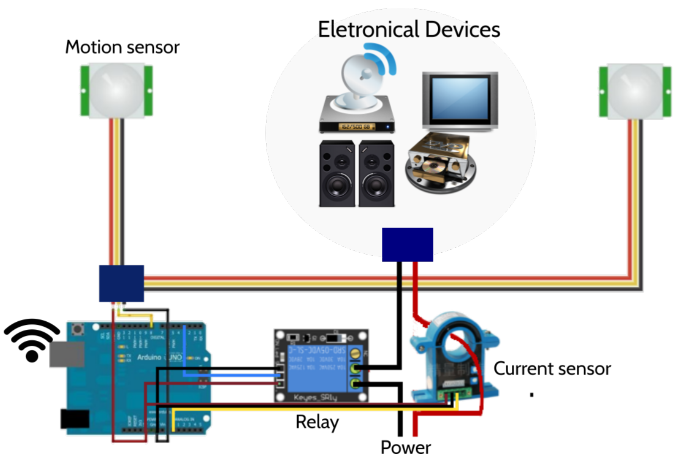


Fig. 1. General scenario of home appliances.

In order to make our devices an IoT enabled device, we used a Smart Gateway, which was developed and evaluated by Andrade et al. [12]. As shown in Figure 2, this gateway implements two middlewares: a service-oriented and a message-oriented. On the one hand, the service-oriented part provides communication between applications with services deployed at gateways using an Enterprise Service Bus (ESB) middleware based on OSGi specification. On the other hand, the message-oriented part provides communication between the gateway and devices.

In the architecture shown in Figure 2, the service-oriented middleware provides interfaces for applications to access devices via RESTful Web Services (Apache CXF in Figure 2). In turn, the broker MQTT Mosquitto serves to provide messaging between the gateway and the devices through a MQTT driver integrated within the RESTful Web Services.

The Figure 2 also illustrates the full scenario implemented to store and analyze generated data of sensors in the cloud. We developed an application that uses HTTP requests to RESTful Web Services in order to get data from sensors and store it at a database in the cloud. We decided to store the data in cloud, because we are using a Smart Gateway in a limited device (Raspberry Pi Model B[2]), that does not have enough storage and processing capacity.

The prediction in our system is performed according to the following steps: i) our application sends a message to the Smart Gateway requesting new information about the environment; ii) the Smart Gateway forwards the message to the devices; iii) devices sent the collected data to the Smart Gateway, which forwards it to our application by using HTTP; iv) new data is stored into a database to be latter analyzed by our classification methods; v) classification methods analyzes the stored data and predicts when turning on/off the standby of appliances; iv) this prediction is sent to the Smart Gateway to make a decision aiming at reducing the power consumption. The prediction processed by cloud is sent to Smart Gateway in offline mode, i.e. the decision of prediction (turn on or turn off the standby of appliances) is sent before the moment to act in appliances – this is useful to avoid communication delays.

Finally, in order to store data about movements (presence) and power consumption, we respectively implemented two tables in the cloud database. In the movement table, we store the date time of each register, and the number of seconds when the last motion was detected. In the power table we register the interval of time (begin and end), which data was collected, and store the average of current and accumulated amount of power consumption for each second.

## III. Experimental Setup

Once the evaluation scenario was configured (Section II), our system started monitoring human presence and storing logs into a database configured in the cloud. Up to the analysis presented in this paper, the system run for 22 days, registering 1134 instances in the database.

Before applying the classification methods, it was necessary to preprocess the instances to integrate and standardize the stored information as presented in the next section.

### A. Preprocessing Step

The classification model was trained using the presence log inferred from the instant time when a motion is detected. The instant time was stored as timestamp, therefore it was necessary to transform it considering a constant and discrete time interval.

On the one hand, to exemplify this step, Figure 3 (A) summarizes a stream of raw data as originally collected by the system. On the other hand, Figure 3 (B) shows few instances used by the training process after applying the transformation.

