Testing and Analysis of Activities of Daily Living Data with Machine Learning Algorithms

Ayse Cufoglu and Adem Coskun
Faculty of Science and Technology
Department of Engineering
University of Westminster
London, UK

***Abstract*—It is estimated that 28% of European Union's population will be aged 65 or older by 2060. Europe is getting older and this has a high impact on the estimated cost to be spent for older people. This is because, compared to the younger generation, older people are more at risk to have/face cognitive impairment, frailty and social exclusion, which could have negative effects on their lives as well as the economy of the European Union. The 'active and independent ageing' concept aims to support older people to live active and independent life in their preferred location and this goal can be fully achieved by understanding the older people (i.e their needs, abilities, preferences, difficulties they are facing during the day). One of the most reliable resources for such information is the Activities of Daily Living (ADL), which gives essential information about people's lives. Understanding this kind of information is an important step towards providing the right support, facilities and care for the older population. In the literature, there is a lack of study that evaluates the performance of Machine Learning algorithms towards understanding the ADL data. This work aims to test and analyze the performance of the well known Machine Learning algorithms with ADL data.**

***Keywords*—Activities of Daily Living (ADL); Machine Learning (ML); Classification Algorithms; Active and Independent Aging**

I. Introduction

Europe is becoming older as a result of lower birth rates and higher life expectancy. According to Eurostat [1], older people will form 28% of the European Union's (EU's) population by 2060 and people that are aged 80 and over will rise from 5% to 12% which is very close to the percentage (around 15%) that is estimated for young people (aged 0-14) in 2060. It is known that older population is more vulnerable towards cognitive impairment, frailty and social exclusion, which could negatively effect the health-care system as well as older people's Quality of Live (QoL) and independence. This reality triggered many investments that are focused on the areas that promote and facilitate 'active and independent ageing', which aims to provide the needed support (i.e. medical support, monitoring services) to let older people live active and independent life in their preferred locations. Activities of Daily Living (ADL) is one of the important parameters that gives essential information about people's lives. Understanding ADL information of older people is an important step forward understanding their:

- needs and abilities,
- difficulties they are facing during their everyday lives
- detecting new medical conditions to provide the right support and personalized care for them.

This work aims to test and analyze the performance of the well known Machine Learning (ML) algorithms with ADL data. This study will provide an opportunity to understand how ML algorithms cope with the ADL data with respect to learning, analyzing, understanding and classifying the ADL activities. Following ML algorithms have been considered for this study: Instance Based Learner (IBL), K Nearest Neighbour (KNN or IBK), KStar (K*), J48, Locally Weighted Learning (LWL) and Naive Bayesian Tree (NBTree).

This paper is organized as follows; Section II provides a literature review for this work. Section III provides details about the ML algorithms that are tested and analyzed within this study. Section IV presents an information about the datasets that have been used for the simulations. Information about the evaluation measures are presented in Section V. Section VI presents the simulation results and their evaluations. Finally, Section VII provides conclusion and future works for this study. Throughout this paper records and instances are used as synonyms.

II. Background

Activities of Daily Living (ADL) defines the activities we perform in our daily living, such as self care (i.e. feeding ourselves, bathing, dressing, grooming), work and leisure activities [2]. Statistics about the EU's older people population for the coming years influenced many works within the health-care domain. There have been various works in the literature focused at the ADL information (i.e. ADL information capture and analysis). For instance, work by Cheng et al. [3] proposed an ADL recognition engine, called the Adaptive Learning Hidden Markov Model (ALHMM), based on Hidden Markov Model combining the Viterbi and Baum-Welch algorithms for enhanced accuracy and learning capabilities. According to the authors, their proposed model provides an effective and practical solution for ADL recognition. The ALHMM's inference efficiency outperforms the Hidden Markov Model (HMM). In another work [2], Fleury et al. classify the data collected through various sensors, such as temperature, hygrometry and infrared presence sensors, into one of the ADL (hygiene, toilet use, eating, resting, sleeping, communication, and dressing/undressing) using Support Vector Machines (SVM) classifier. The ADLs of 13 young and healthy volunteers have

been recorded and used in this study. Results presented in this work showed that the SVM with polynomial kernel achieved 75% accuracy while SVM with Guassian kernel achieved 86%. In another work Tapia et al. [4] proposed an alternative sensing system to detect activities in real homes. Different from the traditional sensing technology, the proposed system uses simple, low cost "tape on and forget" sensors to recognize individual's activities. In this study, the authors used Naive Bayesian (NB) algorithm for activity detection. Here, NB extended to incorporate temporal relationships among sensor firings and recognize activities. The data that has been used for this study collected from two subjects within two home (one-bedroom apartments) settings. Here, one home was fitted with 77 sensors and the other with 84 sensors. The experimental testing with small dataset showed that the detection accuracies ranged from 25% to 89% depending on the evaluation criteria used.

