# Efficient Activity Recognition for Individual Fitness Tracking

Kumar Avusherla & Sagar Sharma

**Department of Computer Science and Engineering**

**Wright State University**

December 10, 2015

## I. Abstract:

With emergence of electronic wearable devices and smartphones, activity recognition has become an important research area in computation world. The wearable devices continuously track an individual's vitals such as movements, heartbeats, and body temperature, hence opening up opportunities to analyze the data to come with inferences about a person's health habits and achieving a healthy goal. Efficient activity recognition explicitly translates to important information such as total calories burned, steps taken, miles walked, and active hours. In addition, some of the analytics also reveal information about potential health problems and other anomaly detections. Inspired from legacy medical tracking systems that were exclusively used by doctors to constantly monitor their patients, the wearable devices provide the advantage of incorporating fitness and healthy habits to an every person's daily lives. This project focuses on the machine learning techniques that are prevalent in activity recognition. We will elaborate on inner working of such algorithms starting from the very first steps of data collection and feature engineering. We will compare two well performing prevalent classification techniques- Random Forest and K Nearest Neighbors, and measure their performance and complexity. The goal of this project is to reassess the benchmark on the popular activity recognition problem and reanalyze the results published by other researchers. In doing so we should also be able to see the gaps where improvements can happen in the area,

## II. Introduction:

With their busy work habits, more and more people are having a hard time managing their time for regular workouts to maintain healthy living habits. Advancement in computation power of small devices gives us the wonderful present of wearable devices, which we can carry with us and make a constant supervision of our health habits and activities. These devices motivate us and even alarm us to progress towards our goals. With features like socially interacting and competing amongst friends are unique features of wearable technologies. For common users it is in fact amusing that the small wearable devices can precisely track information like calories used, steps taken, miles walked, stairs climbed, heartbeats, and body-temperature. However, it involves several machine learning algorithms and data processing techniques that come in use to program the chip

in wearable devices so as to identify a user's activities accurately. The problem of activity recognition is "recognizing the actions of one or more entities using a series of observations on entities' actions and environmental conditions" (W. Liu).

The data collected by wearable devices are raw in format and do not mean anything without proper processing and analysis. Usually it is a time series data that records change in velocity or acceleration of the person's appendages or body every few moments of time. A very common means of data collection is using the accelerometers to keep log of accelerations in *x*, *y*, and *z* axes. Another common means of data collection in some GPS enabled devices depends on GPS location information and relative movements. The former alternative is a cheaper method as its power consumption is significantly lesser than a GPS device. Also, it is not affected by problem in GPS signal reception (M. Liu). In this paper we will be focusing on wearable devices that are based on accelerometer sensors.

The raw signal data that is gathered from the accelerometers do not provide any insight on person's physical activity or goals as such. The next step is to make something out of the raw data. In order to run machine learning algorithms, one needs to extract important features from the data collected that actually mean something. Feature engineering is an important aspect of machine learning and might determine success or failure of a method. Some of the features that are extracted from the raw data produced by the wearable devices are mean acceleration, median, standard deviation, energy, signal magnitude area, and coefficients (Ravi). We will describe these features in detail in a later section. We will also find out not all features help the classification algorithms; hence their avoidance is also desired sometimes.

Once the features have been extracted, a data analyst picks suitable classification algorithms. For activity recognition, the usual trend is to employ supervised learning algorithms as it is easier to label the data collected as a certain type of activity manually. However, unsupervised learning approaches can also be employed. Two of the best performing machine learning algorithms in field of activity recognition are Random Forest and K NN (Peterek).

Following the proper techniques in Machine Learning, we will evaluate the performance and cost of these two algorithms in our experiments. We will then try to understand what

makes one approach superior to the other. The results are presented in ways to clearly depict the distinctions.

**III. Data Collection:**

Several fitness devices such as Fit bit and Nike+ Fuel Band, and smartphones are extensively used in fitness data collection. These devices rely on one or multiple accelerometers to gather such information (Bogdanov). The acquired data is raw and time-series in nature. The sampling rate depends on user's settings; higher the sampling, more precise and concurrent the activity recognition becomes, however with an additional cost in computation and battery power.