[1] Arduino: http://www.arduino.cc/

[2] https://www.raspberrypi.org/products/model-b/

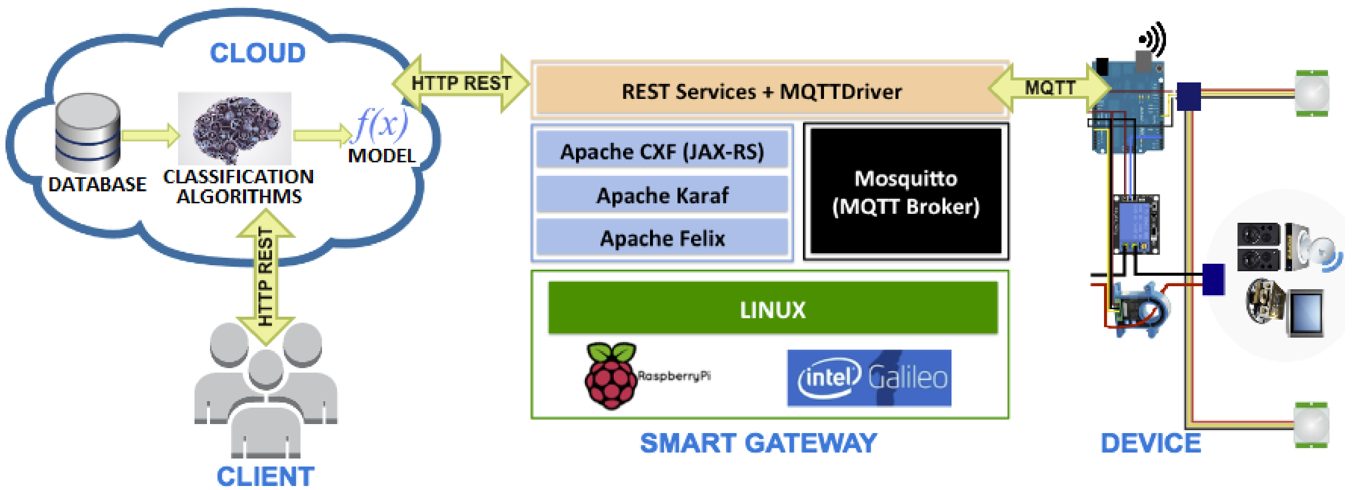


Fig. 2. Scenario overview: gateway, device and client applications with a database in the cloud.

A) Raw data

| id | last_motion | datetime |
|---|---|---|
| 42 | 827 | 2016-04-15 03:52:08.638 |
| 43 | 776 | 2016-04-15 04:07:14.016 |
| 44 | 1676 | 2016-04-15 04:22:15.993 |
| 45 | 27 | 2016-04-15 19:37:23.228 |
| 46 | 44 | 2016-04-15 21:31:29.564 |

B) Pre-processed data

```
month;day;day_of_week;time;presence
4;15;5;15;true
4;15;5;16;true
4;15;5;17;false
4;15;5;78;true
4;15;5;86;true
```

Fig. 3. Sample of human presence log. A) Before preprocessing. B) After preprocessing.

The resultant dataset presented in Figure 3 (B) was obtained by performing the following 4 steps:

1) ID: removed because it adds no useful information;
2) Date resolution: month (from 1 to 12), day (from 1 to 31), and day of week (from 1 to 7);
3) Time resolution: instant time clustered in interval of fifteen minutes: (00:00 AM - 0, 00:15 AM - 1, 00:30 AM - 2: 00:45 AM - 3, 01:00 AM - 4, ..., 11:45 PM - 95);
4) Presence: boolean variable that stores either true, if the system detected any motion within the last 900 seconds (fifteen minutes), or false, otherwise.

After applying these four transformation steps, the resultant dataset was analyzed by the classification methods as presented in the following section.

### B. Classification Methods

As previously stated, the learning problem addressed by this work aims at predicting when devices on standby mode can be turned off to support efficiency in energy consumption. To achieve this goal, we trained a classification model on a dataset characterized by a binary label referred to as "presence". This label indicates if a motion was detected within a time interval.

The classification methods used in this work were: i) *K*-Nearest Neighbor (KNN) [13]; ii) Naïve Bayes [14]; iii) Support Vector Machines (SVM) [15]; and iv) Artificial Neural Network (ANN) [16], [17]. A short definition on these methods is presented next.

