HUMAN ACTIVITY RECOGNITION USING SMARTPHONE DATASET

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology
in
Computer Science and Engineering/Information Technology

By

Umang Agarwal (131260)
Shikhar Dhwaj (131261)

Under the supervision of

Dr. Yashwant Singh

To

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled **" Human activity recognition using smartphone dataset"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2016 to December 2016 under the supervision of **Dr. Yashwant Singh** (Associate Professor , Department of computer science and engineering).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)
Umang Agarwal, 131260

(Student Signature)
Shikhar Dhwaj, 131261

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)
Dr. Yashwant Singh
Associate Professor
Department of computer science and engineering
Dated

Acknowledgement

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to Dr. Yashwant Singh for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards our parents for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to lab attends for giving me such attention and time.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

Contents

List of Abbreviations

- PCA- Principal Component Analysis
- UCI- University of California, Irvine
- HAR- Human Activity Recognition
- NA- Not Applicable
- EDA- Exploratory Data Analysis
- CSV- Comma separated values
- SVM-Support Vector Machine
- KNN-K Nearest Neighbour
- WRT-With Respect to

List of figures

List of graphs

List of tables

Abstract

This project depicts recognition of Human activities using data generated from user's Smart phone. We have used data available at University of California Machine Learning repository to recognize six human activities. These activities are Standing, Sitting, Laying, Walking, Walking upstairs and Walking downstairs. Data is collected from embedded accelerometer, gyroscope and other sensors of Samsung Galaxy S II Smart phone. Data is randomly divided into 7:3 ratios to form training and testing data set respectively. Dimensionality reduction is done using Principal Component Analysis technique. Activity classification is done using Machine Learning models namely Random Forest, Support Vector Machine, Artificial Neural Network and K-Nearest Neighbour. We have compared accuracy and performance of these models using confusion matrix and random simulation.

Chapter 1 INTRODUCTION

1.1 Introduction

Research in Human Activity Recognition is in massive demand due to its applications in Health care domain, Computer Vision, Household safety and Robot Learning [1]. A huge amount of money can be saved if sensors collect and monitor data of patients. System can automatically send reports to doctor in case of any abnormal behaviour.

We have used sensors of low cost Smartphones to identify human activities. Extensive growth in popularity, accessibility and computation power of smartphones makes it ideal candidate for non-invasive body attached sensor [2].

Smartphones have become irreplaceable part of human life. People carry Smartphones throughout the day. This enables smartphone sensors to collect data and hence lets system to detect human activity.

Human Activity Recognition (HAR) aims to identify the actions carried out by a person given a set of observations of him/her and the surrounding environment. Recognition can be accomplished by exploiting the information retrieved from various sources such as environmental or body-worn sensors. Our Aim is to classify the given activities in the form of a dataset into six labels namely sitting, standing, walking, climbing up, climbing down and laying. We present analysis of method for classifying activities, such as walking up stairs or standing, using data from a gyroscope and accelerometer. Analysis is informed by a visualization of the data. We analyse the differences in error rates between different methods.

1.2 Problem Statement

Identifying human activities from Smartphone dataset has proved to be complex task due to large dimensions of dataset. Various Machine learning techniques have been used previously to identify human activities. We have proposed a system that reduces dimensions of Smartphone dataset and uses Machine learning algorithms in an optimised manner to produce efficient result.

1.3 Objectives

a. To clean the given dataset.
b. To study the dataset and reduce its dimensions using PCA.
c. To identify suitable machine learning algorithm to analyse given dataset.
d. To analyse the performance of different Machine Learning algorithms using following process-
    - Apply Machine learning algorithms to the cleaned dataset and generate the confusion matrix.
    - Calculate error rate from the confusion matrix.
    - Calculate time taken to train the model.
    - Compare error rate of different Machine Learning algorithms.
    - Compare time taken by different Machine Learning algorithms.

1.4 Methodology

We have collected data from University of California Machine Learning repository [3]. Data is imported, cleaned and normalized. In order to increase correctness and performance of our system we have reduced dimensions of our original dataset using Principal Component Analysis(PCA) technique. Reduced data is then processed through various supervised Machine Learning algorithms like Random Forest, Support Vector Machine, Artificial Neural Network and K-Nearest Neighbour to classify data into six categories namely Sitting, Standing, Laying, Walking, Walking Upstairs and Walking Downstairs. Correctness of the system is determined by generating confusion matrix and by random simulations.

1.5 Organisation

Chapter 2 describes previous works associated with our work. Methodology and System Design have been described in Chapter 3. Performance of different Machine Learning Algorithms have been compared in Chapter 4 and Chapter 5 deduces conclusion from the analysis performed.

CHAPTER 2 LITERATURE SURVEY

2.1 Human Activity Recognition

The objective of Human activity recognition is to detect the actions performed by a person from a given set of the data about him/her and his surrounding environment. A lot of research is being done in the field of Human activity recognition which human behaviour is interpreted by deducing features derived from movement, place, physiological signals and information from environment etc... Environmental and sensors which are worn by the person generates the information which is used to interpret the activity. Good precision can be obtained from sensors which are worn in waist, wrist, chest and thighs. But these sensors are quite uncomfortable and cannot provide long term solutions. [22]

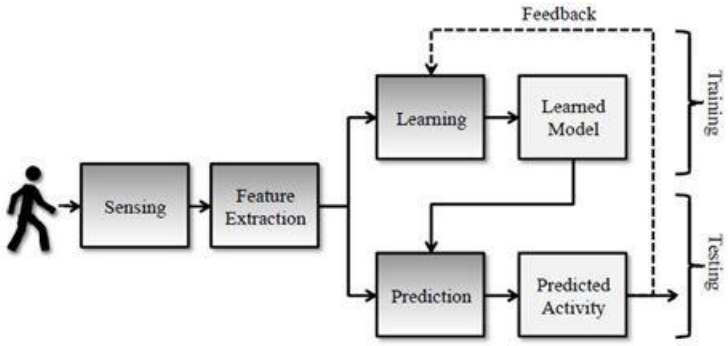


Figure 1: HAR development Process

Smartphones have brought up huge research opportunities in human-centred applications. The latest smartphones came with various embedded sensors like microphones, cameras, accelerometers, gyroscopes, etc. These commonly available devices provide automatic and unobstructed monitoring of daily life activities apart from telephony services. [1] HAR is part of a much larger concept known as context-aware computing or ubiquitous

computing. Ubiquitous computing does its job in the same way as HAR, which is by collecting data from users and assisting them.

No.	Static	Time (sec)	No.	Dynamic	Time (sec)
0	Start (Standing Pos)	0	7	Walk (1)	15
1	Stand (1)	15	8	Walk (2)	15
2	Sit (1)	15	9	Walk Downstairs (1)	12
3	Stand (2)	15	10	Walk Upstairs (2)	12
4	Lay Down (1)	15	11	Walk Downstairs (1)	12
5	Sit (2)	15	12	Walk Upstairs (2)	12
6	Lay Down (2)	15	13	Walk Downstairs (3)	12
			14	Walk Upstairs (3)	12
			15	Stop	0
				Total	**192**

Figure 2: Protocol of activities for the HAR Experiment

HAR has wide use in the field of nursing, military, entertainment and daily life. For example, HAR can be used to assist a soldier with their action reports, to report an abnormality of a person to the hospital staff etc. Integrated motion sensors are being used to provide valuable data to the athletes, which in turn will help them to improve their performance. Clearly HAR is becoming an integral part of our daily life.