As it can be seen from the aforementioned works, there is a lack of work in the literature that compares different ML algorithms with respect to their performance with ADL data. Considering this gap in the literature, this paper focuses on the testing, analysis and comparison of the well-known ML algorithms' performance on ADL data.

III. Machine Learning Algorithms

For this work, performance of IBL, KNN, KStar (K*), LWL, J48 and NBTree classification algorithms have been analysed with the ADL dataset. Following paragraphs give information about each of these algorithms.

A. Lazy Classifiers

1) Instance Based Learner (IBL): The Nearest Neighbour (NN) classifier uses normalized Euclidean distance to compare each test instance with the training instances. The closest training instance predicted to has the same class label with the test instance [5] [6] [7]. In case of more than one training instance qualified as the closest, the class label of the first one is assigned to be the class label of the test instance [8]. The IBL is a comprehensive form of NN. In IBL the comparison between the test instance and the training instance is done as follows;

Assume training instance $X_i = \{x_i(1), x_i(2), ..., x_i(B)\}$ and test instance $Y_j = \{y_j(1), y_j(2), ..., y_j(B)\}$. Here comparison between training instance and test instance is done feature by feature as;

- If feature is numeric,

$$g(y_j(b), x_i(b)) = (y_j(b) - x_i(b))^2 \tag{1}$$

- If feature is symbolic,

$$g(y_j(b) - x_i(b)) = \begin{cases} 0, if & y_j(b) = x_i(b) \\ 1, if & y_j(b) \neq x_i(b) \end{cases} \tag{2}$$

where the $g(y_j(b), x_i(b))$ function shows the similarity between b values of the training and test instances and the distance is calculated as;

$$dist(Y_j, X_i) = \sqrt{\sum_{b=1}^{B} g(y_j(b), x_i(b))} \tag{3}$$

2) K Nearest Neighbor (KNN): K Nearest Neighbor (KNN), also known as IBK, is one of the well known IBL algorithm, where different from IBL, K closest instances are retrieved and the label of the majority class among these instances is assigned as the class label for the test instance [5] [6] [9]. The following paragraphs are associated with IBK:

- If the class attribute is symbolic then the class label of the test instance Y_j is same as the class label of the highest vote among the K nearest neighbours. For a scenario, where $K = 3$, if the three nearest neighbours X_1, X_2 and X_3 belong to the classes, C_1, C_1 and C_2 respectively, then C_1 is assigned as the class label for Y_j ,$Y_j \epsilon C_1$, since C_1 is the predominant class label among nearest neighbours.
- If class attribute is numeric then the class label of the test instance will be the mean of the nearest neighbours. Following the above assumptions, label of $Y_j, L(Y_j)$, is calculated as;

$$L(Y_j) = \frac{\sum_{m=1}^{K} L(X_m)}{K} \tag{4}$$

where $L(X_m)$ represents the class label of X_m.

3) KStar (K):* K* is based on entropy distance measure where the distance between two instances is defined as the complexity of transforming one instance into another [6] [10]. This complexity calculation is done in two steps. First step is to define the finite set of transformations that map instances to instances. A 'program' which transforms one instance Y_j to another instance X_i is a finite sequence of transformations starting at Y_j and terminating at X_i [10]. Kolmogorov is one of the well known entropy distance measures where the distance between two instances is the shortest string connecting them [10]. Hence, this approach is focused on the shortest transformation out of many possible transformations. Here, the resulted distance measure is very sensitive to small changes. For this problem K* is defined as the distance of summing all possible transformations between two instances.

4) Locally Weighted Learning: Locally Weighted Learning (LWL) is a weighted IBL that assigns weights to instances using the IBL and uses these locally weighted training instances for classification [11] [12]. While IBK performs local approximation for each test instance Y_j, LWL performs explicit approximation of $L(Y_j)$ for region surrounding Y_j by fitting linear function and quadratic to K nearest neighbours.