KNN is a method based on distance [18], in which every training instance is represented in a space defined by its attributes. The KNN classifies a new instance by analyzing the input space and returns the classes of closest instances.

The Naïve Bayes classifier is a probabilistic method based on the Bayes theorem [19]. This method uses a previous knowledge, which can be combined to collected instances to determine the probability of a hypothesis. The learning task performed by this method combines the probabilities of multiple hypothesis weighted by their probabilities.

The SVM classifier is formally defined by the Statistical Learning Theory [20], which provides a framework to ensure learning conditions in order to construct models, make decisions, and perform predictions. SVM uses Kernel methods to study classification problems, creating classifiers mathematically well-defined and geometrically intuitive. The main advantage of using this classifier is the possibility of learning disregarding the dimensionality of the feature space.

ANN is a bio-inspired method based on central nervous system, which is composed of neural networks. The learning is obtained from this method by adjusting inputs, neurons, weights and layers. In this work, we adopted the backpropagation algorithm, which uses the outputs to update the network information in order to minimize the loss function and, as a consequence, reducing learning error [16].

The application of such methods in our scenario was assessed by using a 10-fold cross validation method, repeated 5 times to reduce effects of overfitting. Parameters for the KNN, SVM and ANN methods were defined after running empirical experiments using Monte Carlo simulation, in which the best ones were chosen to evaluate our approach. In summary, the evaluation was performed using: i) ROC curve; ii) Area Under Curve (AUC); and iii) Analysis on energy consumption reduction.

For this evaluation, we used R language[3] and the library Caret[4], which implements a miscellaneous functions for training and plotting classification and regression models.

## IV. Results

The experiments and studies performed on this work were guided by the following hypothesis: classification methods can be used to model users' behavior in their residences aiming at reducing energy consumption by controlling passive standby devices.

In summary, the obtained model, estimated by classification methods applied on historical data, can be later used to turn off devices saving energy and, consequently, users' money.

In order to prove such hypothesis, this section presents experimental results obtained after applying classification methods on the dataset collected from the scenario detailed in Section II. The obtained results were organized in three sections, as presented next: i) parameter setup; ii) model evaluation; and iii) energy consumption analysis.

### A. Parameter Setup

The first experiments were carried out to identify the best parameters for each classification method. The parameter choice was conducted by using Monte Carlo simulation, widely varying initial values and assessing the results according to the ROC curve and a sensitivity/specificity report.

The first parameter choice was performed to the KNN classifier. The parameter estimated was the number of neighbors ($k$) and the results are shown in Table I. By analyzing this table, we notice the greater the value ($k$) is, the worse the result is. For this reason, we decided to use KNN with $k = 3$.

TABLE I
KKN with different values of k

| k | ROC | Sens. | Spec. |
|---|---|---|---|
| **3** | **0.8411371** | **0.7217045** | **0.8087375** |
| 4 | 0.8379563 | 0.7161174 | 0.8096899 |
| 5 | 0.8346175 | 0.7191477 | 0.8074308 |
| 7 | 0.8195555 | 0.6916098 | 0.7837209 |
| 9 | 0.8131143 | 0.6744886 | 0.7963123 |
| 11 | 0.8069473 | 0.6665909 | 0.7953821 |
| 13 | 0.7961590 | 0.6574242 | 0.8037763 |
| 15 | 0.7818758 | 0.6580682 | 0.7977298 |
| 17 | 0.7688828 | 0.6459470 | 0.8014839 |
| 19 | 0.7682636 | 0.6293371 | 0.8070875 |
| 21 | 0.7678040 | 0.6159280 | 0.8107973 |

[3]R project: https://www.r-project.org/
[4]Caret: http://caret.r-forge.r-project.org

The next experiment was performed to evaluate the parameters used by the SVM algorithm. The kernel adopted in this work was the radial function. In our Monte Carlo simulation, we changed the parameters *sigma* and *C*. *Sigma* represents the width of Gaussian kernels [21] and *C* is related to the optimization, in which avoids misclassifying training examples. Large values of *C* influence the optimization producing hyperplanes with smaller margins. On the other hand, small values of *C* can produce hyperplanes with larger margins. Thus, we varied the *sigma* values from $0.1$ to $0.4$ and the *C* values from $0.25$ to $16$, as presented in Table II. As highlighted in this table, the best result was obtained with *sigma* equals to $0.4$ and *C* equals to $16$.

