Mining and predicting temperature and smoke sensors data

Mirjana Maksimović
Faculty of Electrical Engineering
East Sarajevo, Bosnia and Herzegovina
mirjana@etf.unssa.rs.ba

Abstract—**Data mining seems to be an effective technique for discovering useful knowledge from a large amount of data observed by many sensors. Prediction in sensor networks can be performed in the way that each sensor node learns a local predictive model for the global target classes, using only its local input data. Only the predicted target class for each reading is then transmitted to the gateway or to the base station. One important class of such algorithms is classifiers, which use the sensor inputs to predict some output function of interest. The purpose of this paper is to analyze different classification algorithms in the case of temperature and smoke sensors and to see which of the methods and parameter values work best for the given problem. Datasets used in the experiments are fuzzy logic generated data and full and incomplete rule base data are considered in order to determine fire confidence.**

Key words - data mining; fuzzy logic; temperature; smoke; sensors; WEKA;

I. INTRODUCTION

Data mining is defined as the process of discovering useful patterns or knowledge from different data sources [1]. The main goal of data mining techniques is to find and describe the structural patterns in the data in order to attempt to explain connections between data and create predictive models based on them [2]. Data mining is a multidisciplinary field which includes machine learning, statistics, databases, artificial intelligence, information theory and visualization [1]. Input data for applying data mining techniques are presented in the form of a set of examples, and the output can be expressed in the prediction or description form of the analyzed data structure. There are four the most common tasks used in data mining applications: supervised learning (or classification), unsupervised learning (or clustering), association rule mining, and sequential pattern mining. Each of them is characterized by different styles of learning.

The process of data mining is consisted of three basic steps: preprocessing, application of data mining algorithms and processing of the obtained patterns or knowledge:

- Preprocessing – the raw data must be cleaned in order to become suitable for mining. Data cleaning includes removing noises and abnormalities, handling too large data, identifying and removing irrelevant attributes, and so on. Data cleaning is procedure that usually consumes a lot of time and it is very labor-intensive but it is absolutely necessary step for successful data mining.
- Data mining – the process of applying data mining algorithm that will produce patterns or knowledge.
- Post–processing – Among all discovered patterns or knowledge, it is necessary to discover ones that are useful for the application. For making the right decision there are many evaluation and visualization techniques that can be used.

In recent years, a lot of research is made in the field of sensor data mining. It is a relatively new area which brings numerous challenges with it in the context of data collection, storage and processing. In other words, a variety of data mining methods such as clustering, classification, frequent pattern mining, and outlier detection are often applied to sensor data in order to extract actionable insights. The main task in creating a successful prediction model is to identify factors that influence the selection of class variable and data mining algorithm that gives the best results in the observed data set. Sensor data usually needs to be compressed and filtered for more effective mining and analysis.

In this paper datasets used for the experiments are based on fuzzy logic. For determining fire confidence temperature, temperature difference, smoke obscuration and smoke obscuration difference measurement are used.

The rest of this paper is organized as following. Second section presents data preparation while third section provides classification algorithms implementations. Comparative analysis of selected data mining techniques and the experimental results are shown in fourth section. Fifth section gives the conclusion.

II. DATA PREPROCESSING

Data preprocessing is a procedure that usually consumes a bulk of time and requires a lot of work, but it is an absolutely necessary step for the successful application of data mining techniques and algorithms [2].

Integration of soft computing technologies like fuzzy logic in sensor nodes can significantly lead to network performances improvements because it provides effective parameter combination, and it is able to be executed in the

low-resourced nodes that compose wireless sensor networks (WSNs) [3].

Like many other human-recognizable events, the phenomenon fire has no real meaning to a sensor node. Therefore, suitable techniques that would allow describing events in ways that sensor nodes would be able to "understand" are needed. One of them is fuzzy technique. What makes fuzzy logic suitable for use in WSNs is that it can tolerate unreliable and imprecise sensor readings, it is much closer to human way of thinking than crisp logic and compared to other classification algorithms based on probability theory, fuzzy logic is much more intuitive and easier to use. It allows using linguistic variables whose values are not numbers but words or sentences in a natural or artificial language [4]. Fuzzy rules are conditional statements in the form of IF-THEN and based on expert knowledge.