A simple data acquisition method records the value of accelerometers (the acceleration) in *x*, *y*, and *z* axes few number of times every second. Figure *1* shows some sample readings of the accelerometer in *x*, *y*, and *z* axes.

| | x | y | z |
|---|---|---|---|
| 0 | 1992 | 2006 | 1655 |
| 1 | 1983 | 2000 | 1645 |
| 2 | 1989 | 2000 | 1645 |
| 3 | 1985 | 1999 | 1652 |
| 4 | 1991 | 2003 | 1653 |
| 5 | 1988 | 1997 | 1651 |
| 6 | 1986 | 2001 | 1649 |

**Figure 1 An example of accelerometer data sampling**

For our project, we decided to use an open source dataset that was made public by UCI Machine Learning Repository (Activity Recognition from Single Chest-Mounted Accelerometer Data Set). A single chest mounted accelerometer was employed in sampling readings on *x, y* and *z* axes at *52 Hz* frequency. Fifteen users were involved in the data collection process. The dataset is well labeled into following classes: - *1:* Working at Computer *2*: Standing Up, Walking and Going up down stairs *3:* Standing *4:* Walking *5:* Going Up Down Stairs *6:* Walking and Talking with Someone *7:* Talking while Standing.

The data is in its raw format and is massive in nature. For multiple users the size of the data collected is huge – 52 sensor readings in a second for a user. The challenge now is to

group the data collected into timeframes that map to certain activities and then come up with the characteristic features that define that class of activity; we will dive into this problem in next section.

**IV. Feature Selection:**

The raw data collected by the sensors are not readily usable. Some data processing is desired in order to run the machine learning algorithms. The signal has both the AC (high frequency) component and the DC (low frequency) component that is related to the influence of gravity. Preliminarily, we looked for some anomalies in the datasets. It was found out that there are data without any labels assigned to them; hence we discarded the data which were not assigned any labels to eliminate inconsistencies. In this section, we describe the process we employ to extract relevant features for our work.

First we divide the datasets into several windows of *1* second each. For the *52* Hz sampling dataset, it means a window consists of *52* sensor readings. We employed window *50%* overlapping as this was suggested to provide success (Bao). Then for each window following *18* features were extracted:

1. Averages of acceleration for *x*, *y*, and *z* axes (3 counts)
2. Standard deviation for *x*, *y*, and *z* axes (3 counts)
3. Energy wrt. each axis (3 counts)
4. Correlation coefficients *xy, yz,* and *zx* (3 counts)
5. Averages of acceleration in *x*, *y*, and *z* axes after filtering out the AC component as described in (cite the process). (3 counts)
6. Standard deviation of acceleration in *x,y,* and *z* axes after filtering out the AC component (3 counts)

Let's briefly describe the features that are extracted in this section. The averages and standard deviations are simple statistical aggregation functions. The Energy function for a window is given as:

$Energy = \Sigma [|x_i|^2 / |w|]$

Where $x_i$ symbol represents the Fast Fourier transform of the acceleration in x- axis and w is the window size (Ravi). Energy feature is similarly computed for the two other axes.

The correlation between each pair of axes is computed so as to understand the independence between the axes. A certain type of activity might involve acceleration in certain axis only (Ravi).

Corr(*x,* y*) =cov(x,y) / (st.d_x st.d_y)*

The features listed in bullets *6* and *7* are computed after separating the AC component of the signals in the dataset. The DC component of the signal is related to the impact of gravity on the motion in that axis hence used to identify static postures (Manini). The AC DC filtering is done using a digital filtering(Mannini).

**V. Machine Learning Algorithms for Activity Recognition**

After the feature selection and data preprocessing is done, time comes to pick a machine learning algorithm that suits our problem. The problem here is to correctly identify each window period in our data as one of the classes – i.e. the activity type. This is an example of classification problem. For our case, our data is labeled during data collection and feature extraction; hence we are looking for supervised learning algorithms.