TABLE II
Support Vector Machines with Radial Kernel with different parameters.

| sigma | C | ROC | Sens. | Spec. |
|---|---|---|---|---|
| 0.1 | 0.25 | 0.7355637 | 0.5576894 | 0.8177187 |
| 0.1 | 0.50 | 0.7264708 | 0.5534280 | 0.8284496 |
| 0.1 | 1.00 | 0.7186582 | 0.5479356 | 0.8396567 |
| 0.1 | 2.00 | 0.7127353 | 0.5387689 | 0.8494241 |
| 0.1 | 4.00 | 0.7104302 | 0.5253030 | 0.8596899 |
| 0.1 | 8.00 | 0.7037453 | 0.5081818 | 0.8676190 |
| 0.1 | 16.00 | 0.7049214 | 0.5026705 | 0.8699668 |
| 0.2 | 0.25 | 0.7214816 | 0.5558712 | 0.8261240 |
| 0.2 | 0.50 | 0.7144061 | 0.5485417 | 0.8363898 |
| 0.2 | 1.00 | 0.7098251 | 0.5369129 | 0.8526910 |
| 0.2 | 2.00 | 0.7053332 | 0.5222917 | 0.8559579 |
| 0.2 | 4.00 | 0.7115323 | 0.5149621 | 0.8559468 |
| 0.2 | 8.00 | 0.7233485 | 0.5198295 | 0.8457032 |
| 0.2 | 16.00 | 0.7306418 | 0.5283712 | 0.8405648 |
| 0.3 | 0.25 | 0.7168290 | 0.5510038 | 0.8307752 |
| 0.3 | 0.50 | 0.7087669 | 0.5393561 | 0.8447508 |
| 0.3 | 1.00 | 0.7090637 | 0.5326515 | 0.8503544 |
| 0.3 | 2.00 | 0.7215082 | 0.5228598 | 0.8499003 |
| 0.3 | 4.00 | 0.7347040 | 0.5259280 | 0.8457032 |
| 0.3 | 8.00 | 0.7474837 | 0.5301515 | 0.8424252 |
| 0.3 | 16.00 | 0.7609659 | 0.5369697 | 0.8577962 |
| 0.4 | 0.25 | 0.7140119 | 0.5473295 | 0.8391694 |
| 0.4 | 0.50 | 0.7107148 | 0.5357386 | 0.8461462 |
| 0.4 | 1.00 | 0.7296676 | 0.5277462 | 0.8466224 |
| 0.4 | 2.00 | 0.7425134 | 0.5247159 | 0.8508306 |
| 0.4 | 4.00 | 0.7561652 | 0.5326136 | 0.8480177 |
| 0.4 | 8.00 | 0.7697257 | 0.5407008 | 0.8624585 |
| **0.4** | **16.00** | **0.7739697** | **0.5498106** | **0.8564120** |

Similarly to previous analysis, the ANN method was performed several times varying the parameters *size* and *decay*. In this context, *decay* defines a penalty to the squared sum of weights, supporting the optimization process and avoiding overfitting [22]. The *size* parameter means the number of units in the hidden layer. Table III shows the results obtained by widely changing these parameters. By looking at this table, we notice the best results were obtained using *size* equals to 7 and *decay* equals to $0.3$.

The next section shows the results obtained after applying the classification methods, with the best parameter choices, on the proposed scenario.