In this work the experiment for fire detection is based on approach given in [5, 6]. Detection of fire is usually based on temperature and smoke sensors readings. Thus, fire confidence determination can be based on temperature, temperature difference, smoke obscuration and smoke obscuration difference measurement. Temperature and smoke sensors will generate an alarm condition if the temperature and smoke within the protected area reaches a predetermined level and when the temperature and smoke rises at a rate exceeding a predetermined value, respectively. Instead of using crisp values, fuzzy logic proposes use of linguistic variables. Therefore, data obtained from proposed detectors according to fuzzy technique and thresholds, for the purpose of the experiment are described with values: *low* (L), *medium* (M) and *high* (H), presented with membership functions shown in Fig. 1 [5]. Due to their simple formulas and computational efficiency, both triangular and trapezoidal membership functions have been used extensively, especially in real-time implementations as it is fire detection.##

#

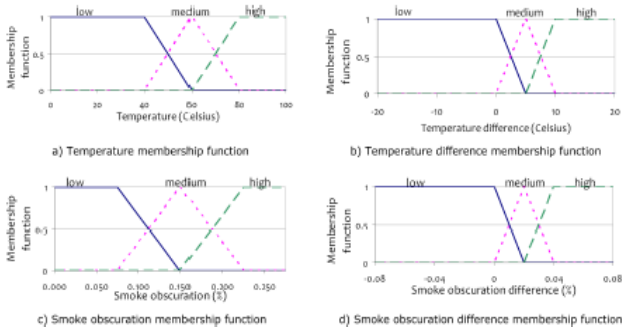

Figure 1. Membership functions of input variables

Fire confidence in this experiment is defined as output variable and is also described with *low* (L), *medium* (M) and *high* (H) linguistic variables as it shown in Fig. 2 [6].

With 4 input variables, each of which can take 3 values, the number of rules in the full fuzzy rule-base of experiment is 81 (3*3*3*3) (with n variables each of which can take m values, the number of rules in the rule-base is m^n) [5]. Table 1 shows first 10 rules.

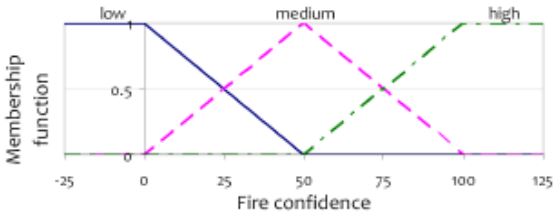

Figure 2. Membership functions of output variable

TABELA I. THE FULL RULE BASE OF FIRE DATA TEST (FIRST 10 RULES)

Temperature	Temperature difference	Smoke obscuration	Smoke obscuration difference	Fire confidence (class)
L	L	L	L	L
L	L	L	M	L
L	L	L	H	L
L	L	M	L	L
L	L	M	M	L
L	L	M	H	L
L	L	H	L	L
L	L	H	M	L
L	L	H	H	M
L	M	L	L	L

The main disadvantage of using fuzzy logic in WSN is a large number of rules what is not appropriate for energy and memory limited sensor node devices. The number of rules grows exponentially to the number of variables [5]. Since sensor nodes have limited memory, storing a full rule base on every node might not be reasonable. Also, constantly traversing a large rule base might considerably slow down the event detection. In other words, the massive streams of sensor data which could be generated in fire detection applications make it impossible to use algorithms that must store the entire data into main memory. For that purpose, on full rule base consisted of fuzzy rules for detection of fire, shown in Table I, an incomplete rule base technique is applied. In the Table I there are rules for every possible combination of the input variables. However, only some of these combinations have outcomes that are important to the particular application. Thus, excluding the rules with consequents that are of no interest it is possible to reduce the size of the rule base even more. As a result, by lowering the level of completeness of the rule base, the number of rules that should be stored by the sensor nodes can be significantly decreased. This "trimming" process, however, should be performed very carefully in order to prevent the removal of important consequents [5]. In this work, all outcomes *low* (L) are excluded from the full rule base and incomplete rule base of 65 rules is shown in Table II. In other words, the normal values are discarded and only anomaly values will be transmitted to the central server. Thus, the number of sensors that need to report their measurements will be reduced by reducing both node activity and bandwidth. It should also lead to faster

response time of detectors what will directly influence on response time of fire protection system entirely.