HAR using smartphones has many advantages like device portability, comfort and unobstructed sensing. The drawback of this approach it consumes and share services with other applications on the phones which may become a problem for devices with low resources.

2.2 Sensors used

2.2.1 Accelerometer

An accelerometer is a device which is used for measuring static or dynamic acceleration forces. We can find out the angle by which the device is tilted by determining the static acceleration due to gravity. And we can sense the direction of movement by measuring the dynamic acceleration. Accelerometer may work on the piezoelectric effect or by

sensing the changes in capacitance. Other less used methods use piezoresitive effect, hot air bubbles, and light. Piezoelectric accelerometers have microscopic crystals which generates voltage when they are under acceleration. [24]

Accelerometer in smartphone contains seismic mass circuit which is made of silicon. The mass changes its orientation according to the orientation of the device.

2.2.2 Gyroscope

Gyroscopes are small and cheap devices which are used for the measurement of rotational motion that is angular velocity. Degrees per second (°/s) or revolutions per second (RPS) are the units of angular velocity. [25]

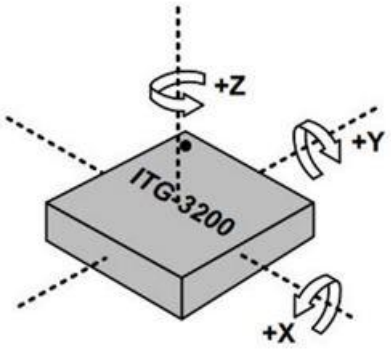


Figure 3: Axis of angular rotation for gyroscopes

A triple axis MEMS gyroscope can measure rotation in 3 axes: x, y, and z. gyroscopes with single or dual axis are becoming less popular as the triple axis gyroscopes have become cheaper and small. Very low-current electrical signals are produced, which are later amplified, when the resonating mass in gyroscope changes its position due to rotational motion. MEMS gyroscopes operate in mA or sometimes micro-ampere range. [25]

2.3 Wearable sensor placement

The placement of wearable sensors defines how and where different sensors are attached. It has a direct effect on the observations taken for a person. The ideal positioning of

sensors is still a topic of debate. Sensors can be attached to different parts of the body but sternum, lower back and waist are most common locations. Closer the placement of sensors to the centre of mass, the better is the representation of movements. The main motive of researchers is to minimise the number of sensors attached to the body, finding an optimum position and to maintain a fairly high accuracy rate of recognition.

Cleland et al. [30] placed 6 sensors on chest, hip, wrist, thigh, foot and lower back to determine which location is best suited to collect more accurate data for recognition. They found that placing sensor on the hip represents the activities more accurately. Chamroukhi et al. [31] has deduced from their research that the accuracy of recognition can be greatly increased by placing accelerometers on both upper and lower parts. He also found that a system of 3 sensors located on chest, thigh and ankle gave least error rate. Also different experiments have shown that the accuracy of recognition increases with the number of sensors attached to a person.

2.4 Signal processing

Tri-axial linear acceleration and angular velocity were captured by the accelerometer and gyro of smartphone at 50Hz sampling rate. To reduce noise the signals were passed through median filter and butter-worth filter with 20Hz cut-off frequency which is appropriate as 99% of the energy is limited to 15Hz. The captured signals have two components generated by the body and gravity. The gravitational component has 0.3 Hz corner frequency which was calculated by earlier experiments. [22]

2.5 Pre-processing

Data pre-processing is inseparable part of data mining. The objective of pre-processing is to filter data, to replace the corrupted values and to extract or select features. In window technique the sensor signals are segmented into small time segments. It is used for feature extraction. After this, on each distinct window, segmentation and classification algorithm is applied. There are 3 types of windowing techniques: (a) sliding (b) event-defined (c) activity-defined. The most useful technique is sliding window for real-time applications [29].

2.6 Principal Component Analysis (PCA)

When a dataset has a huge number of attributes, we may face following situations-

-Many variables could be discovered which are interrelated.

-If we decide to run a model on whole data it would lead to high error rate.

- We may start to device new method to extract some important attributes.

We have used Principal Component Analysis to reduce the number of features in our dataset. The principal component analysis (PCA) is designed to reduce the dimensionality of a large data set containing a huge number of interrelated variables and retain as much as possible of the variation present in the data set [26]. Principal component analysis works by converting the variables in the given dataset to a new set of variables, the principal components (PCs). The Principal Components are uncorrelated and are ordered in terms of the variation present in all of the original variables. In this ordered set of principal components, the first few components contain most of the variation which was present in original dataset [26]. This behavior is shown due to Eigen value decomposition of data co-variance [27] [28].

2.7 Machine learning algorithms

Different supervised, semi-supervised and unsupervised algorithms can be used to solve the problem of real-time recognition. Different algorithm had proved themselves useful in different applications, which is why there is no clear distinction on which algorithm is more appropriate than others. The energy and memory consumption and complexity of the problem varies and form a set of conditions which decide which algorithm is more suitable for a particular problem. [23]

2.7.1 Random forests

Random Forest are an ensemble learning method for classification, regression and other tasks [33]. It works by constructing many decision trees at training time and outputs the class that is the mean prediction of decision trees. Random forest prevents over fitting of

data [32]. Combination of bagging method and randomisation improves the performance of algorithm.

It is observed that larger the labelled data the more accurate the algorithm works. In [34], Random forest outpaced SVM and Naïve Bayes in classifying movement of body in cars, trains and walking. [29]

2.7.2 K-Nearest Neighbour

KNN is an instance based classifier. It operates on the principal that classification of unknown instances can be done by relating the unknown instance to known instance on basis of some function [35,36]. This function is similarity or distance function. Foerster et al. [37] were the first to use the k-NN algorithm to classify 9 activities.

2.7.3 Support Vector Machine (SVM)

Support Vector Machines are based on decision hyperplanes that define decision boundaries. A decision plane separates two set of objects having different class membership. Support Vector Machine aims to maximize decision boundary between hyperplanes. Krause et al. [38] found that frequency domain features provided better results than the time domain features with the use of SVM to classify 8 activities. [29]

2.7.4 Artificial Neural Networks

An artificial neural network mimics the working of neurons in a biological brain. It derives the relationship between the input signals and output signals. [39] The most common method to train data is Back-propagation method. In this method error at the output is determined and then it is propagated back into the network.

Chapter 3 System Development

3.1 Introduction

In this chapter, we will describe step by step process used to design our system. The Machine learning problem we are dealing with falls in category of classification problems. We have used R Studio version 3.2.5 on windows 10 platform to carry out entire experimentation.

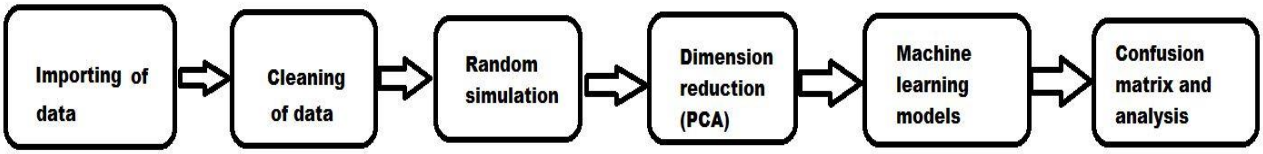


Figure 4: Flow of System Design of Human Activity Recognition

3.1 Importing and Cleaning Data

Data was collected from experimentation carried out by group of 30 volunteers. Each person was supposed to perform six activities (Walking, walk up, walk down, Laying, Standing and Sitting). They had used embedded sensors of Samsung Galaxy SII Smartphone to collect data. Dataset was loaded into R studio as comma separated file. Imported dataset had 586 features.