The literature reveals that some of the choices of Machine Learning algorithms for activity recognition are: Random Forest, K Nearest Neighbors, Neural Networks, and Linear Discriminant Analysis methods (Peterek, Ravi, and Mannini). All these methods are competitive and generate satisfactory results for the purpose. For our project, we have decided to select Random Forest and K Nearest Neighbors methods.

**Random Forest**

Random Forest is one of the ensemble learning methods which is used in classification and regression problem (Breiman). It is an ensemble system in the sense that it is comprised of several correlated binary trees (classification or regression). Random forest is known to be more precise than other ensemble methods such as Adaboost, and its resilience against the outliers.

The two parameters for algorithm are *n* the number of trees, and *k* number of variables or features involved in the trees. Since the problem in hand is a classification problem for us, we will be constructing several decision trees that make classifications. Let's look at the Random Forest Algorthm next .

**Random Forest Algorithm**

1. *construct n bootstrap subset of samples from the original dataset*
2. *construct a classification tree for each subset such that the best split is picked among several possible predictors. The predictors are randomly picked different from bagging where all the predictors are considered. These classification trees serve as the weaker classifiers that are going to ensemble together to make predictions*
3. *New data prediction is done by aggregating the prediction of the n trees. The majority result will derive the final result.*

The goal of random forest is to reduce the variance of each binary classifier that constitute it. The error rate of Random Forest is obtained by predicting data not in a bootstrap subset (out of bag) using the classifier for that bootstrap iteration. The decrease in this error rate will confirm the working of Random Forest (Liao).

An important information derived from Random Forest is the proximity measure that measures the fraction of trees for which data points *x* and *y* lie on same terminal node.

Random Forest works for both supervised and unsupervised learning methods. For our purpose we will be using the supervised learning approach.

**K-Nearest Neighbors (KNN)**

K-Nearest Neighbors is an inferential machine learning technique that takes as input *k* closest sample points and a sample is assigned the class of the most common class in the neighborhood (Altman). For the regression case, however the output is the average value of all target values in the neighborhood. KNN is regarded a lazy learning method as all computations are suspended until the classification happens. One of the simplest machine learning algorithm for classification KNN however is susceptible to local characteristics of data rather than overall view of it.

The membership to a neighborhood can be determined by a metrics such as Euclidean distance or hamming distance. The selection of parameter *k* has some effect on the performance of the algorithm. Higher *k* reduces the classification noise and also potentially eliminates the biasness towards highly weighing classification population.

Let's look at a simple KNN algorithm next.

**K- NN Algorithm**

1. *Select a positive integer k, and one sample x to be classified*
2. *Computing the Euclidean distance, hamming distance or any other distance metrics, return k samples from the dataset that are closest to x*
3. *Identify the most common target classification C in the neighborhood*
4. *Classify x as C*

If *k* is selected to be *1*, the K-NN algorithms simply associate a sample with its nearest neighbor's class.

Both K-NN and Random Forest algorithms are similar in the sense that both can be looked as weighted neighborhood systems (Lin). The difference is on the building blocks of the two systems. Random Forest is depending on several predictors and binary trees to make its inferences, however K-NN algorithm relies on locality of data samples for predictions.

## VI. Experiments

We aim to test the two algorithms – Random Forest and K Nearest Neighbors (K-NN) using pre-existing Weka tool to evaluate the performances. Our experiment data come from the UCI Machine learning repository, and we design several sets of experiments to fully understand the training, crossvalidation, and testing results. The classifiers were experimented with different proportions of training and testing data combinations. The experiments and their details are elaborated in Table 1.