TABELA II. THE INCOMPLETE RULE BASE OF FIRE DATA TEST (FIRST 10 RULES)

Temperature	Temperature difference	Smoke obscuration	Smoke obscuration difference	Fire confidence (class)
L	L	H	H	M
L	M	L	M	M
L	M	L	H	M
L	M	M	L	M
L	M	M	M	M
L	M	M	H	M
L	M	H	L	M
L	M	H	M	M
L	M	H	H	H
L	H	L	H	M

For further analysis two Excel .csv data files are formed based on data given in Table I. and Table II. The next step is their exporting to WEKA data mining tool [7] in order to apply chosen classification algorithms presented in rest of the paper.

III. CLASSIFICATION ALGORITHMS IMPLEMENTATIONS

Supervised learning or classification is type of machine learning analogoue to human learning from past experiences to gain new knowledge in order to improve ability to perform real-world tasks [1]. Computers using machine learning learns from data which are collected in the past and represent past experiences. In most cases classification is used for learning a target function that can be used to predict the values of a discrete class attribute, e. g. classification is one type of predictions methods. The goal of prediction is to infer a target attribute, predicted variable, from some combination of other aspects of the data, another attribute. Classification means the problem of correctly predicting the probability that an example has a predefined class from a set of attributes describing the example. In classification learning, the learning scheme is presented with a set of classified examples from which it is expected to learn a way of classifying unseen examples [2].

There are many methods and measures for estimation the strength and the accuracy of a classification/predictive model [1]. The main measure is the classification accuracy which is the number of correctly classified instances in the test set divided by the total number of instances in the test set. Some of the common methods for classifier evaluation are holdout set, Multiple Random Sampling and Cross-Validation. In applications with only two classes two measures named Precision and Recall are usually used. Their definitions are:

$$P = \frac{TP}{TP + FP} \tag{1}$$

$$R = \frac{TP}{TP + FN} \tag{2}$$

TP, FP and FN used in Eq. (1) and Eq. (2) are the numbers of true positives, false positives and false negatives, respectively. These measures can be also used in case of larger number of classes, which in this case are seen as a series of problems with two classes. It is convenient to introduce these measures using a confusion matrix. A confusion matrix contains information about actual and predicted results given by a classifier. However, it is hard to compare classifiers based on two measures, which are not functionally related [1]. If a single measure to compare different classifiers is needed, the F-measure is often used:

$$FM = \frac{2 \cdot P \cdot R}{P + R} \tag{3}$$

Another measure is the receiver operating characteristic (ROC) [2]. It is a term used in signal detection to characterize the tradeoff between hit rate and false-alarm rate over a noisy channel. ROC curves depict the performance of a classifier without regard to class distribution or error costs. They plot the true positive rate on the vertical axis against the true negative rate on the horizontal axis.

There are many other methods and measures for estimation the strength and the accuracy of a classification/predictive model. Performance measures for numeric prediction are given in Table III.

