Machine Learning Prediction of Human Activity Recognition

Getinet Yilam and Dileep Kumar

*School of Electrical Engineering and Computing: Computer Science & Engineering program. *corresponding author E-mail: getyilma@astu.edu.et*

Abstract

Wearable computation is getting integrated into our daily life. It has got wide acceptance due to their small sizes, and reasonable computation power. These wearable devices loaded with sensors are good candidates to monitor user's daily behavior (walking, jogging, sleeping…). Human Activity Recognition (HAR) has the potential to benefit the development of assistive technologies in order to support care of the chronically ill and people with special needs. Activity recognition can be used to provide information about patients' routines to support the development of e-health systems, like Ambient Assisted Living (AAL). Despite human activity recognition being an active field for more than a decade; the development of context-aware systems, there are still key aspects that, if addressed, would constitute a significant turn in the way people interact with mobile devices. The study discusses the principal issues and challenges of HAR systems. A general and data acquisition architecture for HAR systems are presented. HAR systems made use of machine learning techniques and tools, which are helpful to build patterns to describe, analyze, and predict data. Since a human activity recognition system should return a label such as walking, sitting, running, etc., most HAR systems work in a supervised fashion. The objective of proposed study is applying multiple machine learning algorithms on the HAR dataset from Groupware. Out of the 5 machine learning algorithms that random forest yields the highest accuracy in predicting activities correctly, results showed the accuracy of 100%. All the models were also ensembled to improve overall accuracy.

Keywords: *Human Activity Recognition, Wearable Computing, Machine Learning, IoT, R.*

1. INTRODUCTION

Internet of Things (IoT) is "a network of interconnected things/ devices which are embedded with sensors, software, network connectivity and necessary electronics that enables them to collect and exchange data making them responsive." – Wiki. IoT is "a network of items - each embedded with sensors - which are connected to the Internet." - IEEE Definition.

ITU has pointed out 4 dimensions of IoT: tagging things identification of objects Using RFID, Bar Code, GPS, Accelerometer etc., feeling things through sensors, near field communication (NFC) and wireless sensor networks, thinking things using embedded systems and special instructions, shrinking things using nanotechnology.

According to IDC, within 2020, the number of things connected to the internet will be about 50 Billion and the world's data will amount to 44 zettabytes by 2020, 10% of it from the internet of things which makes the amount of data generated from IoT tremendous [1].

Recently, wearable devices such as Smart watches, Google glasses, Fitness trackers, Sports watches, Smart clothing, Smart jewelry, Implantable etc. have got a lot of interests and wide acceptance due to their small sizes, reasonable computation power, and practical power capabilities. These wearable devices loaded with sensors (e.g. accelerometer, gyroscope) provide a good candidate to monitor user's daily behavior (e.g. walking, jogging, and smoking). Recent advancement of wearable technology has resulted in utilization of wearable and non-intrusive systems for health and activity monitoring. Such continuous monitoring of life and daily activities, motivate the users to maintain healthy living style. Wearable device can comprise 4 tri-axial ADXL335 accelerometers connected to an ATmega328V microcontroller [2].

The accelerometers can be positioned in the waist [1], left thigh [2], right ankle [3], and right arm [4]. All accelerometers have to be calibrated prior to the data collection. The calibration consists of positioning the sensors and the performance of the reading of values to be considered as "zero"'. From the calibration, the read values of each axis during data collection are subtracted from the values obtained at the time of the calibration [3-4].

Machine learning (ML), algorithms, tools and techniques are helpful to build patterns to describe, analyze, and predict data. In a machine learning context, patterns are to be discovered from a set of given observations denominated instances. Such input set is training set [5]

The paper is organized as the following: Section II presents related work, Section III discusses HAR and its techniques, Section IV presents experimentation in R and Results obtained.

2. RELATED WORK

Up to now, there have been many studies related to human activity recognition. Machine Learning based methods that have been previously employed for recognition include Naive Bayes, SVMs, Threshold-based and Markov chains [5]. Although it has been not fully clear which method performs better for AR, SVMs have confirmed successful application in several areas including heterogeneous types of recognition such as handwritten characters [6] and speech [7].