B. Decision Trees

Decision Trees (DTs) are data structures that can examine the data and induce a classification tree and its rules to make predictions [13]. A successful classification with the DTs requires well-defined classes and pre-classified training data [14]. The classification accuracy on the training data set and the size of the tree affect the quality of the DT. Construction of the tree model incorporates into two-phases; building phase and pruning phase. The building phase includes a series of divisions of the training dataset that is carried out based on the decision rules [14]. This partitioning is continued until the resulted classes have homogenous instances. In the pruning phase, on the other hand, the nodes that may cause over fitting and low accuracy are pruned [6] [14].

1) Naive Bayesian Tree (NBTree): Naive Bayesian (NB) classifier is one of the Bayesian classifier algorithms. In many works it has been proven that NB classifiers are one of the most computationally efficient and simple algorithms for ML and DM applications [15]-[19]. Unlike BN, NB classifiers assume that all attributes within the same class are independent, given the class label. Based on this assumption, which also reduces the computational complexity of BN classifier, the NB classifier modifies the Bayesian rule as follows [16] [20]:

- Assume Cl_b as class label where $Cl_b = \{Cl_1, Cl_2, ..., Cl_B\}$ and $b = 1, 2, ..., B$
- Assume Y_j as unclassified test instance where $Y_j = \{y_j(1), y_j(2), ..., y_j(A)\}$ for $k = 1, 2, ..., A$

Based on these assumptions the NB is calculated as;

$$P(Cl_b|Y_j) = \arg\max_{Cl_b} P(Cl_b)P(y_j(1), y_j(2), ..., y_j(A)|Cl_b) \tag{5}$$

$$P(Cl_b|Y_j) = \arg\max_{Cl_b} \Pi_{k=1}^{A} P(y_j(k)|Cl_b) \tag{6}$$

NBTree classifier generates a DT with NB classifier at the leaves [6].

2) J48: J48 is another DT algorithm. The J48 classification algorithm is the enhanced version of C4.5 decision tree and has been developed to generate a pruned or un-pruned C4.5 DT [6] [21]. The C4.5 DT uses divide-and-conquer approach to growing DTs. The J48 has been developed to address the problems of both C4.5 and Id3 [8] classifiers.

IV. Datasets

For this study we have used the ADL dataset provided by Bruno et al. [22] at the UCI Machine Learning Repository [23]. This dataset includes accelerometer data for Human Primitive detection and it is a collection of labeled accelerometer data recordings that was collected through the tri-axial wrist-worn accelerometer. The data transmission from accelerometer to the PC is done via wired USB cable where the dataset is a collection of 14 ADLs (i.e. brushteeth, climbstairs, combhair) performed by the 16 volunteers (11 Male and 5 female) aged between 19 and 81. For this study, 11 ADLs of the three volunteers (2 male and 1 female) have been chosen from the complete dataset. The chosen ADLs for our study are brushteeth, climbstairs, combhair, drinkglass, getupbed, liedownbed, pourwater, sitdownchair, standupchair, usetelephone and walk. For the simulations we have created three datasets from the complete dataset. The first dataset uses only nine datapoints (x,y,z parameters) to represent each performed activity. This dataset has 8495 total records with 11 attributes. Table I shows example records from this dataset. In the second dataset, each performed activity represented with 15 datapoints. This dataset has 5091 records and each record has 17 attributes. Table II shows few records from dataset2 and Fig.1 shows the count of each activity in dataset2. The third dataset, on the other hand, uses 30 datapoints to represent each performed activity. This dataset has 2538 records and 32 features to represent each record.

TABLE I: Two sample records from ADL dataset1 with three datapoints

x1	y1	z1	x2	y2	z2	x3	y3	z3	gender/id	HMP
22	49	35	22	49	35	22	52	35	f1	brushteeth
29	44	40	35	11	28	41	13	24	m1	combhair