The following states original features of dataset.

The dataset contains:

* A 561-feature vector with time and frequency domain variables.

* Its activity label.

* An identifier of the subject who carried out the experiment.

Feature names were modified and were used as headers. Missing data and NA values were imputed using mean for that particular feature.

Data was normalized using the following formula:

$$zi = \frac{xi - \bar{x}}{sd}$$

Where xi is the an instance of data, $\bar{x}$ is mean of that feature column and sd is standard deviation of that feature column

test

Filter

	tBodyAcc.mean...X	tBodyAcc.mean...Y	tBodyAcc.mean...Z	tBodyAcc.std...X	tBodyAcc.std...Y	tBodyAcc.std...Z	tBodyAcc.mad...X	tBodyAcc.mad...Y	tBodyAcc.mad...Z	tBodyAcc.max...X	tBody
2	0.27841883	-0.0164105680	-0.123520190	-0.998245280	-0.975300220	-0.960321990	-0.99880719	-0.9749143700	-0.957686220	-0.9430675100	
7	0.27945388	-0.0196407760	-0.110022150	-0.996921040	-0.967185930	-0.983117830	-0.99700268	-0.9660967100	-0.983116270	-0.9409866300	
10	0.28058569	-0.0099602983	-0.106065160	-0.994803440	-0.972758400	-0.986243870	-0.99540462	-0.9736632200	-0.985641950	-0.9400275100	
11	0.27688027	-0.0127218050	-0.103438320	-0.994815110	-0.973076920	-0.985357020	-0.99550927	-0.9739479600	-0.985172470	-0.9400275100	
12	0.27622817	-0.0214413020	-0.108202340	-0.998245950	-0.987213760	-0.992726590	-0.99825127	-0.9859965400	-0.993181880	-0.9439057800	
13	0.27845700	-0.0204147610	-0.112731720	-0.999134880	-0.984680040	-0.996274240	-0.99907654	-0.9829370200	-0.996410310	-0.9439057800	
15	0.29794572	0.0270939080	-0.061668123	-0.988640790	-0.816698600	-0.901906530	-0.98895795	-0.7942804200	-0.888014600	-0.9259766900	
18	0.28013490	-0.0139169510	-0.106370480	-0.997694920	-0.987515670	-0.990407440	-0.99801432	-0.9879544800	-0.992190120	-0.9420759800	
22	0.27715238	-0.0179833280	-0.106601170	-0.997763220	-0.989957270	-0.996585670	-0.99829082	-0.9896687000	-0.996700450	-0.9414724200	
23	0.27567630	-0.0212642340	-0.110801220	-0.997862110	-0.990090760	-0.994592570	-0.99833345	-0.9894726600	-0.994484510	-0.9445667200	
24	0.27920020	-0.0177144270	-0.109161350	-0.998389290	-0.987307840	-0.990831590	-0.99886852	-0.9867713100	-0.989637390	-0.9436751900	
39	0.23715407	0.0078251224	-0.122837910	-0.979953580	-0.866193410	-0.968290470	-0.98017878	-0.8823158700	-0.966096710	-0.9396369300	
41	0.28150539	-0.0184348590	-0.111392870	-0.995468040	-0.984330220	-0.990995140	-0.99597751	-0.9823093700	-0.990283570	-0.9397887700	
42	0.27843225	-0.0196543020	-0.107970290	-0.994390290	-0.984562780	-0.991985780	-0.99505023	-0.9845219500	-0.992617050	-0.9397887700	
44	0.27905615	-0.0162609700	-0.112815650	-0.995018770	-0.970446310	-0.989266120	-0.99481764	-0.9651497800	-0.990246360	-0.9417154100	
48	0.27746121	-0.0174910160	-0.106359450	-0.996428430	-0.984673120	-0.990682660	-0.99673775	-0.9831506700	-0.989338010	-0.9385725700	
52	0.40347433	-0.0150744040	-0.118167390	-0.914811150	-0.895231120	-0.891748110	-0.91769589	-0.9246235400	-0.905894780	-0.7851039800	
59	0.28020640	-0.0183962600	-0.107488630	-0.996474930	-0.994068760	-0.991861310	-0.99729491	-0.9943383800	-0.993439600	-0.9394526400	
61	0.27672941	-0.0172095480	-0.105637970	-0.994788110	-0.991031320	-0.993187220	-0.99588986	-0.9906248800	-0.992794740	-0.9334585100	
64	0.27937115	-0.0176449530	-0.108181110	-0.995236830	-0.995810090	-0.994430450	-0.99551468	-0.9953381400	-0.993719970	-0.9392213100	
65	0.01901615	-0.0070373566	-0.028333356	-0.661293610	-0.713352570	-0.701155170	-0.69394813	-0.7065575100	-0.755421840	-0.8912835600	
66	0.34020850	-0.0364529840	-0.106258190	-0.958596480	-0.908055820	-0.984669480	-0.95685496	-0.8984938500	-0.987138760	-0.8912835600	
72	-0.27706634	-0.6840965900	0.346657720	-0.596410470	0.024682958	-0.160404490	-0.63181950	-0.0688567320	-0.235159590	-0.8662961700	
84	0.25546822	0.0212190630	-0.048949431	-0.224536990	0.022312942	-0.113196240	-0.25062407	-0.0219882870	-0.099186333	-0.0734760940	
87	0.31263404	-0.0263677490	-0.130951210	-0.353098600	-0.017382150	-0.127807600	-0.39527374	-0.0543888000	-0.095763059	-0.2903971600	
88	0.27691540	-0.0354862100	-0.080569176	-0.262941830	0.111277810	-0.212700270	-0.30989024	0.0896067960	-0.233325980	-0.0685826050	
90	0.30699599	-0.0137869800	-0.180161970	-0.239029850	0.072865343	-0.247204440	-0.28254589	0.0080078037	-0.261673890	-0.0912851760	