In ML, fixed-point arithmetic models have been previously studied [8-9] initially because devices with floating-point units were unavailable or expensive. The possibility of retaking these approaches for AmI systems that require either low cost devices or to allow load reduction in multitasking mobile devices has nowadays become particularly appealing. Anguita et al. in [10] introduced the concept of a Hardware-Friendly SVM (HF-SVM). This method exploits fixed-point arithmetic in the feed-forward phase of the SVM classier, so as to allow the use of this algorithm in hardware-limited devices.

3. Human Activity Recognition

The recognition of human activities has become a task of high interest, especially for medical, military, and security applications. For instance, patients with diabetes, obesity, or heart disease are often required to follow a well-defined exercise routine as part of their treatment [11]. HAR has the potential to benefit the development of assistive technologies in order to support care of the elderly, the chronically ill, monitoring energy expenditure and for supporting weight-loss, programs digital assistants for weight lifting exercises, and people with special needs. Example using smart homes to detect and analyze health events is given below.

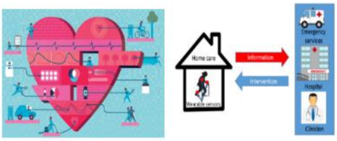


Figure 1: Smart Home based Health Data Analysis [2]

The home supportive environment delivers trend data and detection of incidents using non-intrusive wearable sensors. This facilitates a quick measurement and fast acceptance at the same time. Through real-time processing and data transmission, healthcare suppliers will be able to monitor the subject’s motions during daily activities and also to detect unpredictable events that may occur, like a fall. The subject’s records can be used in medical decision support, in prediction and prevention of accidents [12-14].

The two approaches commonly used for HAR are (1) image processing with computer vision and (2) use of wearable sensors. The image processing approach does not require the use of equipment in the user’s body, but imposes some limitations such as restricting operation to the indoor environments, requiring camera installation in all the rooms, lighting and image quality concerns and user privacy.

But, the use of wearable sensors minimizes these problems even though they require users to wear the equipment through extended periods of time. Hence, the use of wearable sensors may lead to inconveniences with battery charges, positioning, and calibration of sensors.

Table 1: Some of the activities recognized by HAR systems are given as follow

Group of Activities	Activities
Ambulation	Walking, running, sitting, standing still, lying, climbing stairs, descending stairs, riding escalator, and riding elevator.
Transportation	Riding a bus, cycling, and driving
Phone Usage	SMSing, Making a call.
Daily Activities	Eating, drinking, working at PC, reading, watching TV, brushing teeth, stretching, scrubbing and vacuuming.
Exercise/Fitness	Rowing, lifting weights, spinning, Nordic walking, and doing pushups.
Military	Crawling, kneeling, situation assessment, and opening a door
Upper body	Chewing, speaking, swallowing, sighing and moving the head.
Others	Heartbeat, respiration, temperature, location, contraction, and etc.

4. APPLYING MACHINE LEARNING

Similar to other machine learning applications, activity recognition requires two stages, i.e., training and testing (also called evaluation). The training stage initially requires a time series dataset of measured attributes from individuals performing each activity. The time series are split into time windows to apply feature extraction thereby filtering relevant information in the raw signals. Later, learning methods are used to generate an activity recognition model from the dataset of extracted features. Likewise, for testing, data are collected during a time window, which is used to extract features. Such feature set is evaluated in the priory trained learning model, generating a predicted activity label.

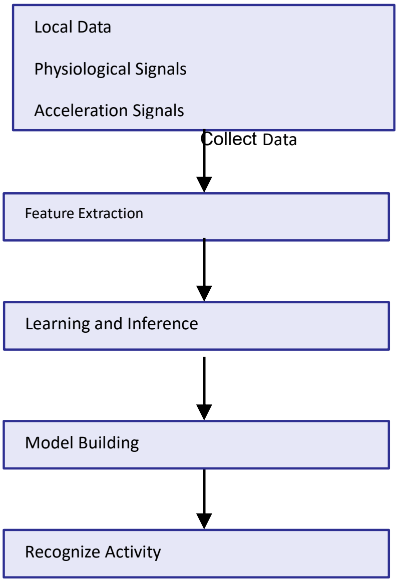


Figure 2: Machine Learning Approach based on wearable sensors

Generic Data Acquisition Architecture:

In the first place, wearable sensors are attached to the person's body to measure attributes of interest such as motion, location, temperature, and ECG, among others. These sensors should communicate with an integration device (ID), which can be a cellphone, a PDA, a laptop, or a customized embedded system. The main purpose of the ID is to preprocess the data received from the sensors and, otherwise send raw signal to an application server for real time monitoring, visualization, and/or analysis. The communication protocol might be UDP/IP or TCP/IP, according to the desired level of reliability.