**Table 1 Experiment Setup**

| SN | Descriptions |
|---|---|
| *Experiment 1* | Train and test with all the same data. |
| *Experiment 2* | Train and test using a 10-fold cross validation method on the set of extracted features. |
| *Experiment 3* | Train with only 1 subject data and test with different subject data. |
| *Experiment 4:* | Train with all the data and test with a new dataset. |

The results for our activity recognition experiments are presented in Table 2 in the Results section. Overall, Random Forest classifier offered better performance, yielding

*100%, 91.24%, 50.24%* and *63.18%* accuracies for the experiments mentioned above in the given order. Three variations of K-NN parameters were evaluated (K-NN1, K-NN3, and K-NN5). For comparison purpose, we run another set of experiments with AdaBoostM1 and Multilayer perception classifiers, but the results were as bad as K-NN5. Hence, we didn't include those results here. In the following we will provide details to our experimental datasets, system resources, and result interpretations.

**Datasets**

We used the existing dataset published by UCI Machine Learning data repository to evaluate the performances of classifiers on Human Activity Recognition. The dataset consists of data from a chest mounted wearable accelerometer. Accelerometer Data are collected from *15* participants while they perform *7* activities. The dataset was originally intended for Activity Recognition research purposes.

*Relevant Information:*

- ✓ The dataset collects data from a wearable accelerometer mounted on the chest
- ✓ Sampling frequency of the accelerometer: 52 Hz
- ✓ Accelerometer Data are Uncalibrated
- ✓ Number of Participants: 15
- ✓ Number of Activities: 7
- ✓ Data Format: CSV

*Dataset Information:*

- ✓ Data separated by participants
- ✓ Series of sequential number, *x* acceleration, *y* acceleration, *z* acceleration, and label
- ✓ Labels codified by numbers
  - 1: Working at Computer
  - 2: Standing Up, Walking and Going up\down stairs
  - 3: Standing
  - 4: Walking
  - 5: Going Up\Down Stairs
  - 6: Walking and Talking with Someone
  - 7: Talking while Standing

**System Resources**

Here, we briefly list the system resources that were used in our evaluations.

*Tools:* Weka 3.6

*Operating System:* Windows/Linux machine

*RAM:* 2GB

*Processor:* Intel Core 1.90GHz

*Java:* Java 5.0 or higher

## VII Results:

Table 2 shows the summary results for Activity Recognition experiments evaluated by different classifiers. We see Random Forest perfectly fits the training dataset as compared to the K-NN learning's. For all the experiments, Random Forest outperforms K-NN. The lowest accuracy for Random Forest of *50.24%* happens when it is trained with user A's data and tested upon a different user B' data. Experiment *4* which is the best depiction of real life scenario still gives *63.18%* accuracy. This information is also depicted in Figure *2*.

**Table 2 Overall Accuracy Results**

| | Accuracy | | | |
|---|---|---|---|---|
| | **Experiment 1** | **Experiment 2** | **Experiment 3** | **Experiment 4** |
| **Classifiers** | Test with Training dataset (%) | Cross Validation-10 folds (%) | Train with 1 subject and test with another (%) | Train with all the data and test with new dataset (%) |
| Random Forest | **100** | **91.24** | **50.24** | **63.18** |
| KNN-1 | **100** | 82.24 | 30 | 43.8 |
| KNN-3 | 88.54 | 80.43 | 28.29 | 40.4 |
| KNN-5 | 88.58 | 80.56 | 30.74 | 38.52 |