TABELA III. PERFORMANCE MEASURES FOR NUMERIC PREDICTION

Mean-squared error	$\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n}$
Root mean-squared error	$\sqrt{\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n}}$
Mean-absolute error	$\frac{\lvert p_1 - a_1\rvert + \ldots + \lvert p_n - a_n\rvert}{n}$
Relative-squared error*	$\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(a_1 - \bar{a})^2 + \ldots + (a_n - \bar{a})^2}$
Root relative-squared error*	$\sqrt{\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(a_1 - \bar{a})^2 + \ldots + (a_n - \bar{a})^2}}$
Relative-absolute error*	$\frac{\lvert p_1 - a_1\rvert + \ldots + \lvert p_n - a_n\rvert}{\lvert a_1 - \bar{a}\rvert + \ldots + \lvert a_n - \bar{a}\rvert}$
Correlation coefficient**	$\frac{S_{PA}}{\sqrt{S_P S_A}}$, where $S_{PA} = \frac{\sum_i (p_i - \bar{p})(a_i - \bar{a})}{n-1}$, $S_P = \frac{\sum_i (p_i - \bar{p})^2}{n-1}$, $S_A = \frac{\sum_i (a_i - \bar{a})^2}{n-1}$

*Here, $\bar{a}$ is the mean value over the training data.
**Here, $\bar{a}$ is the mean value over the test data.

As it is already stated, one of the most common tasks used in data mining applications is the classification. In this paper the classifiers from WEKA data mining tool [7] are used in order to analyze the classification accuracy of generated data. A several algorithms must be applied to application before a suitable algorithm for selected data types can be found. Online algorithms provide an attractive alternative to conventional batch algorithms for handling large data sets. The selection of a correct data mining algorithm depends on not only the goal of an application, but also on the compatibility of the data set.

The following algorithms were used for classification purposes in this work:

Algorithm 1: Furia
Algorithm 2: Naïve Bayes
Algorithm 3: Decision Tree classifier
Algorithm 4: Support Vector Machines – SVM
Algorithm 5: Neural Network classifier

The applied algorithms are shortly described in rest of the paper. More information about them can be found in [2, 8].

A. *FURIA*

FURIA (Fuzzy Unordered Rule Induction Algorithm) is a fuzzy rule-based classification method proposed in 2009 [9]. FURIA extends the well-known RIPPER algorithm preserving its advantages (simple and comprehensible rule sets) and includes a number of modifications and extensions. It obtains fuzzy rules instead of the usual strict rules, as well as an unordered rule set instead of the rule list. Moreover, to deal with uncovered examples, it makes use of an efficient rule stretching method (generalizing the existing rules until they cover the example).

B. *Naïve Bayes*

Bayesian classifier is statistical classifier which can be used to predict class membership probabilities. Bayesian classification is based on Bayes theorem and "naïvely" assumes independence - it is only valid to multiply probabilities when the events are independent [10]. For each class value it estimates the probability that a given instance belongs to that class. Naïve Bayes gives a simple approach, with clear semantics, to representing, using, and learning probabilistic knowledge and it can achieve impressive results [2].

C. *Decision Tree Classifier*

The decision tree classifier is a tree based classifier which selects a set of features and then compares the input data with them. Learned patterns are represented as a tree where nodes in the tree embody decisions based on the values of attributes and the leaves of the tree provide predictions. The main advantage of a decision tree classifier is its classification speed. WEKA uses the J48 decision tree which is an implementation of the C 4.5 algorithm [8].

D. *Support Vector Machines*

The support vector machine (SVM) classifier works by generating functions from the input training data. This function is used as a classification function. They operate by finding a hypersurface in the space of possible inputs. This hypersurface will attempt to split the positive examples from the negative examples i.e., *low* from *high.* If the dimensionality of the input data is high then the SVM takes more time for training [8].

E. *Neural Network Classifier*

A neural network is a collection of neurons like processing units with weighted connection between them. It is composed of many elements, called nodes which are connected in between. The connection between two nodes is weighted and by the adjustment of this weight, the training of the network is performed. The neural network classifier is used for many pattern recognition purposes. It uses the backpropogation algorithm to train the network [2]. There are many advantages of neural networks such as adaptive learning ability, self-organization, real time operation and insensitivity to noise [10].

