Prediction of Human Activity Using Machine Learning

by

Sadia Nasrin Tisha
16101101
Benjir Islam Alvee
16101112

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
December 2019

Declaration

It is hereby declared that,

1. The thesis submitted is our own original work while completing degree at BRAC University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Sadia Nasrin Tisha	Benjir Islam Alvee
16101101	16101112

Approval

The thesis titled "Prediction of Human Activity using Machine Learning" submitted by

1. Sadia Nasrin Tisha (16101101)
2. Benjir Islam Alvee (16101112)

Of Fall, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on December 26, 2019.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Mahbubul Alam Majumdar
Professor and Chairperson (CSE)
Department of Computer Science and Engineering
BRAC University

Abstract

Involving machine learning in recognizing human activities is a widely discussed topic of this era. It has a noticeable growth of interest for implementing a wide range of applications such as health monitoring, indoor movements, navigation and location-based services. The process is implemented gradually through several methods obtaining better accuracy than before. The data of human activities can be collected by wifi module, bioharness or wearable device which can be waist, wrist or thighs mounted. The purpose of our research is predicting human activities by classifying sequences of remotely recorded data of well-defined human movements using responsive sensors. The data are collected by a waist mounted device which contains mobile phone sensors (e.g. accelerometer and gyroscope) for observing human activities of different aged people. The observed data are modeled using machine learning and neural network. Here we have used machine learning algorithms which are Support Vector Machine (SVM), K Nearest Neighbour (KNN), Linear Regression, Logistic Regression, Decision Tree, Naive Bayes Classifier and Random Forest Classifier. Moreover, we have also used artificial recurrent neural network (RNN) architecture- Long Short-Term Memory algorithm and Multi Layer Perceptron (MLP) algorithm. Modeling the data using various algorithms and obtaining results accurately are not convenient, because human motions recorded through wearable sensors have variations and complexity. For overcoming these problems we have used four dimension reduction techniques e.g. Principal Component Analysis (PCA), Singular Value Decomposition (SVD), Linear Discriminant Analysis (LDA) and Independent Component Analysis (ICA) for achieving more accurate activity prediction performance with less complex and faster computations.

Keywords: Machine Learning, HAR, Prediction, LSTM, SVM, KNN, MLP, Decision tree, Linear Regression Analysis, PCA, SVD, ICA, LDA.

Dedication

We want to dedicate our research work to our beloved family specially to our loving parents. Without their immense support and belief in us we would not be able to complete our work successfully.

Acknowledgement

Firstly, all praises to Almighty Allah for enabling us to complete our thesis. Secondly, we want to thank and express our heartiest gratitude towards our supervisor Dr. Amitabha Chakrabarty Sir for his immense support and guidance. Our work wouldn't have been completed without his valuable feedback and help. He always motivated us and had belief in us that we can do this. After that, We would like to thank BRAC University IT Department for facilitating us with a desktop PC. Then, we are really thankful to our classmates who gave their time in data collection through our device. Above all, we want to thank each and everyone who has helped us in our thesis work whenever we needed. Finally, we want to acknowledge the efforts and sacrifices of our parents which they do for us every moment. Without their immense nurture and care, completion of this thesis work would have been impossible for us.

Table of Contents

Declaration i

Approval ii

Ethics Statement iii

Abstract iii

Dedication iv

Acknowledgment v

Table of Contents vi

List of Figures viii

List of Tables xi

Nomenclature xii

1 Introduction 1
- 1.1 Motivation . . . 2
- 1.2 Problem Statement . . . 3
- 1.3 Thesis Statement . . . 3

2 Literature Review 5

3 Dataset 9
- 3.1 Existing Dataset . . . 10
  - 3.1.1 Dataset Description . . . 10
  - 3.1.2 Dataset Preprocessing . . . 10
  - 3.1.3 Dimension Reduction Techniques . . . 12
  - 3.1.4 Data Visualization . . . 15
- 3.2 New Dataset . . . 18
  - 3.2.1 Device Description . . . 18
  - 3.2.2 Data Description . . . 20
  - 3.2.3 Data Visualization . . . 21