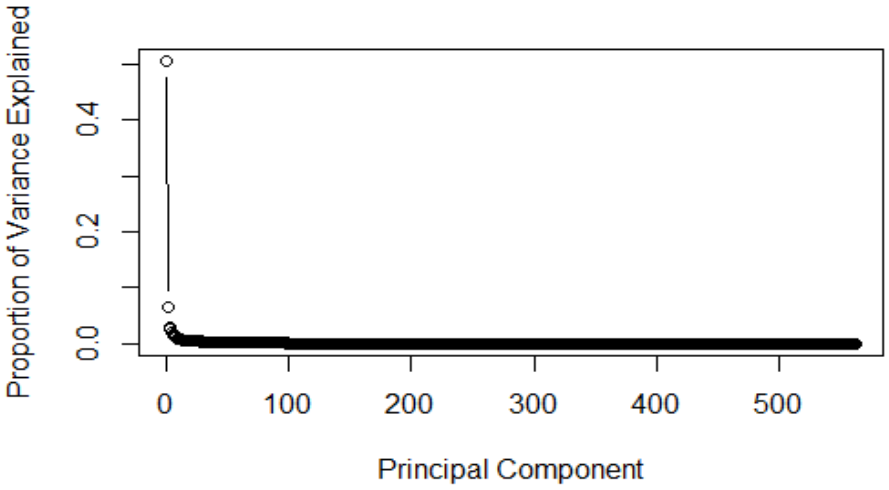
Figure 5: Snapshot of cleaned Dataset

3.2 Random Simulation

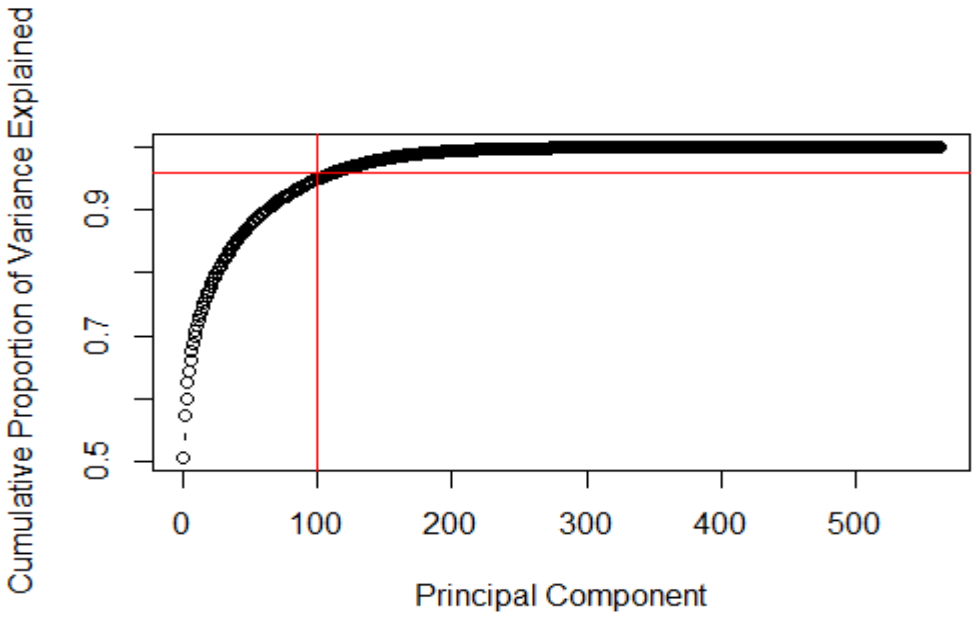
Random Simulation is used to test accuracy of predictive models and prevent overfitting and underfitting of data. The technique involves randomly dividing the dataset into training and testing set in the ratio 7:3. The whole simulation is repeated 50 times to improve accuracy of model as per the statistical Central limit theorem [10] [11]. Testing dataset provides us with approximation of real time data and provides us with a mechanism to test stability of our model in real life scenario.

3.3 Dimensionality Reduction

We have used Principal Component Analysis to reduce the number of features in our dataset. The principal component analysis (PCA) is designed to reduce the dimensionality of a large data set containing a huge number of interrelated variables and retain as much as possible of the variation present in the data set [41]. Principal component analysis works by converting the variables in the given dataset to a new set of variables, the principal components (PCs). The Principal Components are uncorrelated and are ordered in terms of the variation present in all of the original variables. In this ordered set of principal components, the first few components contain most of the variation which was present in original dataset [41]. This behaviour is shown due to Eigen value decomposition of data co-variance [11] [12]. The following figure shows this transformation.



Graph 1: Proportions of Variances wrt Principal Component



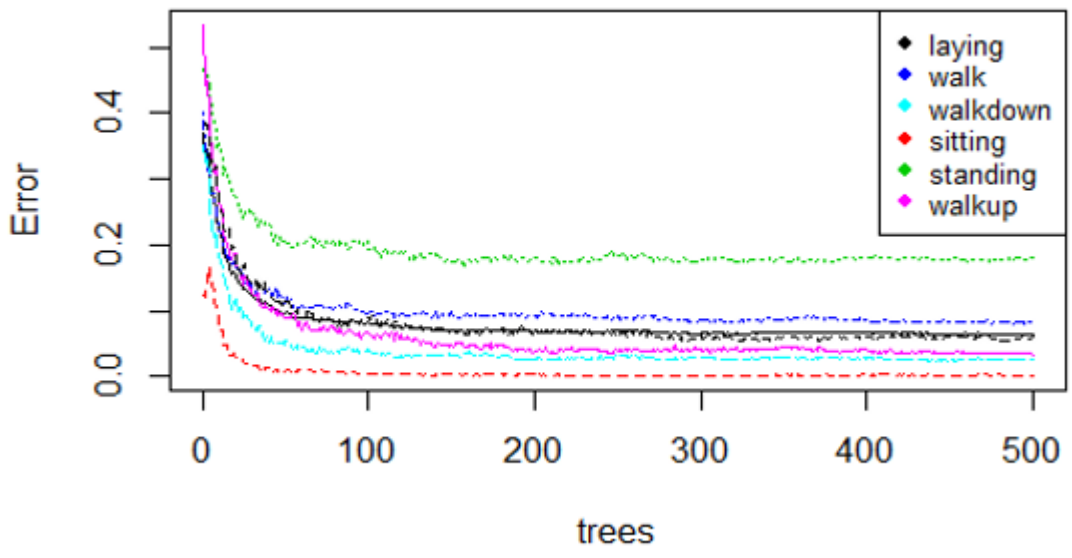
Graph 2: Cumulative Proportions of Variances

It is observed from the figure that first 100 Principal Components account for more than 95% of the variance in dataset, so we choose only first 100 components and disregard

other components. The changes were made in training dataset and exact transformation was applied to testing dataset.

3.4 Random Forest

Random Forest are an ensemble learning method for classification, regression and other tasks [14]. It works by constructing many decision trees at training time and outputs the class that is the mean prediction of decision trees. Random forest prevents over fitting of data [13]. In R programming environment, 500 decision trees are constructed by default. By our experimentation, we had observed that there was no need for constructing 500 decision trees. Our model will work in the same way and will produce the same results if we had constructed only 80 trees as shown in the figure below.



Graph 3: Error rate measurements w.r.t to Number of Trees

Above figure shows that error rate is constant after construction of 80 trees, so we need only 80 decision trees.

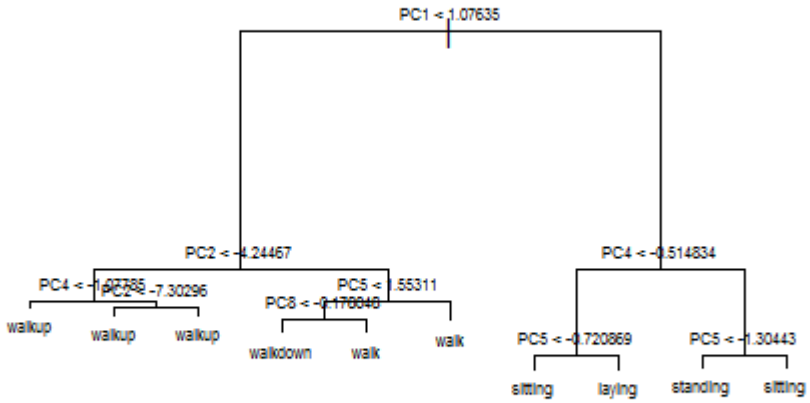
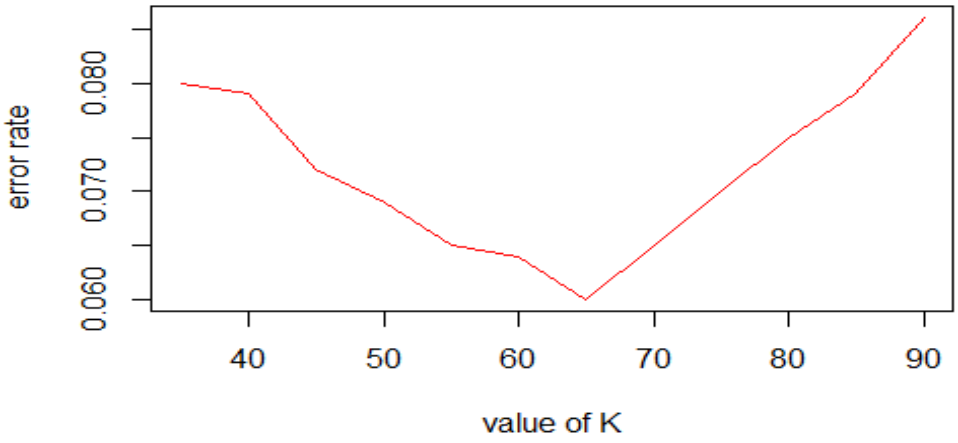