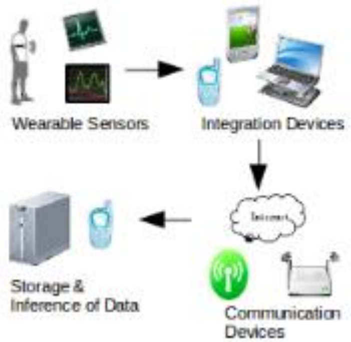


Figure 3: General Data Collection Process for HAR

HAR systems make use of machine learning (ML) tools, which are helpful to build patterns to describe, analyze, and predict data [15]. It is used to classify the mistakes in activity recognition. In a machine learning context, patterns are to be discovered from a set of given examples or observations denominated instances. Such input set is called training set. Each instance is a feature vector extracted from signals within a time window. The examples in the training set may or may not be labeled, i.e., associated to a known class (e.g., walking, running, sleeping etc.). In some cases, labeling data is not feasible because it may require an expert to manually examine the examples and assign a label based upon their experience. This process is usually tedious, expensive, and time consuming in many data mining applications. Since a human activity recognition system should return a label such as walking, sitting, running, etc., most HAR systems work in a supervised fashion. Indeed, it might be very hard to discriminate activities in a completely unsupervised context. Some systems work in a semi supervised fashion allowing part of the data to be unlabeled.

5. IMPLEMENTATION AND RESULTS

In general, the selection of the classification algorithm for HAR has been merely supported by empirical evidence. The vast majority of the studies use cross validation with statistical tests to compare classifier’s performance for a particular dataset. The classification results for a particular method can be organized in a confusion matrix M_{nxn} for a classification problem with n classes. This is a matrix such that the element M_{ij} is the numbers of instances from class i that was actually classified as class j. The following values can be obtained from the confusion matrix in a binary classification problem:

- True Positives (TP): The number of Class A activities that were classified as Class A.
- True Negatives (TN): The number of Non Class A

activities that were classified as Non Class A.

- False Positives (FP): The number of Non Class A activities that were classified as Class A.
- False Negatives (FN): The number of Class A activities that were classified as Non Class A.

Table 2 Classification algorithms for HAR.

Type	Classifier
Decision Tree	CD4.5, ID3
Bayesian	Naïve Bayes and Bayesian Networks
Instance Based	K-Nearest Neighbors
Neural Network	Multi-layer Perceptron
Domain Transform	Support Vector Machines
Markov Models	MLR, ALR
Classifier Ensembles	Boosting and Bagging

The accuracy is the most standard metric to summarize the overall classification performance for all classes and it is defined as follows:

$$\text{Accuracy} = \frac{TP + TN}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \dots\dots 1$$

We used R for our experimentation. R is a free, open source language with highly active community members available across all platforms (Linux, Mac, and Windows). Due to its underlying philosophy and design; R is useful for statistical computation and graphic visualization [16].

The goal of this study is to build a model that can predict the type of activity or exercise listed in table 1 above performed based on measurements of human movement. We used machine learning techniques to build a model to predict the manner of the exercise, "classes", based on a variety of collected information. Machine learning algorithms are applied on the Human Activity Recognition dataset from

Groupware. Out of five ML algorithms, which are bagging with classification trees, logistic regression, support vector machines, random forest, gradient boosting model and classification trees, random forest yields the highest accuracy rate of 100%. The entire models except classification tree are ensemble to give a better prediction. The final outcome has shown 98% accuracy rate on the 20 testing data point.

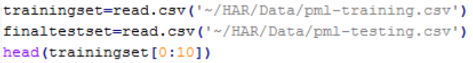
Dataset:

The data used in this analysis is the Human Activity Recognition Dataset (weight lifting exercise), provided by Groupware [17]. The dataset consists of 19,622 observations of 160 variables that describe subjects and their physical movement during activities. The approach for the Weight Lifting Exercises dataset is to investigate "how (well)" an activity was performed by the wearer. The "how (well)" investigation has only received little attention so far, even though it potentially provides useful information for a large variety of applications, such as sports training. Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions [17]:

- exactly according to the specification (Class A),
- throwing the elbows to the front (Class B),
- lifting the dumbbell only halfway (Class C),
- lowering the dumbbell only halfway (Class D) and
- Throwing the hips to the front (Class E) [17].