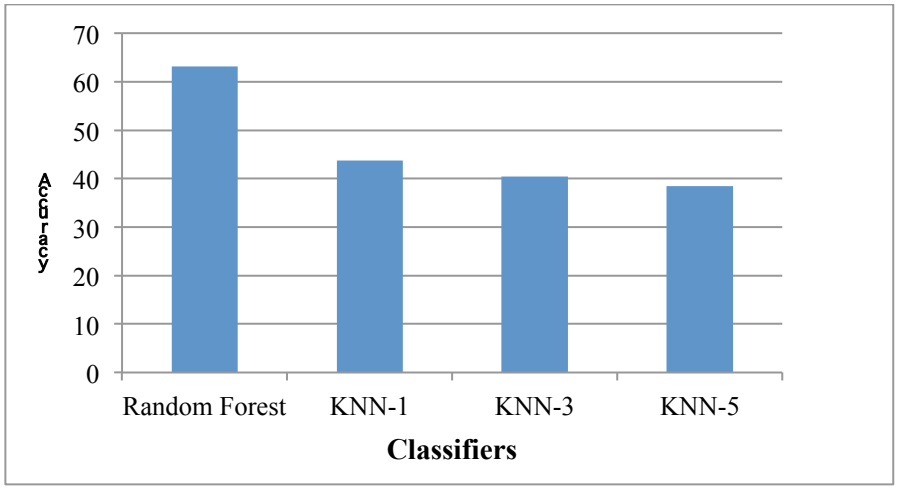


**Figure 2. Overall Accuracy Comparison for Experiment 4**

Table 3 and 4 show the confusion matrix results for the two algorithms. We see that classes 2 , 3, and 5 which are a complex or combinations of one or more of classes 1, 4, 6, and 7 not accurately identified. The predictions have overlapped because of confusion in the activities that make those complex tasks.

**Table 3 Confusion Matrix for KNN**

| | | Prediction | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Actual | **1** | **471** | 10 | 248 | 71 | 7 | 3 | 1025 |
| | **2** | 26 | **0** | 22 | 12 | 0 | 0 | 83 |
| | **3** | 240 | 27 | **30** | 27 | 5 | 7 | 647 |
| | **4** | 51 | 2 | 7 | **10** | 3 | 0 | 850 |
| | **5** | 20 | 0 | 6 | 2 | **1** | 4 | 129 |
| | **6** | 38 | 7 | 3 | 0 | 0 | **12** | 251 |
| | **7** | 0 | 0 | 0 | 0 | 0 | 0 | **2464** |

**Table 4 Confusion Matrix for Random Forest**

| | | Prediction | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Actual | **1** | **1237** | 17 | 2 | 12 | 0 | 0 | 567 |
| | **2** | 122 | **0** | 2 | 2 | 0 | 0 | 17 |
| | **3** | 197 | 2 | **35** | 98 | 6 | 0 | 645 |
| | **4** | 70 | 0 | 18 | **566** | 1 | 0 | 268 |
| | **5** | 0 | 0 | 13 | 17 | **1** | 0 | 131 |
| | **6** | 0 | 2 | 0 | 0 | 0 | **7** | 302 |
| | **7** | 0 | 0 | 0 | 0 | 0 | 0 | **2464** |

Figure 3 shows the class-wise accuracy for Random Forest and KNN-1 classifiers. We see Random Forest is equally or more accurate than KNN for all the classes.