### B. Model Evaluation

In this section, we present the results obtained by using the classification methods and their better parameter configu-

TABLE III
NEURAL NETWORK WITH ALGORITHM BACKPROPAGATION WITH DIFFERENT PARAMETERS

| size | decay | ROC | Sens. | Spec. |
|---|---|---|---|---|
| 1 | 0.001 | 0.6006110 | 0.4309091 | 0.9140421 |
| 1 | 0.100 | 0.7124662 | 0.4887121 | 0.9137320 |
| 1 | 0.200 | 0.7140390 | 0.4917614 | 0.9090919 |
| 1 | 0.300 | 0.7108930 | 0.4935795 | 0.9072204 |
| 1 | 0.400 | 0.7121334 | 0.4960038 | 0.9062791 |
| 3 | 0.001 | 0.6706374 | 0.4655303 | 0.9156035 |
| 3 | 0.100 | 0.7263270 | 0.4947917 | 0.9007420 |
| 3 | 0.200 | 0.7237474 | 0.4959848 | 0.9044075 |
| 3 | 0.300 | 0.7278335 | 0.4978220 | 0.8988261 |
| 3 | 0.400 | 0.7327441 | 0.4971970 | 0.8974308 |
| 5 | 0.001 | 0.6603696 | 0.4821023 | 0.9035105 |
| 5 | 0.100 | 0.7429804 | 0.4942045 | 0.9002326 |
| 5 | 0.200 | 0.7424462 | 0.4990909 | 0.8960133 |
| 5 | 0.300 | 0.7442428 | 0.5026515 | 0.8853488 |
| 5 | 0.400 | 0.7428021 | 0.5014962 | 0.8862348 |
| 7 | 0.001 | 0.6803601 | 0.4795455 | 0.8955814 |
| 7 | 0.100 | 0.7544946 | 0.5125947 | 0.8721262 |
| 7 | 0.200 | 0.7584984 | 0.5315720 | 0.8624252 |
| **7** | **0.300** | **0.7633098** | **0.5168939** | **0.8745958** |
| 7 | 0.400 | 0.7603285 | 0.5124621 | 0.8718051 |

rations. The models estimated for every method were assessed by the ROC curve, as shown in Figure 4.

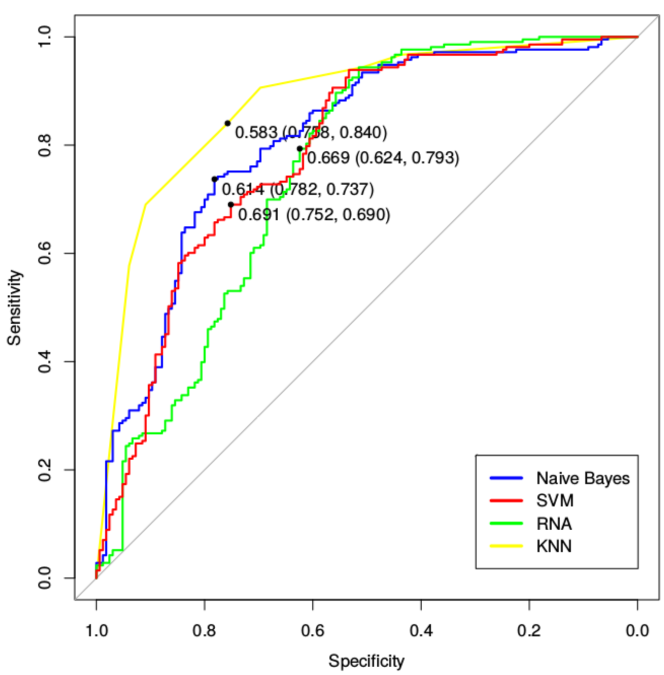


Fig. 4. ROC curve.

According to these results, it is possible to notice the best prediction performance was obtained by KNN. This conclusion is confirmed by analyzing the area under the ROC curve as presented in Table IV, in which we analyzed the threshold, accuracy and AUC.

Finally, we also analyzed the classification methods by using the boxplot presented in Figure 5. Although there is no significant difference among the methods, the KNN has

TABLE IV
PERFORMANCE OF ALGORITHMS THRESHOLD, ACCURACY AND AUC

| Algorithm | Threshold | Accuracy | AUC |
|---|---|---|---|
| **KNN** | **0.5833333** | **0.8042328** | **0.878** |
| Naive Bayes | 0.6139580 | 0.7566138 | 0.8096 |
| RNA back-prop. | 0.6688069 | 0.7195767 | 0.7542 |
| SVM | 0.6905673 | 0.7169312 | 0.7897 |

presented better results, as expected.