Figure 6: A sample tree of the forest constructed

3.5 K-Nearest Neighbour(KNN)

KNN is an instance based classifier. It operates on the principal that classification of unknown instances can be done by relating the unknown instance to known instance on basis of some function [15,16]. This function is similarity or distance function.

We have used Euclidean distance function to approximate our learning function. We had determined value of K by plotting graph of error rate vs K value as shown in figure below.



Graph 4: Variation of error rate w.r.t value of K

From the figure we had found that error rate decreases from K=40 to K=65 and reaches minima at K=65.

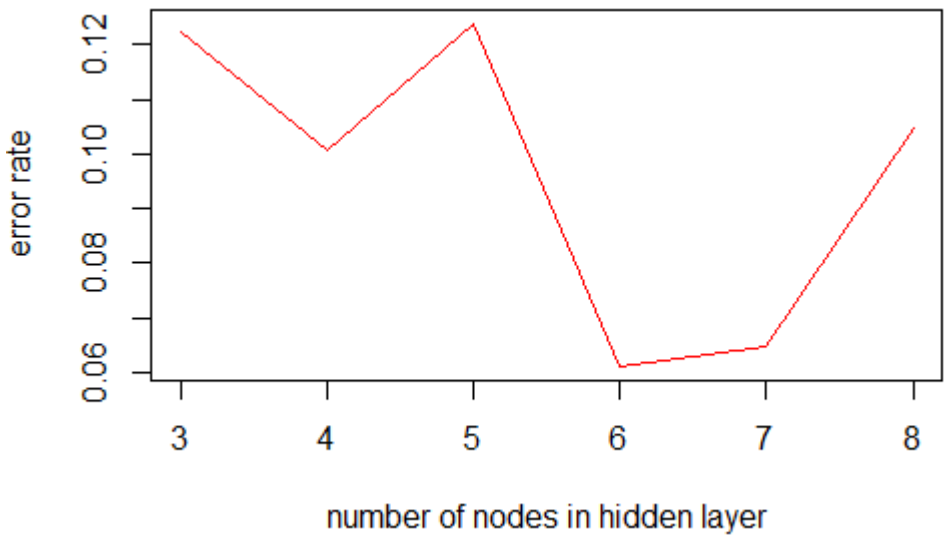
3.6 Support Vector Machine

Support Vector Machines are based on decision hyperplanes that define decision boundaries. A decision plane separates two set of objects having different class membership. Support Vector Machine aims to maximize decision boundary between hyperplanes. We had used "e1071" SVM library in R to train our dataset [17]. SVM algorithm takes kernel type, cost and gamma as parameters. We had chosen Gaussian(radical) kernel as similarity function.

Value of cost is kept 1 as it makes regularization term constant and prevents overfitting of data [18]. Value of gamma determines shape of vector hyperplane. It is kept as 0.013758, as value of gamma is equal to 1/ (number of features).

3.7 Artificial Neural Network

Artificial Neural Network (ANN) models the relationship between a set of input signals and an output signal using a model derived from our understanding of how a biological brain reacts to stimuli from inputs [40]. We have used nnet package to train our dataset, which is used for feed-forward neural networks with a single hidden layer [20]. The nnet package trains the artificial neural network using backpropagation method. In this method error at the output is determined and then it is propagated back into the network. To minimize the error resulting from each neuron, the weights are updated [21]. The Linout parameter was set False by default as we are dealing with a classification problem [19].



Graph 5: Variation of error rate w.r.t number of nodes in hidden layer

From the above figure it is observed that error rate is minimum is at number of nodes in hidden layer equal to 6. Therefore, we have assigned value of size parameter as 6.

3.8 Prediction of Testing Dataset

After calculating Principal Components of training set, we need to predict testing data using these Principal Components. This appears to be very simple but, we need to understand few points here:

1. Applying PCA to whole data at once will "leak" features of training dataset into testing dataset and will have an effect on prediction ability of our model [12].

2. Applying PCA on testing and training dataset separately will result in vectors that have different direction. Hence, will give inappropriate results [12].

So, what do we do now?

We will apply transformation to the testing set in the same direction as training set along with same center and scaling feature.

Chapter 4 Performance Analysis

In this section we have compared performance of different machine learning models based on confusion matrix and time taken to train the model. Time taken to train the model is calculated by taking mean of 30 simulations.

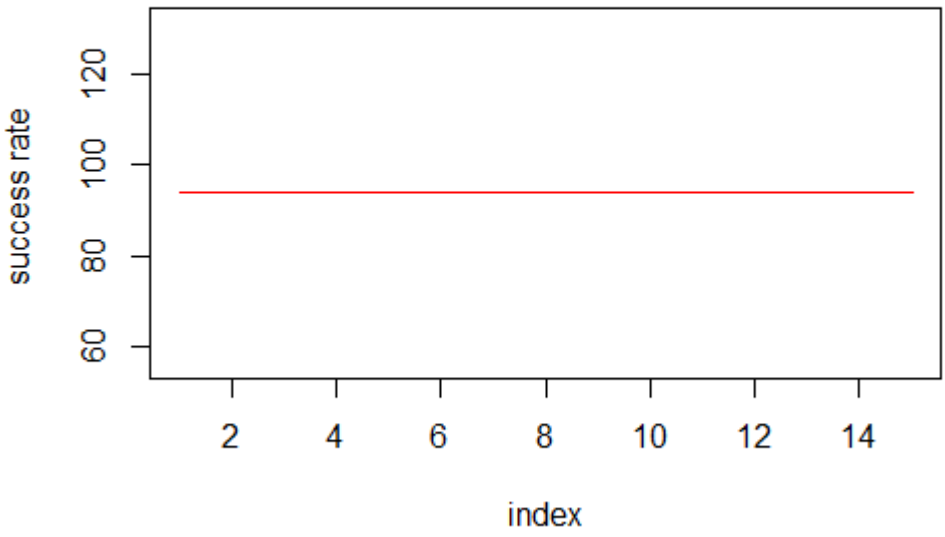
4.1 Random Forest

Table 1: Confusion matrix of data tested with Random Forest Model

Result/references	Laying	Sitting	Standing	Walk	Walk up	Walkdown
Laying	420	0	0	0	0	0
Sitting	14	326	52	0	0	0
Standing	0	35	373	0	0	0
Walk	0	0	0	368	3	2
Walk up	0	0	0	2	284	6
Walkdown	0	0	0	7	17	297

From the confusion matrix it is observed that percentage error rate is 6.25% and percentage success rate is 93.75%.