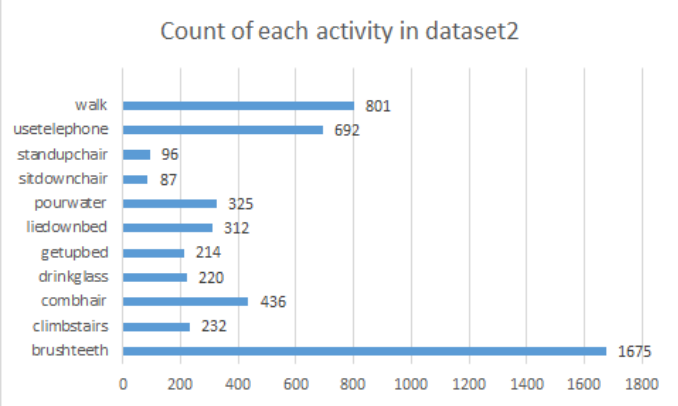

Fig. 1: Count of each ADL activities within the dataset2

V. Evaluation Measures

There are a number of evaluation measures that are commonly used to evaluate the performance of each classification algorithm. Here, we focused on six evaluation measures that are True Positive Rate (TPR), False Positive Rate (FPR), Precision (or Positive Predictive Value (PPV)), Recall, F-measure (F1) and Receiver Operating Characteristics (ROC) curve. These evaluations are mainly done based on four parameters that are True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). Simplest way to explain each of these parameters is through an example.

Given an instance and the classifier, there can be four outcomes [24];

- If an instance is positive and it is predicted to belong to a positive class then this is counted as TP,
- If a positive instance predicted to belong to a negative class then it is counted as FN,
- If an instance is negative and it is predicted to belong to a negative class then this is counted as TN, but,
- If the same instance is predicted to belong to a positive class then this is counted as FP.

Based on these parameters each of the evaluation measures is calculated as follows [24][25];

$$\begin{aligned} TPR(or Recall) &= TP/(TP+TN) \\ FPR &= FP/(FP+TN) \\ Precision &= TP/(TP+FP) \\ F1 &= 2TP/(2TP+FP+FN) \end{aligned} \tag{7}$$

Following this the accuracy is calculated as;

$$Accuracy = (TP+TN)/(P+N) \tag{8}$$

TABLE II: Two sample records from ADL dataset2 with fifteen datapoints

x1	y1	z1	x2	y2	z2	x3	y3	z3	x4	y4	z4	x5	y5	z5	gender/id	HMP
34	48	49	35	48	45	34	51	45	34	48	47	35	49	45	m2	pourwater
27	46	46	26	43	46	26	48	46	26	48	47	28	48	45	m1	drinkglass

TABLE III: Classification accuracy results of the algorithms with nine datapoints

Algorithm	Correctly classified instances (%)
IBL	69.05
Kstar	70.86
KNN	68.65
LWL	52.61
NBTree	64.60
J48	65.96

TABLE IV: Classification accuracy results of the algorithms with fifteen datapoints

Algorithm	Correctly classified instances (%)
IBL	69.75
Kstar	70.53
KNN	69.57
LWL	53.13
NBTree	64.60
J48	64.74

where (P+N) corresponds to the sum of positive and negative instances (the total number of instances). Here, each of the measure can have a value between 0 and 1. For TP, Precision, Recall, F-measure and ROC, the high accuracy means values closer to 1. For FP, as the accuracy gets higher the value of FP gets closer to 0. Precision is one of the performance measures and differs from accuracy as it does not relate to the true value (accepted reference value) [26]. Precision shows the closeness of the independent test results on homogeneous data and usually computed as a standard deviation of the results [26]. The ROC curves are two dimensional graphs where Y axis represents TPR and X axis represents FPR and the graph depicts the relative tradeoffs between the TP (benefits) and FP (costs) [24].

VI. Simulations

For the simulations, datasets that are mentioned in Section IV have been used. The 10-fold cross-validation has been chosen as the test mode and for each simulation the same dataset has been used as a training and test datasets. Here, 'HMP' is chosen as a class attribute for all simulations. Each of the six ML algorithms are trained on the same training set and tested on the same test set (i.e for the first simulation all algorithms trained and tested on dataset1).

Table III shows the accuracy performance of all six algorithms with dataset1. Here it can be seen that the lazy classifiers have mixed accuracy results where some are quite high and some are quite low. DTs, on the other hand, was consistent on performing average classification accuracies. From the results, it can be seen that the best result is achieved by the Kstar with 70.86% accuracy (6020 instances classified correctly). This result is followed by the IBL, which classified 5866 instances correctly and 2629 instances incorrectly. The KNN algorithm performed very similar results with IBL by classifying 68.65% of the instances correctly. DT algorithms followed the results of lazy classifiers by classifying 5604 (J48) and 5372 (NBTree) instances correctly. The lowest accuracy is achieved by the LWL algorithm. Here LWL classified 52.61% of the instances correctly. Hence, from the first simulation results it can be said that the highest and lowest results both achieved by the lazy classifiers.