Data Loading:

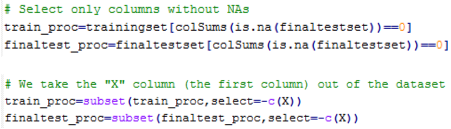
Firstly, we loaded the data into memory using the following



```
trainingset=read.csv('~/HAR/Data/pml-training.csv')
finaltestset=read.csv('~/HAR/Data/pml-testing.csv')
head(trainingset[0:10])
```

Data Preprocessing:

Many of the entries in the observation contain NAs (Not applicable) values. We excluded columns that have NAs from the table, as these columns will not add any useful information to the model that we build. This reduces the number of columns from 160 to 60.

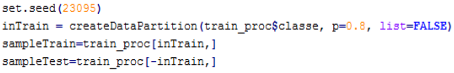


```
# Select only columns without NAs
train_proc=trainingset[colSums(is.na(finaltestset))==0]
finaltest_proc=finaltestset[colSums(is.na(finaltestset))==0]

# We take the "X" column (the first column) out of the dataset
train_proc=subset(train_proc,select=-c(X))
finaltest_proc=subset(finaltest_proc,select=-c(X))
```

Data Analysis:

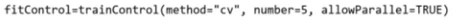
Now, started to analyze our data,

Step 1: Split the data into training testing data. Notice that the ratio of training data to testing data is 0.8 reduce variance and increase performance.



```
set.seed(23095)
inTrain = createDataPartition(train_proc$classe, p=0.8, list=FALSE)
sampleTrain=train_proc[inTrain,]
sampleTest=train_proc[-inTrain,]
```

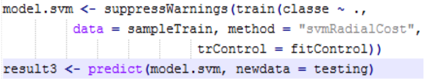
Before applying machine learning algorithms to train our model, first the cross-validation parameters were tuned. Out-of-sample error was low because 5-fold Cross Validation takes its effect and avoid over fitting.



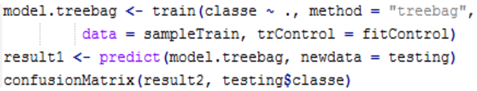
```
fitControl=trainControl(method="cv", number=5, allowParallel=TRUE)
```

Step 2: Train models with the training data using 6 chosen machine learning algorithms.

Bagging with trees:

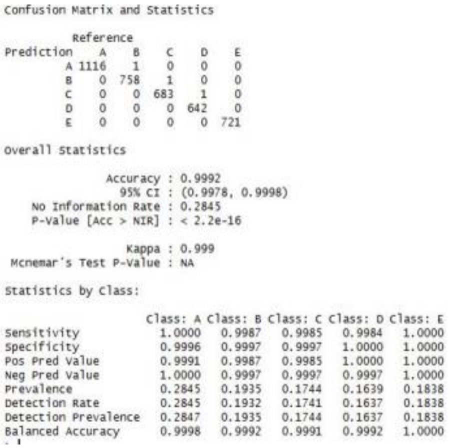


```
model.svm <- suppressWarnings(train(classe ~ .,
            data = sampleTrain, method = "svmRadialCost",
                        trControl = fitControl))
result3 <- predict(model.svm, newdata = testing)
```



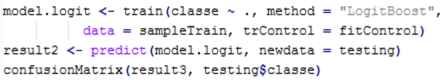
```
model.treebag <- train(classe ~ ., method = "treebag",
            data = sampleTrain, trControl = fitControl)
result1 <- predict(model.treebag, newdata = testing)
confusionMatrix(result2, testing$classe)
```