In this paper, the output of the simulator is used to learn the difference between a subject that is *low*, *medium* and *high*, in the first experiment and *medium* and *high* in the second one. The question of predicting performance based on limited data is an interesting, and still controversial, one. Repeated cross-validation technique is probably the method of choice in most practical limited-data situations. Different testing strategies can be used to train and test based on the given datasets. In the averaging process, given dataset is divided into two parts. One part is first used to train the classification algorithm. So the percentage of data to be used for training purposes should be specified first (using option *Percentage Split* in WEKA) [2]. Then concepts learned during the training process are used to test the remaining data. For the experiments performed in this paper 10-fold cross validation testing technique is used. During the process the data set is divided into 10 subsets and the classification algorithms are fed with these subsets of data. The left-out subsets of the training data are used to evaluate classification accuracy. When seeking an accurate error estimate, it is standard procedure to repeat the cross-validation process 10 times (10 times tenfold cross-validation) and average the results. This involves invoking the learning algorithm 100 times on datasets that are all nine-tenths the size of the original. Getting a good measure of performance is a computation-intensive undertaking [2].

IV. SIMULATION RESULTS

After analyzing each of implemented data mining techniques, next step is to make a comparative analysis between FURIA, Naïve Bayes, J48, SVM and Neural networks. For this purpose the Experimenter interface of WEKA is used. To make a comparative analysis it is necessary to prepare experiment going through several steps shown in Fig. 3.

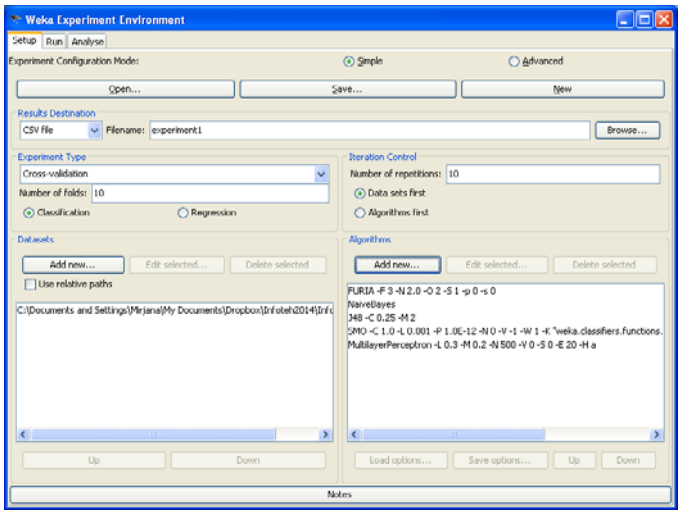

Figure 3. Setting up the experiment

After choosing dataset and classifiers, experiment can be started. A brief report is displayed when the run step is finished. The results are shown in CSV file and can be read directly into a spreadsheet (Fig. 4).

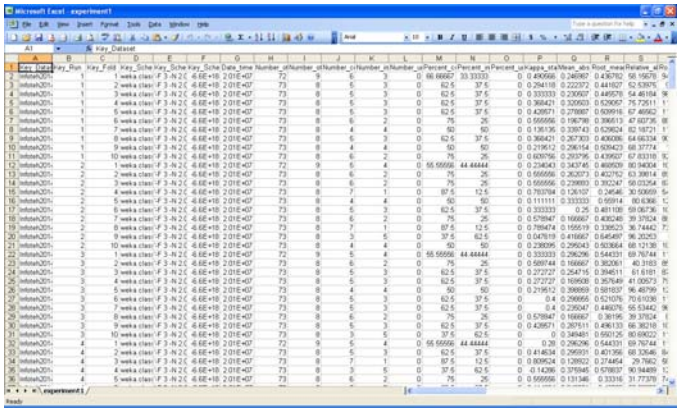

Figure 4. The spreadsheet with results

Each row of file, shown in Fig. 4, represents 1 fold of a 10 fold cross-validation. The cross-validation is run 10 times (the Run column) for each classifier (the Scheme column). Thus, the file contains 100 rows for each classifier, which makes 500 rows in all (plus the header row). Each row contains plenty of information, including the options supplied to the machine learning scheme; the number of training and test instances; the number (and percentage) of correct, incorrect, and unclassified instances; the mean absolute error and the root mean-squared error; and much more. Even there is a great amount of information in the spreadsheet, it is still hard to answer which of the methods and parameter values work best for the given problem.