Second simulations has been carried out with dataset2. Table IV shows the results of these simulations. From the results it can be seen that the DT based algorithms achieved around 64% accuracy where NBTree classified 3289 (64.60%) instances correctly and J48 classified 3296 (64.74%) instances correctly. In general, lazy classifiers performed with higher accuracy results compared to the DT based algorithms. According to the results, the best classification result is archived by the Kstart algorithm with 70.53% of instances classified correctly. This results is followed by the IBL algorithm with %69.75 correctly classified instances. The KNN classifier closely followed the result of IBL by classifying 3542 (69.57%) out of 5091 instances correctly. Within all six algorithms, the lowest classification accuracy is archived by LWL. Here LWL classifier classified 2705 instances correctly and 2386 instances incorrectly, which corresponds to 46.86% of the total instances. We carried out additional number of simulations only with the KNN algorithm to see if changing the number of neighbors (N) affects the classification performance of the KNN algorithm with ADL dataset2. Fig.2 shows the results of these simulations. From the figure it can be seen that the classification accuracy increases slightly between N=2 (69.57%) and N=4 (71.83%). After that the accuracy stays around 71% between N=4 and N=8. For N=9 and N=10 the classification accuracy of the KNN drops again and stays around 70%. In general, results showed that increasing the number of neighbours improved the accuracy of KNN but very little. Hence, it can be said that increasing the number of neighbours did not have a great positive impact, with respect to the accuracy performance, on the KNN. Moreover, by comparing the KNN results with IBL, it can be said that using more than two nearest neighbours for classification achieved slightly (%2.08 when N=4) better results than using only one neighbour (N=1).

Table V shows the time taken for each algorithm to build the classification model for the second simulations. The time taken to build the model is the system time that was used to run the classifier and is converted from millisecond into seconds by WEKA. These results shows that, in general, the lazy classifiers took less time to build the classification model than DT classifiers. Here, the NBTree algorithm took the longest time, 20.95, compared to all other five algorithms. The NBTree algorithm is a hybrid algorithm which carries both DT and NB algorithms' working principles, and therefore it was expected from it to require more time. The lowest time requirement (0

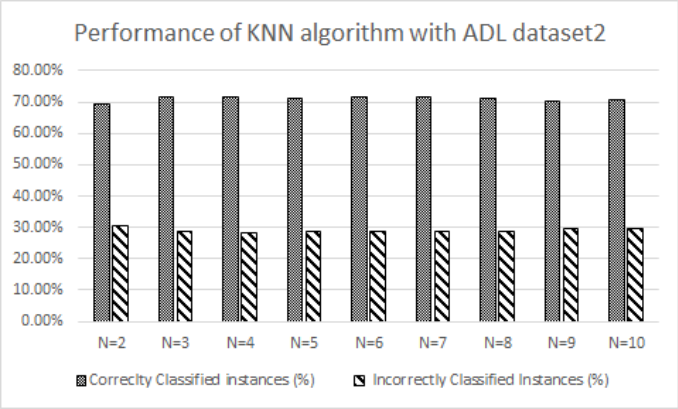

Fig. 2: Performance of KNN algorithm with ADL dataset2

TABLE V: Algorithms vs. time taken to build the model with fifteen datapoints

Algorithm	Time
IBL	0.02 sec
Kstar	0 sec
KNN	0 sec
LWL	0 sec
NBTree	20.95 sec
J48	0.52 sec

TABLE VI: Detailed accuracy measures of the algorithms' performance with fifteen data points

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
IBL	0.698	0.043	0.696	0.698	0.695	0.827
Kstar	0.705	0.044	0.7	0.705	0.7	0.948
KNN	0.696	0.046	0.709	0.696	0.693	0.872
LWL	0.531	0.17	0.342	0.531	0.411	0.859
NBTree	0.646	0.062	0.641	0.646	0.629	0.898
J48	0.647	0.054	0.641	0.647	0.644	0.823

sec), on the other hand, is archived by the KNN (N=2), Kstar and LWL. Within lazy classifiers, only IBL needed little bit more time than 0 sec. The J48 needed 0.52 sec to build the classification model, which is quite less than what NBTree required.