This gave a very high accuracy rate, 99.92%

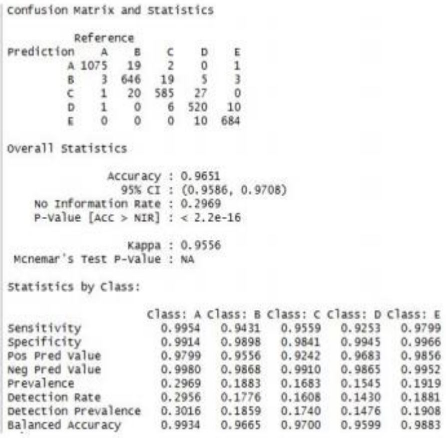


```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1116    1    0    0    0
         B    0  758    1    0    0
         C    0    0  683    1    0
         D    0    0    0  642    0
         E    0    0    0    0  721

Overall Statistics

               Accuracy : 0.9992
                 95% CI : (0.9978, 0.9998)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.999
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9987   0.9985   0.9984   1.0000
Specificity            0.9996   0.9997   0.9997   1.0000   1.0000
Pos Pred Value         0.9991   0.9987   0.9985   1.0000   1.0000
Neg Pred Value         1.0000   0.9997   0.9997   0.9997   1.0000
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2845   0.1932   0.1741   0.1637   0.1838
Detection Prevalence   0.2847   0.1935   0.1744   0.1637   0.1838
Balanced Accuracy      0.9998   0.9992   0.9991   0.9992   1.0000
```

Logistic Regression with boosting:



```
model.logit <- train(classe ~ ., method = "LogitBoost",
            data = sampleTrain, trControl = fitControl)
result2 <- predict(model.logit, newdata = testing)
confusionMatrix(result3, testing$classe)
```



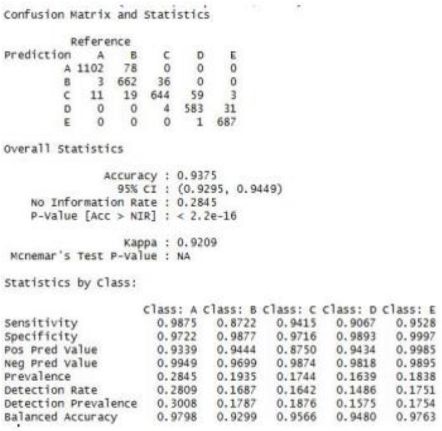
```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1075   19    2    0    1
         B    3  646   19    5    3
         C    1   20  585   27    0
         D    1    0    6  520   10
         E    0    0    0   10  684

Overall Statistics

               Accuracy : 0.9651
                 95% CI : (0.9586, 0.9708)
    No Information Rate : 0.2969
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9556
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9954   0.9431   0.9559   0.9253   0.9799
Specificity            0.9914   0.9898   0.9841   0.9945   0.9966
Pos Pred Value         0.9799   0.9556   0.9242   0.9683   0.9856
Neg Pred Value         0.9980   0.9868   0.9910   0.9865   0.9952
Prevalence             0.2969   0.1883   0.1683   0.1545   0.1919
Detection Rate         0.2956   0.1776   0.1608   0.1430   0.1881
Detection Prevalence   0.3016   0.1859   0.1740   0.1476   0.1908
Balanced Accuracy      0.9934   0.9665   0.9700   0.9599   0.9883
```

This model gave 96.51% accuracy rate, an efficient model as well.

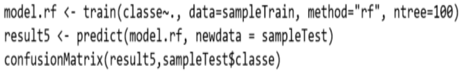
Support Vector Machines with boosting:

This model gave 93.75% accuracy rate.

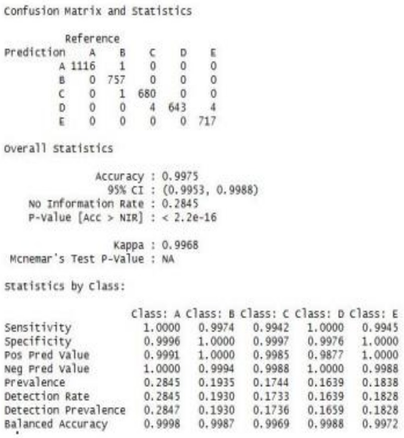
Random forest:



```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1102   78    0    0    0
         B    3  662   36    0    0
         C   11   19  644   59    3
         D    0    0    4  583   31
         E    0    0    0    1  687

Overall Statistics

               Accuracy : 0.9375
                 95% CI : (0.9295, 0.9449)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9209
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9875   0.8722   0.9415   0.9067   0.9528
Specificity            0.9722   0.9877   0.9716   0.9893   0.9997
Pos Pred Value         0.9339   0.9444   0.8750   0.9434   0.9985
Neg Pred Value         0.9949   0.9699   0.9874   0.9818   0.9895
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2809   0.1687   0.1642   0.1486   0.1751
Detection Prevalence   0.3008   0.1787   0.1876   0.1575   0.1754
Balanced Accuracy      0.9798   0.9299   0.9566   0.9480   0.9763
```