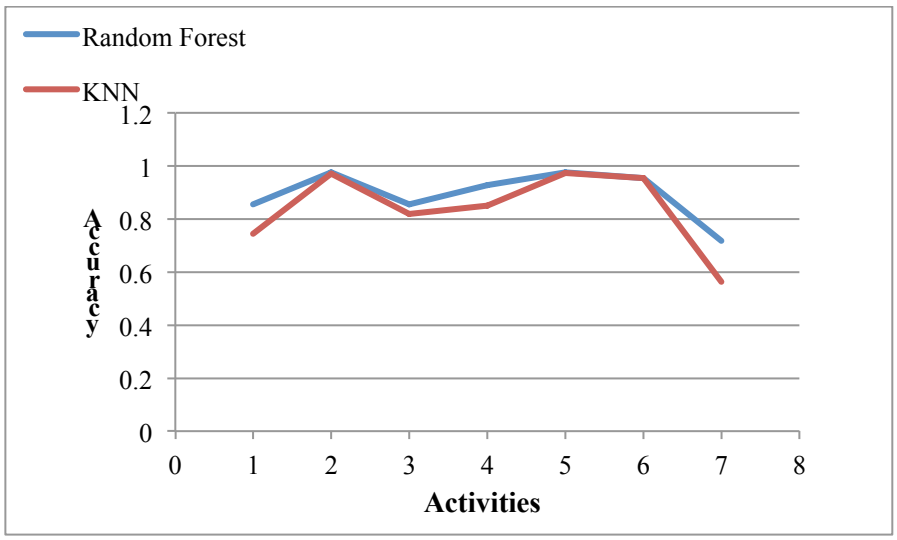


**Figure 3 Class-wise accuracy for Experiment 4**

Figure 4 gives a comparison of Receiver Operation Characteristic (ROC) analysis for Random Forest and KNN-1 algorithm for experiment 4. We see that Random Forest is predicting classes 1, 4, and 7 (Working at Computer, Walking, and Talking while Standing respectively) with more probability precision than KNN, while KNN is doing better job in predicting classes 2, 3, 5, and 6 (Standing Up Walking and Going up\down stairs Walking, Going up\down stairs, Standing, and Walking and Talking with Someone respectively).

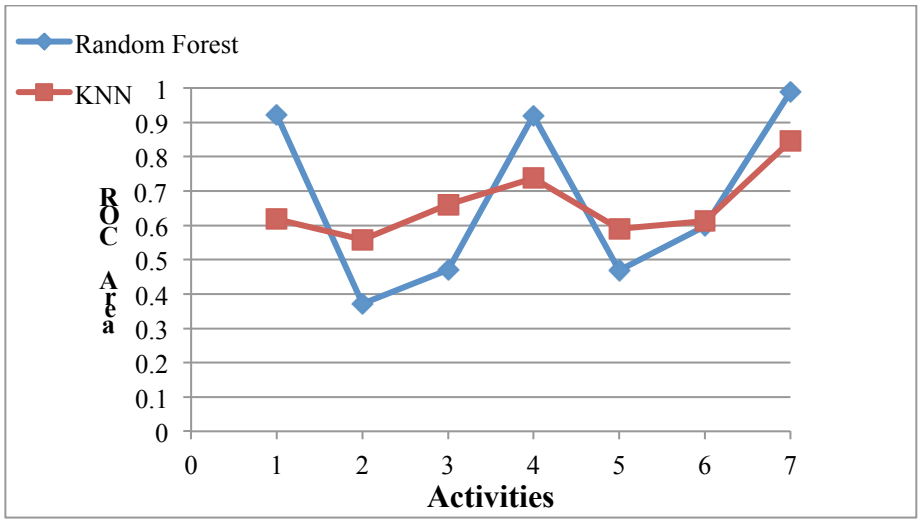


**Figure 4 ROC Area Comparison**

Figure 5 gives the ROC area under the curve for the cross validation experiment. We see that Random Forest outperforms KNN for predicting all the classes. Random Forest is performing very well in identifying the activities in this experiment.

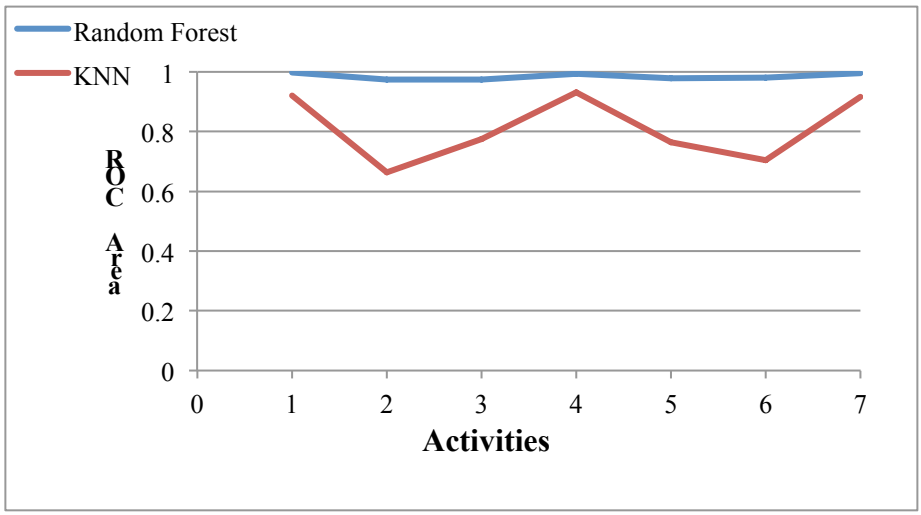


**Figure 5 ROC Area comparison for Cross validation with 10-folds**

Figure 6 gives the time cost for running the different algorithms. We see that Random Forest is more expensive than KNN algorithms. The reason behind Random Forest being more expensive is because it is predicting classes depending upon several weaker binary trees (*100* trees in our case).