Time taken to train the model is 10.56 seconds.



Graph 6: Variation of success rate in Random Forest Model

From the above figure, it is observed that there is no variation in success rate of Random Forest Model.

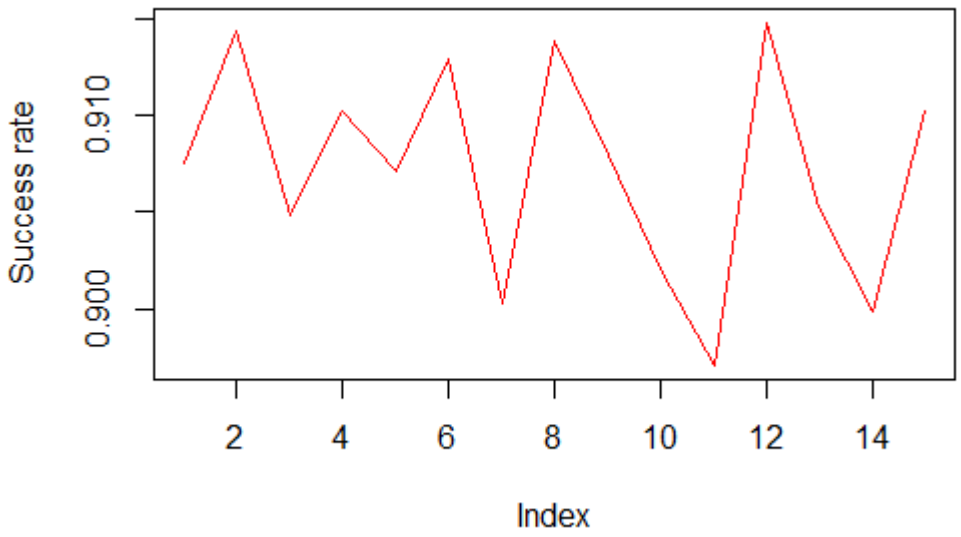
4.2 K-Nearest Neighbour

Table 2: Confusion matrix of data tested with K Nearest Neighbour Model

Result/references	Laying	Sitting	Standing	Walk	Walk up	Walkdown
Laying	421	4	0	0	0	0
Sitting	1	269	23	0	0	0
Standing	10	104	407	0	0	0
Walk	0	0	0	365	17	8
Walk up	0	0	0	7	259	5
Walkdown	2	0	0	1	16	287

From the confusion matrix it is observed that percentage error is 8.98 % and percentage success rate is 91.02%.

Time taken to train the model is 2.7 seconds.



Graph 7: Variation of success rate in K Nearest Neighbour

From the above figure, it is observed that success rate of K nearest neighbour varies from 91.5% to 89.7%.

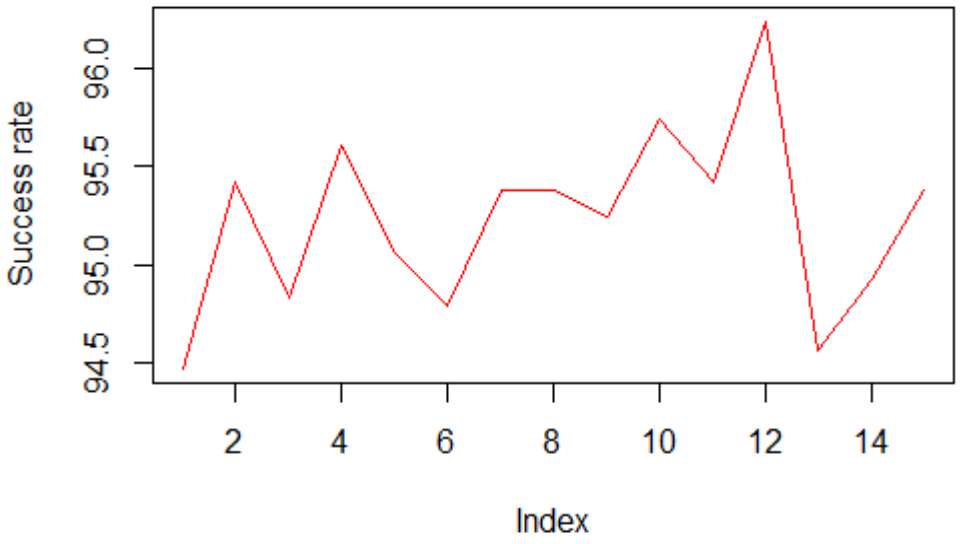
4.3 Support Vector Machine

Table 3: Confusion matrix of data tested with Support Vector Machine Model

Result/references	Laying	Sitting	Standing	Walk	Walk up	Walkdown
Laying	429	0	0	0	5	0
Sitting	0	333	42	0	2	0
Standing	0	35	393	0	2	0
Walk	0	0	0	362	11	0
Walk up	0	0	0	0	292	0
Walkdown	0	0	0	0	5	295

From the confusion matrix it is observed that percentage error is 4.63 % and percentage success rate is 95.37%

Time taken to train the model is 12.9 seconds.



Graph 8 Variation of success rate in Support Vector Machine

From the above figure, it is observed that success rate of support vector machine varies from 94.5% to 96.4%.

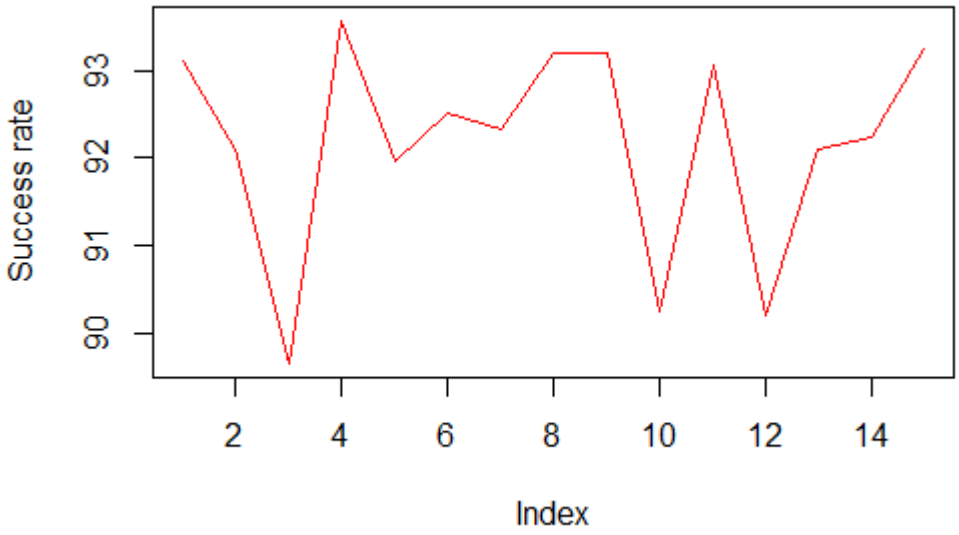
4.4 Artificial Neural Network

Table 4: Confusion matrix of data tested with Artificial Neural Network

Result/references	Laying	Sitting	Standing	Walk	Walk up	Walkdown
Laying	432	0	2	0	0	0
Sitting	8	318	50	1	0	0
Standing	1	48	374	3	0	4
Walk	0	0	0	361	3	9
Walk up	0	1	0	4	279	8
Walkdown	0	0	0	2	5	293

From the confusion matrix it is observed that percentage error is 6.76 % and percentage success rate is 93.24%

Time taken to train the model is 5.2 seconds.