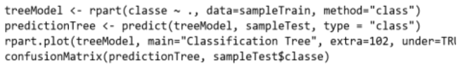
```
model.rf <- train(classe~., data=sampleTrain, method="rf", ntree=100)
result5 <- predict(model.rf, newdata = sampleTest)
confusionMatrix(result5,sampleTest$classe)
```



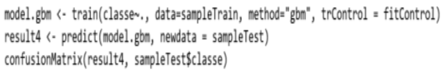
```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1116    1    0    0    0
         B    0  757    0    0    0
         C    0    1  680    0    0
         D    0    0    4  643    4
         E    0    0    0    0  717

Overall Statistics

               Accuracy : 0.9975
                 95% CI : (0.9953, 0.9988)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9968
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9974   0.9942   1.0000   0.9945
Specificity            0.9996   1.0000   0.9997   0.9976   1.0000
Pos Pred Value         0.9991   1.0000   0.9985   0.9877   1.0000
Neg Pred Value         1.0000   0.9994   0.9988   1.0000   0.9988
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2845   0.1930   0.1733   0.1639   0.1828
Detection Prevalence   0.2847   0.1930   0.1736   0.1659   0.1828
Balanced Accuracy      0.9998   0.9987   0.9969   0.9988   0.9972
```

This model gave 100% accuracy rate.

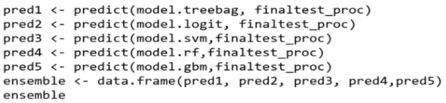
Generalized Boosting Regression Model (GBM):



```
treeModel <- rpart(classe ~ ., data=sampleTrain, method="class")
predictionTree <- predict(treeModel, sampleTest, type = "class")
rpart.plot(treeModel, main="Classification Tree", extra=102, under=TRU
confusionMatrix(predictionTree, sampleTest$classe)
```



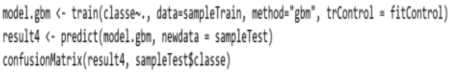
```
model.gbm <- train(classe~., data=sampleTrain, method="gbm", trControl = fitControl)
result4 <- predict(model.gbm, newdata = sampleTest)
confusionMatrix(result4, sampleTest$classe)
```



```
pred1 <- predict(model.treebag, finaltest_proc)
pred2 <- predict(model.logit, finaltest_proc)
pred3 <- predict(model.svm,finaltest_proc)
pred4 <- predict(model.rf,finaltest_proc)
pred5 <- predict(model.gbm,finaltest_proc)
ensemble <- data.frame(pred1, pred2, pred3, pred4,pred5)
ensemble
```

This model gave 99.75% accuracy rate.

Classification tree algorithm:



```
model.gbm <- train(classe~., data=sampleTrain, method="gbm", trControl = fitControl)
result4 <- predict(model.gbm, newdata = sampleTest)
confusionMatrix(result4, sampleTest$classe)
```

This model had given 85.5% accuracy rate. All models are evaluated, now we turned to ensemble them.

Step 3: Ensemble learning algorithms & predict

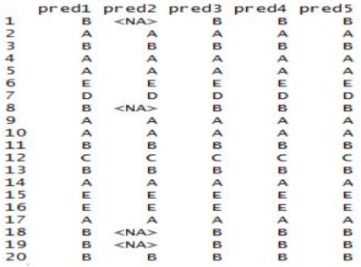
Table 3: prediction variable and respective algorithms description

variable	Algorithm
Pred1	Bagging with tree
Pred2	Logistic regression with boosting
Pred3	Support vector machine
Pred4	Random forest
Pred5	Generalized Boosting Regression Model

Ensemble result [B, A, B, A, A, E, D, B, C, B, A, E, E, AB, B, B] with ensemble accuracy 98% and 0.02 error. Here we can see the final outcome; note pred4 has much higher accuracy than any other predictors. Table 4 gives the rules to pick out that final answer:

If all 5 models give the same answer, that’s definitely the correct answer. If not, compare the results of others, and pick the one that gives higher class value. If all 5 give different answers, follow the one pred4 i.e. random forest gives which is 100% accurate. So here we get the final answers, which reportedly 98% accuracy rate on the final outcome.