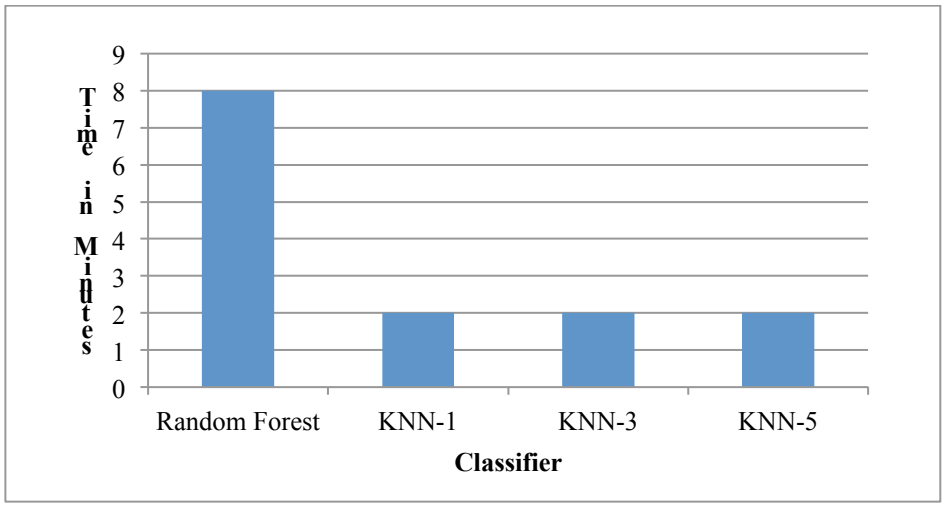


**Figure 6 Time Cost Comparison**

**Discussions:**

From the Results section, we saw that overall performance of Random Forest is better than KNN in most of the cases; however, random forest is more time costly.

In experiment 4, for the simpler classes (distinct classes such as working at a computer vs walking or Going up/down the stairs), the classifiers perform really well – Random Forest evidently performing very well. However for complex classes (colliding properties such as walking vs. walking and talking with someone) probability of correctly predicting the classes is lower for both the algorithms. To identify these complex activities, perhaps more accelerometer or other sensor data might help in better prediction.

With cross validation, as we saw in experiment 2, Random Forest is very accurate. Our results clearly put Random Forest as a better choice for activity recognition purpose.

We believe that simplifying the complex tasks i.e. either separating them from the dataset or merging them with the related simpler activities would have increased the accuracy for both the algorithms.

**VII Conclusion:**

The science behind popular wearable technology is an interesting one. The wearable devices work well only if the underlying machine learning algorithms that generate the prediction models are performing accurately. We showed that among the two popular classification algorithms- Random Forest and K-NN algorithm, the former outperforms the latter. Current devices and their prediction models are fairly accurate at identifying simple human activities. The future improvement in the area of wearable technology should require improvement in the sensor technologies as well so as optimization of existing machine learning algorithms so as to precisely predict both simpler and complex human activities.

**VIII References:**

"Activity Recognition from Single Chest-Mounted Accelerometer Data Set." *UCI Machine Learning Repository: Activity Recognition from Single Chest-Mounted Accelerometer Data Set*. N.p., n.d. Web. 06 Nov. 2015. <https://archive.ics.uci.edu/ml/datasets/Activity+Recognition+from+Single+Chest-Mounted+Accelerometer>.

Altman, N. S. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression." *Taylor & Francis*. N.p., n.d. Web. 09 Dec. 2015. <http://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879>.