Analyzing the results is the final step in these experiments. The Analyze panel is shown in Fig. 5. To analyze the experiment that has just been performed, the Experiment button at the top right should be choose. FURIA classifier is chosen as baseline scheme. After clicking on Perform test button, the result of a statistical significance test of the performance of the FURIA versus the other four is displayed in the large panel on the right of Fig. 5. Now it is possible to compare the percent correct statistic.

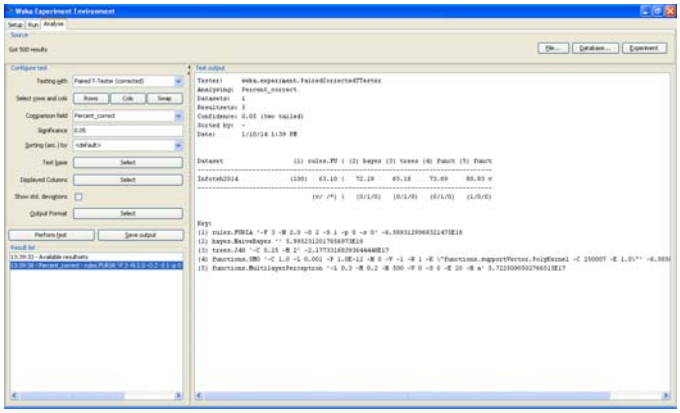

Figure 5. Statistical test results for the experiment

As it can be seen in Fig. 5, the five methods are displayed horizontally, numbered from (1) to (5), as the heading of a little table. The inscrutable integers beside the scheme names identify which version of the scheme is being used. They are present by default to avoid confusion among results generated using different versions of the algorithms. The value in brackets at the beginning of the data row (100) is the number of experimental runs: 10 times tenfold cross-validation.

The percentage correct for the five classifiers is shown in Fig. 5: 63.10% for FURIA, 72.18% for Naïve Bayes, 65.18% for J48, 73.69% for SVM and 80.83% for Neural networks. The symbol placed beside a result indicates that it is statistically better (v) or worse (*) than the baseline scheme - in this case FURIA - at the specified significance level (0.05, or 5%). The corrected resampled t-test is used here. At the bottom of columns 2 and 3 are counts (x/y/z) of the number of times the scheme was better than (x), the same as (y) or worse than (z) the baseline scheme on the datasets used in the experiment. The results indicate that Neural networks are statistically better than FURIA.

Experiment performed on incomplete rule base indicates that the percentage correct for the five classifiers is: 67% for FURIA, 76.88% for Naïve Bayes, 74.69% for J48, 80.12% for SVM and 86.76% for Neural networks.

Using the *Test base* menu it is possible to change the baseline scheme. The performed test was on the error rate. Other statistics can be selected from Comparasion field dropdown menu instead: percentage incorrect, percentage unclassified, root mean-squared error, the remaining error measures and various entropy figures. It is also possible to see the standard deviation of the attribute being evaluated by ticking the *Show std deviations* checkbox [2, 7].

Apart from the learning schemes, there are two other choices in the *Select base* menu: *Summary* and *Ranking*.

The *Summary* compares each learning scheme with every other scheme and prints a matrix with cells that contain the number of datasets on which one is significantly better than the other (Fig. 6).

```
   a     b     c     d     e  (No. of datasets where [col] >> [row])
   - 2 (0) 2 (0) 2 (0) 2 (2) | a = (1) rules.FURIA '-F 3 -N 2.0 -O 2 -S 1 -p 0 -s 0' -6.5893129968321475E18
0 (0)     - 0 (0) 2 (0) 2 (0) | b = (2) bayes.NaiveBayes '' 5.9952312017856973E18
0 (0) 2 (0)     - 2 (0) 2 (1) | c = (3) trees.J48 '-C 0.25 -M 2' -2.1773316839364448E17
0 (0) 0 (0) 0 (0)     - 2 (0) | d = (4) functions.SMO '-C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K \"functions.supportVector.P
0 (0) 0 (0) 0 (0) 0 (0)     - | e = (5) functions.MultilayerPerceptron '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a' 5.72250905027
```