Table VI shows the detailed evaluation measure results of all six algorithms with dataset2. From this table it can be seen that accuracy results of all six algorithms are reflected and justified with the evaluation measures too. Kstart, which has the highest classification accuracy results with the second dataset, has the TP, Precision, Recall and F-measure values around 0.7 and ROC area around 0.9, all of which are the closest to 1 among all other algorithms. Moreover, LWL, which performed the lowest classification accuracy, has the values for the same evaluation measures between 0.3 and 0.5 that are the ones closest to 0 among all other algorithms.

Final simulations have been carried out with the third dataset that has 30 datapoins that corresponds to 10 sets of x,y and z compositions. As previously mentioned, this dataset have 32 features to define each instance where 30 out of 32 features are datapoints. Table VII shows the simulation results

TABLE VII: Classification accuracy results of the algorithms with thirty datapoints

Algorithm	Correctly classified instances (%)
IBL	69.70
Kstar	68.71
KNN	68.71
LWL	54.60
NBTree	64.38
J48	61.74

TABLE VIII: Detailed accuracy measures of the algorithms' performance with thirty data points

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
IBL	0.697	0.042	0.7	0.697	0.696	0.827
Kstar	0.687	0.043	0.694	0.687	0.687	0.946
KNN	0.695	0.046	0.708	0.695	0.691	0.871
LWL	0.546	0.143	0.367	0.546	0.437	0.861
NBTree	0.644	0.06	0.634	0.644	0.624	0.882
J48	0.617	0.06	0.613	0.617	0.615	0.802

for all six algorithms. From the table it can be seen that the best results, among all six algorithms, is achieved by the IBL algorithm with 69.70% classification accuracy. This result very closely followed by the KNN algorithm with 69.50% accuracy which corresponds to 1769 correctly classified instances and 774 incorrectly classified instances. KStar, which achieved the best classification results in first (9 datapoints) and second (15 datapoints) simulations, classified 68.71% (1744) instances correctly with third dataset. Approximately with 4% less instances, NBTree classified 64.38% instances correctly. Similar to the first two simulations, LWL performed the lowest classification accuracy by classifying only 54.60% (1386) of the total instances correctly. From the results it can be seen that, different from the previous simulations with dataset1 and dataset 2, the best classification accuracy results are achieved by the IBL algorithm rather than Kstart and this has been reflected on the evaluation measure results that are presented on Table VIII. Here, IBL has the values for TP, Precision, Recall, F-measure and ROC area closest to 1 compared to all other algorithms. Moreover, the FP value for the IBL is the closest to 1 when compared with all other five algorithms.

VII. Conclusion and Future Works

Europe is aging and understanding the elderly people plays very critical role in finding the best solutions for demographic challenges. The ADL information of the elderly can give insight into elderly people's lives with respect to what they do during the day and what kind of problems they encounter, such as when they start showing symptoms for the new medical conditions. This information then could be used to formulate the best personalized solutions which will make it possible for them to live independent, health and active life. As the popularity of wearable devices is in rapid increase, such devices could be used to gather ADL information from the elderly people. However, analyzing and understanding such information is as important as the collection of the information. In this paper we tested, analyzed and compared the performance of well-known lazy and decision tree classification algorithms with the ADL data. The simulation results with three different sized ADL

datasets showed that, in general, lazy classifiers performed better than DT algorithms with respect to understanding and classifying the ADL data more accurately with very small time requirements. The results also showed that the ML algorithms could be used for such purposes. On the other hand, the maximum classification accuracy result, reflected through several evaluation measures, was around only 71%, which is still not good enough considering we are dealing with very sensitive data. Hence, as a future work we would focus on investigating different ways of improving the accuracy performance of the algorithms which resulted the best performances in this study.

REFERENCES

[1] The 2015 Aging Report, Underlying assumtions and projection Methodologies, European Commission (DG ECFIN) and Economic Policu Commttee (AWG), ISSN 0379-0991 (print), ISSN 1725-3217 (online), 2014, Available [online]: http://tinyurl.com/o7dnhhs

[2] A. Fleury, M. Vacher and Norbert Noury. SVM-based multimodal classification of activities of daily living in health smart homes:sensors, algorithms and first experimental results, 14(2), pp. 274-283, March 2010.