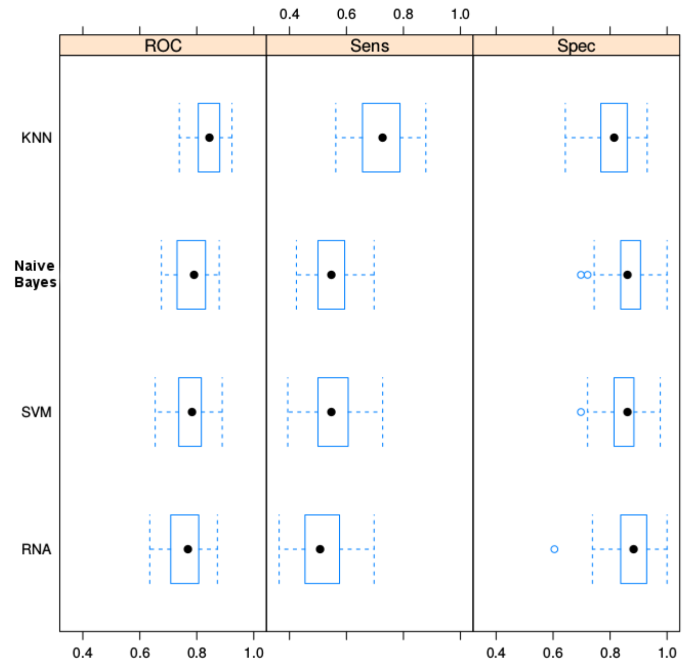


Fig. 5. Boxplot graph with ROC, sensibility and specify.

It is worth to emphasizing that we do not intend to compare the classification methods. All experiments presented in this section were performed to select the best algorithm to be, specifically, applied in our scenario. Therefore, considering the analysis presented in this section, KNN was used to classify new instances and, in the next section, we evaluated whether or not our approach, using IoT and ML, makes possible to save energy.

## C. Energy Consumption Analysis

Finally, in this last experiment, we analyzed whether or not classification methods can be used to save energy by turning off appliances in predicted moments when there is nobody (no motion detected) at home.

Thus, data collected from our IoT scenario, containing weekly motion information, was analyzed by the KNN model estimated in the experiments presented in previous section.

The IoT scenario presented in Section II continuously monitors and collects information about energy consumption. Basically, we use the KNN model to predict if there is someone at home. Based on this prediction, the system can turn off standby devices. If the prediction is right, energy consumption is saved. Otherwise, user can alert the system keeping appliances turned on.

In a whole week, the IoT system registered an accumulated consumption of $11.73$ kWh. Every 15 minutes of appliances on standby mode represents an average consumption of $0.023$ kWh.

By using the KNN classifier, $66.75$ hours were predicted with no activity at home. This number represents $39.73\%$ of all collected data in a week. Once the accuracy of KNN was around $80.42\%$, the energy consumption was $4.94$ kWh per week, representing a reduction of $42.10\%$ over the standby power consumption.

Lee et al. [9] presents an approach that reduces of energy consumption between 27% and 44%. The percent varies according to the time, in which no one was in home, and the number of appliances. However, this time can only be measured by using an App installed in users' smartphones. In contrast, our proposed infrastructure of hardware is minimal. Besides providing better results, we use motion sensor to infer human presence without any extra configuration.

This final result points out the importance of using our approach, which combines IoT technologies and Machine Learning methods, in the analyzed scenario. The energy reduction justifies the use of classification methods, which can be easily applied on real-world scenarios.

## V. Final Remarks

This paper presents an approach to, automatically, control appliances on standby mode. The main contribution of our approach is the combination of Internet of Things (IoT) technologies and classification methods proposed by the Machine Learning area.

In summary, experimental results were important to demonstrate classification methods are suitable to model users' behavior in their residences, reducing energy consumption by controlling passive standby devices. Besides that, we conclude the design of a low cost solution to provide an efficient management of energy is feasible.