Bao, Ling, and Stephen S. Intille. "Activity Recognition from User-Annotated Acceleration Data." *Lecture Notes in Computer Science Pervasive Computing* (2004): 1-17. Web.

Bogdanov, Vik. "How Wearables Work: Fitness Bracelets and Activity Trackers." *Fitness Bracelets and Activity Trackers*. N.p., n.d. Web. 04 Dec. 2015. <http://intersog.com/blog/tech-tips/how-wearables-work-fitness-bands-and-activity-trackers/>.

Breiman, Leo. "Random Forests." - *Springer*. N.p., n.d. Web. 09 Dec. 2015. <http://link.springer.com/article/10.1023%2FA%3A1010933404324>.

Carus, Juan Luis, Victor Palaez, Gloria Lopez, and Vanesa Lobato. "Result Filters." National Center for Biotechnology Information. U.S. National Library of Medicine, n.d. Web. 26 Oct. 2015.

Casale, Pierluigi, Oriol Pujol, and Petia Radeva. "Human Activity Recognition from Accelerometer Data Using a Wearable Device." ResearchGate. N.p., n.d. Web. 26 Oct. 2015.

Karantonis, Dean M., and Nigel H. Lovell. "Implementation of a Real-time Human Movement Classi er Using a Triaxial Accelerometer for Ambulatory Monitoring." IEEE Xplore. N.p., n.d. Web. 26 Oct. 2015.

Li, Aiguang, Lianying Ji, Shaofeng Wang, and Jiankang Wu. "Physical Activity Classi

cation Using a Single Triaxial Accelerometer Based on HMM." IEEE Xplore. N.p., n.d. Web. 26 Oct. 2015.

Liao, Andy, and Matthew Weiner. "Classification and Regression by RandomForest." (n.d.): n. pag. Web. <http://www.bios.unc.edu/~dzeng/BIOS740/randomforest.pdf>.

Lin, Yi, and Yongho Jeon. "Random Forests and Adaptive Nearest Neighbors." N.p., n.d. Web. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.153.9168>.

Liu, Ming. "A Study of Mobile Sensing Using Smartphones." *A Study of Mobile Sensing Using Smartphones*. N.p., n.d. Web. 09 Dec. 2015. <http://www.hindawi.com/journals/ijdsn/2013/272916/>.

Liu, Wei, and Daoli Huang. "A Survey on Context Awareness." *IEEE Xplore*. N.p., n.d. Web. 02 Dec. 2015.

Mannini, Andrea, and Angelo Maria Sabatini. "Machine Learning Methods for Classifying Human Physical Activity from On-Body Accelerometers." *MDPI*. N.p., n.d. Web. 15 Nov. 2015. <http://www.mdpi.com/1424-8220/10/2/1154>.

Meyer, Francois G. "Physical Activity Classi cation Using a Single Triaxial Accelerometer Based on HMM." IEEE Xplore. N.p., n.d. Web. 26 Oct. 2015. Penhaker, Marek, Petr Gajdos, and Pavel Dohnalek. "Comparison of Classi cation Algorithms for Physical Activity Recognit." Ion. N.p., n.d. Web. 26 Oct. 2015.

Peterek, Tomas, Marek Penhaker, Petr Gajdos, and Pavel Dohnalek. "Comparison of Classification Algorithms for Physical Activity Recognit." *Ion*. N.p., n.d. Web. 02 Dec. 2015. <http://link.springer.com/chapter/10.1007/978-3-319-01781-5_12>.

Ravi, Nishkam, and Nikhil Dandekar. "Activity Recognition from Accelerometer Data." *Department of Computer Science, Rutgers University* (n.d.): n. pag. Web. <http://paul.rutgers.edu/~nravi/accelerometer.pdf>.

Tapia, Emanuel Munguia, Stephen S. Intille, and Kent Larson. "Multiple People Activity

Recognition Using Simple Sensors." Proceedings of the 1st International Conference on Pervasive and Embedded Computing and Communication Systems (2011): n. pag. Http://web.media.mit.edu/~intille/papers-les/TapiaIntilleLarson04.pdf. Web.