[3] B. C. Cheng, Y. A. Tsai, G. T. Liao and E. S. Byeon. HMM machine learning and inference for Activities of Daily Living recognition,The Journal of Supercomputing, Springer, 54(1), pp. 29-42, October 2010.

[4] E. M. Tapia, S. S. Intille and K. Larson. Activity recognition in the home using simple and ubiquitous sensors, Proceedings of Pervasive, vol. 3001, Springer-Verlag, Berlin/Heidelberg, pp. 158-175, April 2004.

[5] D. W. Aha, D. Kibler and M. K. Albert. Instance-based learning algorithms, Machine Learning Journal, 1(6), pp. 37-66, January 1991.

[6] I. H. Witten, E. Frank and M. A. Hall. Data mining practical machine learning tools and techniques 3rd Edition, Morgan Kaufmann, USA, pp. 472-550, 2011.

[7] A. Cufoglu, M. Lohi and C. Everiss. Weighted Instance Based Learner (WIBL) for user profiling, IEEE Symposium on Applied Machine Intelligence and Informatics, pp. 201-205, January 2012.

[8] I. H. Witten and E. Frank. Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[9] O. Gomez, E F. Morales and J. A. Gonzales. Weighted instance-based learning using representative intervals, Mexican International Conference on Advances in Artificial Intelligence, vol. 4827,pp. 420-430, November 2007.

[10] J. G. Clear and L. E. Trigg. K*: An instance-based learner using an entropic distance measure, International Conference on Machine Learning, pp. 108-114, July 1995.

[11] C.G. Atkeson, A.W. Moore, and S. Schaal. Locally weighted learning. Artificial Intelligence Review, 11(1):11-73, 1997.

[12] P. Englert. Locally Weighted Learning, Seminar Class on Autonomous Systems, 2012.

[13] V. P. Bresferean. Analysis and predictions on students behaviour using decision trees in WEKA environment, IEEE International Conference on Information Technology Interfaces, pp. 51-56, June 2007.

[14] A.L. Symeanidis and P.A. Mitkas. Agent intelligence through data mining multiagent systems, artificial societies and simulated organizations; 14, Electron. Book, New York: Springer Science and Business Media, pp. 21-23, 27-28, 2005.

[15] D. R. Amancio, C. H. Comin, D. Casanova, G. Travieso, O. M. Bruno, F. A. Rodriguesand and L. D. F. Costa. A Systematic Comparison of Supervised Classifiers. PLoS ONE 9(4), April 2014.

[16] Z. Shi, Y. Huang and S. Zhang. Fisher score based naive Bayesian classifier, IEEE International Conference on Neural Networks and Brain, pp. 1616-1621, October 2005.

[17] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms, IEEE Transactions on Knowledge and Data Engineering, 17(3), pp. 299-310, March 2005.

[18] Z. Xie and Q. Zhang. A study of selective neighbourhood-based naive Bayes for efficient lazy learning, IEEE International Conference on Tools with Artificial Intelligence, pp. 758-760, November 2004.

[19] G. Santafe, J.A. Loranzo and P. Larranaga. Bayesian model averaging of naive bayes for clustering, IEEE Transactions on Systems, Man, and Cybernetics, 36(5), pp. 1149 -1161, October 2006.

[20] F.V. Jensen. Introduction to Bayesian networks, Springer, August 2000.

[21] J. R. Quinlan. C4.5: Programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[22] B. Bruno, F. Mastrogiovanni and A. Sgorbissa. ADL recognition with wrist-worn accelerometer dataset, February 2014. Available [online]:http://archive.ics.uci.edu/

[23] M. Lichman. UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science, 2013. Available[online]:http://archive.ics.uci.edu/ml

[24] T. Fawcett. An introduction to ROC analysis, Pattern Recognition Letters, Elsevier, 27(8), pp. 861-874, June 2006.

[25] D. M. W. Powers. Evalution:From precision,recall and F-factor to ROC, informedness, markedness & Correlation, Journal of Machine Learning Technologies, 2(1), pp. 37-63, December 2011.

[26] M. Ester, H.P. Kriegel, J. Sander and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise, International Conference on Knowledge Discovery and Data Mining, pp. 226-231, August 1996.