The proposed approach still needs to be assessed in others scenarios in order to understand its limitations and advantages in heterogeneous environments. As future works, we intend to implement our proposal in a business environment as, for instance, in a computing lab to evaluate the results in such type of environment. The IoT infrastructure necessary to run our experiments in a computing lab will be as simple as such one presented in Figures 1 and 2, i.e., we just need to add more sensors. Besides this, we are also interested in investigating an online approach to incrementally update the model without decreasing the system performance by retraining the model.

### *Acknowledgments*

Authors would like to thank the following organizations: CAPES, CNPq, and FAPESB for supporting the Graduate Program in Computer Science at the Universidade Federal da Bahia; CNPq (205266/2014-2) for the financial support to Cassio Prazeres; and CAPES (88881.134608/2016-01) for the financial support to Leandro Andrade.

## References


[1] D. Evans, "Cisco internet of things white paper - how the next evolution of the internet is changing everything," in *Cisco Internet Business Solutions Group (IBSG)*, 2011.

[2] K. Ashton, "That 'internet of things' thing," http://www.rfidjournal.com/articles/view?4986, June 2009.

[3] D. Giusto, A. Iera, G. Morabito, and L. Atzori, Eds., *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications*. New York: Springer, 2010.

[4] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelffle, Eds., *Vision and Challenges for Realising the Internet of Things*. European Commission, March 2010. [Online]. Available: http://www.internet-of-things-research.eu/pdf/IoT_Clusterbook_March_2010.pdf

[5] C. Beckel, W. Kleiminger, T. Staake, and S. Santini, "Improving device-level electricity consumption breakdowns in private households using on/off events," *SIGBED*, vol. 9, no. 3, pp. 32–38, 2012.

[6] J. Burgess and M. Nye, "Re-materialising energy use through transparent monitoring systems," *Energy Policy*, vol. 36, no. 12, pp. 4454 – 4459, 2008, foresight Sustainable Energy Management and the Built Environment Project.

[7] J. Ross and A. Meier, "Measurements of whole-house standby power consumption in california homes," *Energy*, vol. 27, no. 9, pp. 861 – 868, 2002.

[8] K. Clement, I. Pardon, and J. Driesen, "Standby power consumption in belgium," in *Electrical Power Quality and Utilisation, 2007. EPQU 2007. 9th International Conference on*, Oct 2007, pp. 1–4.

[9] S. Lee, G. Ryu, Y. Chon, R. Ha, and H. Cha, "Automatic standby power management using usage profiling and prediction," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 6, pp. 535–546, Nov 2013.

[10] G. Zhang, X. Wu, C. Li, L. Tao, X. Ji, H. Zhao, and Y. Cui, "Building electrical equipment internet of things with applications to energy saving," in *Mechatronics and Control (ICMC), 2014 International Conference on*, July 2014, pp. 1022–1026.

[11] D. m. Han and J. h. Lim, "Smart home energy management system using ieee 802.15.4 and zigbee," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1403–1410, Aug 2010.

[12] N. V. de Andrade Jr, J. L. de Almeida, R. D. Costa, L. J. S. Andrade, G. P. Pinto, and C. V. S. Prazeres, "Web of things gateway: A performance evaluation," in *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, ser. WebMedia '15. New York, NY, USA: ACM, 2015, pp. 25–32. [Online]. Available: http://doi.acm.org/10.1145/2820426.2820440

[13] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, January 1967.

[14] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, ser. UAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[16] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[17] P. J. Werbos, *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*. New York, NY, USA: Wiley-Interscience, 1994.

[18] K. Faceli, A. C. Lorena, J. Gama, and A. Carvalho, "Inteligência artificial: Uma abordagem de aprendizado de máquina," *Livros Técnicos e Científicos*, 2011.

[19] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[20] V. N. Vapnik, *Statistical Learning Theory*, 1st ed. Wiley, Sep. 1998.

[21] P.-T. Lin, S.-F. Su, and T.-T. Lee, "Support vector regression performance analysis and systematic parameter selection," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, July 2005, pp. 877–882 vol. 2.

[22] W. Venables and B. Ripley, *Modern Applied Statistics with S*, ser. Statistics and Computing. Springer New York, 2003.