Figure 6. The *Summary* comparative analysis of both tests

The *Ranking* ranks the schemes according to the total number of datasets that represent wins (>) and losses (<) and prints a league table. The first column in the output gives the difference between the number of wins and the number of losses (Fig. 7).

```
>-<   >   <  Resultset
  3   3   0  functions.MultilayerPerceptron '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a' 5.72250905027665 15E17
  0   0   0  functions.SMO '-C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K \"functions.supportVector.PolyKernel -C 250007 -E 1.0\"
  0   0   0  bayes.NaiveBayes '' 5.9952312017856973E18
 -1   0   1  trees.J48 '-C 0.25 -M 2' -2.1773316839364448E17
 -2   0   2  rules.FURIA '-F 3 -N 2.0 -O 2 -S 1 -p 0 -s 0' -6.5893129968321475E18
```

Figure 7. The *Ranking* comparative analysis of both tests

V. CONCLUSION

Data mining in sensor networks is the process of extracting application-oriented models and patterns with acceptable accuracy from a continuous, rapid, and possibly non ended flow of data streams from sensor networks. This paper focuses on comparative analysis of various data mining techniques and

algorithms with primary goal to see which of them has the best classification accuracy and is the most appropriate for a particular application of fire detection uncovering useful information hidden in temperature and smoke sensor data. Experiments were performed on two types of the datasets, based on full and incomplete fuzzy logic rule base.

Obtained results show that Neural network classifier generates the best prediction models on full rule base datasets. The best performances Neural networks have shown and in the case of incomplete rule base what means that by using Neural Networks classifier it can be created more precise model compared to other algorithms. The *Summary* and *Ranking* comparative analysis of both datasets have confirmed that Neural networks are statistically better than other applied algorithms.

Even applied data mining techniques are efficient, none of them can be considered as unique or general solution for each case. On the contrary the selection of a correct data mining algorithm depends of an application and the compatibility of the observed data set. Thus, each situation should be considered as a special case and choice of adequate predictor or classifier should be performed very carefully based on empirical arguments. This kind of analysis provide an opportunity for data mining researchers to develop more advanced methods for handling some of the issues specific to sensor data.

LITERATURE

[1] B. Liu, Web DataMining - Exploring Hyperlinks, Contents, and Usage Data, Springer-Verlag Berlin Heidelberg, 2007

[2] I. H. Witten, E. Frank, M.A. Hall, Data mining: practical machine learning tools and techniques, 3rd edition, Elsevier, 2011

[3] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases", World Scientific Publishing: Singapore, Volume 19., 2001

[4] H.J. Zimmermann, Fuzzy Set Theory and Its Applications-3rd Edition, Kluwer Academic Publishers, 1996

[5] K. Kapitanova et al., "Using fuzzy logic for robust event detection in wireless sensor networks, " Ad Hoc Netw. , 2011

[6] K. Kapitanova, S.H. Son, and K.D. Kang, "Event Detection in Wireless Sensor Networks -Can Fuzzy Values Be Accurate?", Ad Hoc Netw., Vol. 49, pp 168-184, 2010

[7] WEKA data mining tool, Available: www.cs.waikato.ac.nz/ml/WEKA

[8] J. Natajaran, Simulation of sensor responses of advanced security systems, Master Thesis, University of Texas at Arlington, 2006

[9] J. Hühn and E. Hüllermeier, "FURIA: An Algorithm For Unordered Fuzzy Rule Induction," Data Mining and Knowledge Discovery, 19, 293-319, 2009

[10] S. Garg, A.K. Sharma, "Comparative Analysis of Data Mining Techniques on Educational Dataset", International Journal of Computer Applications (0975 – 8887) Volume 74– No.5, July 2013