Graph 9: Variation of success rate in Artificial Neural Network

From the above figure, it is observed that success rate of artificial neural network varies from 93.5% to 89.6%.

Chapter 5 Conclusion

We had studied the Human Activity Recognition dataset and had learned Machine Learning Algorithms Namely Random Forest, K-Nearest Neighbour, Support Vector Machine and Artificial Neural Networks. We had reduced dimensions of our dataset from 586 features to 100 features by applying Principal Component Analysis. We had performed Data Analysis and found that Support Vector Machine had greatest efficiency (95.37%) in predicting Human Activities. SVM is most efficient because we have used Gaussian kernel in reduced dataset. Least time to train the machine learning model was taken by K-Nearest Neighbour (2.7 seconds) because it is least complex and it uses Euclidian distance function. Maximum time to train the model was taken by Support Vector Machine (12.9 seconds).

- Future Aspects: The model developed can be used to predict Human Activity on real time basis. An application on Android Platform can be used to convey measurements and run the model on those measurements. This application has aspects in monitoring health and performance of athletes etc.

- Innovative Idea: We have found that processing data through PCA (Principal Component Analysis) resulted in optimised model creation which led to improvement in performance.

References

1. Adil Mehmood Khan, Young-Koo Lee, Sungyoung Y. Lee, and Tae-Seong Kim,A triaxial accelerometer-based physical -activity recognition via augmented-signal features and a hierarchical recognizer, Information Technology in Biomedicine, IEEE transactions on information technology in biomedicine, Volume 14, NO. 5, September 2010

2. Charles Arthur,"Smartphone explosion in 2014 will see ownership in India pass US.", accessed on 30/3/2017,http://www.theguardian.com/technology/2014/jan/13/smartphone-explosion-2014-india-us-china-firefoxos-android

3. "Human activity recognition using smartphone dataset", accessed on 1/11/2016, https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones

4. Lester, Jonathan and Choudhury, Tanzeem and Borriello, Gaetano ,"A practical approach to recognizing physical activities", accessed on 4/10/2016, https://www.cs.cornell.edu/~tanzeem/pubs/JonathanLester_EDAS-1568973904.pdf

5.T.B. Moeslund, A.Hilton,V.Kruger, "A survey of advances in vision-based human motion capture and analysis, Computer Vision Image Understanding", Volume 104, Issues 2–3 ,December 2006.

6.L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," Pers Comput., Lecture Notes in computer Science, vol. 3001, pp. 1–17, 2004.

7.Kwapisz, Jennifer R and Weiss, Gary M and Moore, Samuel A, "Cell phone-based biometric identification", Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference, 10.1109/BTAS.2010.5634532, September 2010

8. Kwapisz, Jennifer R and Weiss, Gary M and Moore, Samuel A, "Activity recognition using cell phone accelerometers", accessed on 21/11/2016, http://www.cis.fordham.edu/wisdm/includes/files/sensorKDD-2010.pdf

9. Ramiro J.Caro, "Central Limit Theorem Simulator" , accessed on 15/11/2016,

https://rpubs.com/RamiroJC/CLT_Slides

10. Central Limit Theorem ,accessed on 12/11/2016, http://www.investopedia.com/terms/c/central_limit_theorem.asp

11." Principal component analysis" ,accessed on 10/11/2017, https://en.wikipedia.org/wiki/Principal_component_analysis

12." Practical Guide to Principal Component Analysis (PCA) in R & Python", accessed on 5/11/2016,

https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/

13.Leo Breiman , Adele Cutler, “Random forests”, accessed on 10/11/2016,

https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

14. Tavish Srivastava, “Introduction to Random forest – Simplified”, accessed on 8/11/2016,

https://www.analyticsvidhya.com/blog/2014/06/introduction-random-forest-simplified/

15. “k-nearest neighbors algorithm”,accessed on 12/2/2017, https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

16. Ola Soder, “knn classifiers”,accessed on 4/3/2017,

http://www.fon.hum.uva.nl/praat/manual/kNN_classifiers_1__What_is_a_kNN_classifier_.html

17. David Meyer , Evgenia Dimitriadou , Kurt Hornik , Andreas Weingessel , Friedrich Leisch , Chih-Chung Chang ,Chih-Chen Lin ,”Department of Statistics ,Probability theory group,TU Wein”,aceessed on 2/3/2017,

https://cran.r-project.org/web/packages/e1071/index.html

18. Yu-Wei, David Chiu,” Machine Learning with R Cookbook”,

19. Gary H.Y ,” Notes for nnet, tree”,accessed on 2/2//2017,

http://statisticsr.blogspot.in/2008/10/notes-for-nnet.html

20. Brian Ripley , William Venables,”Package nnet”,accessed on 20/1/2017,

https://cran.r-project.org/web/packages/nnet/nnet.pdf

21. Aarshay Jain,” Fundamentals of Deep Learning – Starting with Artificial Neural Network”, accessed on 15/1/2017,
https://www.analyticsvidhya.com/blog/2016/03/introduction-deep-learning-fundamentals-neural-networks/

22. Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz, “A Public Domain Dataset for Human Activity Recognition Using Smartphones”, European Symposium on Artificial Neural Networks, 24-26 April 2013.

23. Jorge L. Reyes-Ortiz, Alessandro Ghio, Davide Anguita , Xavier Parra, Joan Cabestany, Andreu Catal, “Human Activity and Motion Disorder Recognition: Towards Smarter Interactive Cognitive Environments”, European Symposium on Artificial Neural Networks, 24-26 April 2013.

24. TONI_K, "Accelerometer Basics", accessed on 20/9/2016,

https://learn.sparkfun.com/tutorials/accelerometer-basics

25. Toni_K, "Gyroscope Basics" ,accessed on 20/9/2016,

https://learn.sparkfun.com/tutorials/gyroscope

26. I.T. Jolliffe ,"Principal Component Analysis", Second Edition, springer, April 2002

27. " Principal component analysis" ,accessed on 10/11/2017, https://en.wikipedia.org/wiki/Principal_component_analysis

28. "Practical Guide to Principal Component Analysis (PCA) in R & Python", accessed on 5/11/2016,

https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/

29. Physical Human Activity Recognition Using Wearable Sensors Ferhat Attal 1 , Samer Mohammed 1,*, Mariam Dedabrishvili 1 , Faicel Chamroukhi 2 , Latifa Oukhellou 3 and Yacine Amirat 1 Received: 11 September 2015; Accepted: 8 December 2015; Published: 11 December 2015 Academic Editor: Vittorio M.N. Passaro

30. Cleland, I.; Kikhia, B.; Nugent, C.; Boytsov, A.; Hallberg, J.; Synnes, K.; McClean, S.; Finlay, D. Optimal Placement of Accelerometers for the Detection of Everyday Activities. Sensors 2013, 13, 9183–9200.

31. Chamroukhi, F.; Mohammed, S.; Trabelsi, D.; Oukhellou, L.; Amirat, Y. Joint segmentation of multivariate time series with hidden process regression for human activity recognition. Neurocomputing 2013, 120, 633–644.