Table 4: ensemble result



```
   pred1 pred2 pred3 pred4 pred5
1      B  <NA>     B     B     B
2      A     A     A     A     A
3      B     B     B     B     B
4      A     A     A     A     A
5      A     A     A     A     A
6      E     E     E     E     E
7      D     D     D     D     D
8      B  <NA>     B     B     B
9      A     A     A     A     A
10     A     A     A     A     A
11     B     B     B     B     B
12     C     C     C     C     C
13     B     B     B     B     B
14     A     A     A     A     A
15     E     E     E     E     E
16     E     E     E     E     E
17     A     A     A     A     A
18     B  <NA>     B     B     B
19     B  <NA>     B     B     B
20     B     B     B     B     B
```

6. CONCLUSIONS AND FUTURE WORK

The paper studied human activity recognition techniques and presented the general data collection process for HAR and also the machine learning based data analysis process using R Out of the 5 ML algorithms that are applied, the random forest method yielded 100% accuracy over the other methods. A next step in this path would be to store real-time sensor data and analyze.to provide real-time recommendation to users.

REFERENCES:

[1]http://www.computerweekly.com/news/2240217788/Data-set-to-grow-10-fold-by-2020-as-internet-of-things-takes-off

[2] Adam Thierer. “The Internet of Things and Wearable Technology: Addressing Privacy and Security Concerns without Derailing Innovation.” Mercatus Working Paper, Mercatus Center at George Mason University, Arlington, VA, November 2015.

[3] J. Perez, M. A. Labrador, and S. J. Barbeau, "G-sense: A scalable architecture for global sensing and monitoring IEEE Network", vol. 24, no. 4, pp. 57-64, 2010.

[4] J. Yin, Q. Yang, and J. Pan, “Sensor-based abnormal human-activity detection”, IEEE Transactions on Knowledge and Data Engineering vol. 20, no. 8, pp. 1082-1090, 2008.

[5] Mannini, A., Sabatini, A.M.: Machine learning methods for classifying human physical activity from on-body

accelerometers. Sensors 10(2) (2010) 1154{1175

[6]. Ravi, N., D, N., Mysore, P., Littman, M. L.: Activity recognition from accelerometer data. In: In Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI, AAAI Press (2005), pages 1541-1546.

[7]. Kwapisz, J.R., Weiss, G.M., Moore, S. A.: Activity recognition using cell phone accelerometers. SIGKDD Explor. Newsl. 12(2) (March 2011), pages 74-82.

[8]. LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., Drucker, H., Guyon, I., Mller, U., Sckinger, E., Simard, P., Vapnik, V.: Comparison of learning algorithms for handwritten digit recognition. In: International Conference on Articial Neural Networks. (1995) pages 53-60.

[9]. Ganapathiraju, A., Hamaker, J., Picone, J., "Applications of support vector machines to speech recognition". Signal Processing, IEEE Transactions on 52(8) (August 2004) 2348 - 2355.

[10]. Anguita, D., Ghio, A., Pischiutta, S., Ridella, S.: A hardware-friendly support vector machine for embedded automotive applications. In: Neural Network

[11] T. van Kasteren, G. Englebienne, and B. Krse, "An activity monitoring system for elderly care using generative and discriminative models", Journal on Personal and Ubiquitous Computing, 2010.

[12] D. Choujaa and N. Dulay, "Tracme: Temporal activity recognition using mobile phone data", in IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, vol. 1, pp. 119-126, 2008.

[13] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen, "Activity classification using realistic data from wearable sensors", IEEE Transactions on Information Technology in Biomedicine, vol. 10, no. 1, pp. 119-128, 2006.

[14] O. D. Lara and M. A. Labrador, "A mobile platform for real time human activity recognition", in IEEE Conference on Consumer Communications and Networks, 2012.

[15] M. Stikic, D. Larlus, S. Ebert, and B. Schiele, "Weakly supervised recognition of daily life activities with wearable sensors", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 12, pp. 2521-2537, 2011.

[16] https://www.r-project.org/

[17] http://groupware.les.inf.puc-rio.br/har