32.Leo Breiman , Adele Cutler, "Random forests", accessed on 10/11/2016,

https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

33. Tavish Srivastava, "Introduction to Random forest – Simplified", accessed on 8/11/2016,

https://www.analyticsvidhya.com/blog/2014/06/introduction-random-forest-simplified/

34. Breiman, L.; Friedman, J.; Stone, C.J.; Olshen, R.A. Classification and Regression Trees; CRC press: Boca Raton, FL, USA, 1984.

35. "k-nearest neighbors algorithm",accessed on 12/2/2017, https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

36.Ola Soder, "knn classifiers",accessed on 4/3/2017,

37.Foerster, F.; Smeja, M.; Fahrenberg, J. Detection of posture and motion by accelerometry: A validation study in ambulatory monitoring. Comput. Hum. Behav. 1999, 15, 571–583.

38. Krause, A.; Ihmig, M.; Rankin, E.; Leong, D.; Gupta, S.; Siewiorek, D.; Smailagic, A.; Deisher, M.; Sengupta, U. Trading off prediction accuracy and power consumption for context-aware wearable computing. In Proceedings of the 2005 Ninth IEEE International Symposium on Wearable Computers, Osaka, Japan, 18–21 October 2005; pp. 20–26

39.Brett Lantz, "Machine learning with R", Second edition, Packt publishing, July 2015

40. Brett Lantz, "Machine learning with R", Second edition, Packt publishing, July 2015

41. I.T. Jolliffe ,"Principal Component Analysis", Second Edition, springer, April 2002

Appendix

Code 1

```
load("C:/Users/user/Downloads/samsungData.rda")

names(samsungData)

table(is.na(samsungData)) ## no NAs

table(samsungData$subject)

nameVec <- make.names(names(samsungData),unique=TRUE)

names(samsungData) <- nameVec

set.seed(123)

smp_size <- floor(0.70 * nrow(samsungData))

train_ind <- sample(seq_len(nrow(samsungData)), size = smp_size)

train <- samsungData[train_ind, ]

test <- samsungData[-train_ind, ]

par(mfrow=c(1,2))

plot(train[,1],train[,2], col=factor(train$activity),pch=19, cex=0.5)

plot(train[,2],train[,3], col=factor(train$activity),pch=19, cex=0.5)

legend("bottomleft",legend=unique(factor(train$activity)),
col=unique(factor(train$activity)), pch=19, cex=0.8)

pc <- prcomp(train[,-563], center=TRUE, scale=TRUE)

train.data<-data.frame(activity=train$activity,pc$x)

train.data<-train.data[,1:100]

## Construction of Tree

library(tree)

train.tree <- tree(factor(activity)~.,data=train.data)
```

```
par(mfrow=c(1,1))
plot(train.tree)


#Random Forest Constrcution
library(randomForest)
train.rf <- randomForest(factor(activity) ~., data=train.data, prox=TRUE, ntree=100)
train.rf$confusion
text(train.tree, cex=0.5)

#Transformation of testing data according to Principal Component values
test.data<-predict(pc,newdata=test)
test.data<-as.data.frame(test.data)
test.data<-test.data[,1:100]
result <-predict(train.rf,test.data)

test.data$activity=test$activity
references<-test.data$activity

table(references,result)
```

Code 2

```
load("C:/Users/user/Downloads/samsungData.rda")
g=c()
ti=c()
for(i in 1:15){
nameVec <- make.names(names(samsungData),unique=TRUE)
names(samsungData) <- nameVec

  set.seed(123+10*i+100*i)
smp_size <- floor(0.70 * nrow(samsungData))
train_ind <- sample(seq_len(nrow(samsungData)), size = smp_size)
train <- samsungData[train_ind, ]
test <- samsungData[-train_ind, ]

pc <- prcomp(train[,-563], center=TRUE, scale=TRUE)

train.data<-data.frame(activity=train$activity,pc$x)
train.data<-train.data[,1:80]
library("nnet")
a<-Sys.time()
nmodel<-nnet(activity~.,data=train.data,size=7)
b<-Sys.time()
ti<-c(ti,b-a)
test.data<-predict(pc,newdata=test)
test.data<-as.data.frame(test.data)
```

```
test.data<-test.data[,1:80]
result<-predict(nmodel,test.data,type="class")

test.data$activity=test$activity
references<-test.data$activity

t<-table(references,result)
g<-c(g,(t[1,1]+t[2,2]+t[3,3]+t[4,4]+t[5,5]+t[6,6])/sum(t))
}
g<-g*100
plot(g,type = "l",col="red",y="succcess rate",main="variation of success rate")
gv<- var(g)
#mean time taken is 5.2 sec for training the model
#variance in success rate is 0.000149 approx
```

Code 3

```
load("C:/Users/user/Downloads/samsungData.rda")
nameVec <- make.names(names(samsungData),unique=TRUE)
names(samsungData) <- nameVec
g=c()
ti=c()
for(i in 1:15){
  set.seed(123+10*i+100*i)
smp_size <- floor(0.70 * nrow(samsungData))
train_ind <- sample(seq_len(nrow(samsungData)), size = smp_size)
train <- samsungData[train_ind, ]
test <- samsungData[-train_ind, ]
pc <- prcomp(train[,-563], center=TRUE, scale=TRUE)
train.data<-data.frame(activity=train$activity,pc$x)
```

```
train.data<-train.data[,1:80]
library("e1071")
a<-Sys.time()
svm_model <- svm(activity ~ ., data=train.data)
b<-Sys.time()
ti<-c(ti,b-a)
test.data<-predict(pc,newdata=test)
test.data<-as.data.frame(test.data)
test.data<-test.data[,1:80]
result<-predict(svm_model,test.data,type="class")
test.data$activity=test$activity
references<-test.data$activity
t<-table(references,result)
aa<-t[1,1]+t[2,2]+t[3,3]+t[4,4]+t[5,5]+t[6,6]
g<-c(g,aa/sum(t)*100)
}
plot(g,type = "l",col="red")
vg<-var(g)
#mean time taken is 12.9 sec for training the model
#variance in success rate is 0.21 approx
```

Code 4

```
load("C:/Users/user/Downloads/samsungData.rda")
nameVec <- make.names(names(samsungData),unique=TRUE)
names(samsungData) <- nameVec
x=c()
y=c()
ti=c()
for(i in 1:15){
```

```
  smp_size <- floor(0.70 * nrow(samsungData))
  train_ind <- sample(seq_len(nrow(samsungData)), size = smp_size)
  train <- samsungData[train_ind, ]
  test <- samsungData[-train_ind, ]
  pc <- prcomp(train[,-563], center=TRUE, scale=TRUE)
  train.data<-data.frame(activity=train$activity,pc$x)
  train.data<-train.data[,1:80]
  test.data<-predict(pc,newdata=test)
  test.data<-as.data.frame(test.data)
  test.data<-test.data[,1:80]

  library("class")
  a<-Sys.time()
  kmodel <- knn(train = train.data[,-1], test = test.data[,-80],cl = train.data$activity,
k=20+5*i)
  b<-Sys.time()

  t<-table(kmodel,test$activity)

  aa<-t[1,1]+t[2,2]+t[3,3]+t[4,4]+t[5,5]+t[6,6]
  x<-c(x,30+5*i)
  y<-c(y,1-aa/sum(t))
ti<-c(ti,b-a)
}
plot(g,type="l",col="red",ylab="succcess rate",main="variation of success rate")
gv<-var(g)
#mean time taken is 2.737 sec for training the model
#variance in success rate is 0.0000318 approx
```