4 Methodology **24**
4.1 Machine Learning . . . 24
4.1.1 Linear regression . . . 24
4.1.2 Logistic Regression . . . 26
4.1.3 Random Forest Classifier . . . 27
4.1.4 Decision Tree . . . 28
4.1.5 Naïve Bayes (NB) . . . 28
4.1.6 Support vector machine(SVM) . . . 29
4.1.7 K-Nearest Neighbors (KNN) . . . 30
4.2 Neural Network . . . 30
4.2.1 Multilayer perceptron (MLP) . . . 30
4.2.2 Long short-term memory (LSTM) . . . 31
4.3 Dimension Reduction Techniques . . . 34
4.3.1 Principle Component Analysis(PCA) . . . 34
4.3.2 Singular Value Decomposition(SVD) . . . 34
4.3.3 Linear discriminant analysis (LDA) . . . 35
4.3.4 Independent Component Analysis (ICA) . . . 35

5 Result Analysis **36**
5.1 Existing Dataset . . . 36
5.1.1 Linear Regression . . . 36
5.1.2 Logistic Regression . . . 40
5.1.3 Random Forest Classifier . . . 45
5.1.4 Support Vector Machine (SVM) . . . 49
5.1.5 K-Nearest Neighbors (KNN) . . . 54
5.1.6 Naive Bayes classifier (NB) . . . 59
5.1.7 Decision Tree . . . 63
5.1.8 Multilayer perceptron (MLP) . . . 64
5.1.9 Long short-term memory (LSTM) . . . 67
5.2 New Dataset . . . 69
5.2.1 Random Forest Classifier . . . 69
5.2.2 K-Nearest Neighbors(KNN) . . . 69
5.2.3 Support Vector Machine(SVM) . . . 70

6 Discussion **72**

7 Conclusion & Future works **77**

Bibliography **83**

List of Figures

3.1 Workflow of existing dataset 9
3.2 Workflow of new dataset 9
3.3 CSV file of existing dataset 11
3.4 Distribution of PCA components 13
3.5 Distribution of SVD components 13
3.6 Distribution of LDA components 14
3.7 Distribution of ICA components 14
3.8 Scatter and Density Plot of existing dataset 15
3.9 Histogram of existing dataset 16
3.10 Pie chart of existing dataset 16
3.11 Count plot of existing dataset 17
3.12 Heatmap plot of existing dataset 17
3.13 Correlation matrix of existing dataset 18
3.14 Device setup for data collection 19
3.15 CSV file of new dataset 20
3.16 Scatter and Density Plot of new dataset 21
3.17 Histogram of new dataset 22
3.18 Pie chart of new dataset 22
3.19 Count plot of new dataset 23
3.20 Heatmap plot of new dataset 23
3.21 Correlation matrix of new dataset 23

4.1 Linear regression with regression line scattering 24
4.2 Sigmoid function graph 27
4.3 Hyperplane of support vector machine 29
4.4 Visual representation of K factor 30
4.5 ReLU Activation Function Graph 31
4.6 Folded pattern of RNN 32
4.7 Unfolded pattern of RNN 32
4.8 Outline of LSTM 33
4.9 Principle Component Analysis 34
4.10 Singular Value Decomposition matrices 35

5.1 Residual error of linear regression 37
5.2 Result analysis of linear regression 39
5.3 Confusion matrix for logistic regression 40
5.4 Confusion matrix for logistic regression by applying 40 components PCA 41

5.5 Confusion matrix for logistic regression by applying 150 components PCA . . . 41
5.6 Confusion matrix for logistic regression by applying SVD . . . 42
5.7 Confusion matrix for logistic regression by applying LDA . . . 43
5.8 Confusion matrix for logistic regression by applying 20 components ICA . . . 44
5.9 Confusion matrix for logistic regression by applying 150 components ICA . . . 44
5.10 Result analysis of logistic regression . . . 45
5.11 Confusion matrix for random forest classifier . . . 46
5.12 Confusion matrix for random forest classifier by applying 40 components PCA . . . 46
5.13 Confusion matrix for random forest classifier by applying 150 components PCA . . . 46
5.14 Confusion matrix for random forest classifier by applying SVD . . . 47
5.15 Confusion matrix for random forest classifier by applying LDA . . . 48
5.16 Confusion matrix for random forest classifier by applying 20 components ICA . . . 49
5.17 Confusion matrix for random forest classifier by applying 150 components ICA . . . 49
5.18 Result analysis of Random forest classifier . . . 49
5.19 Confusion matrix for support vector machine . . . 50
5.20 Confusion matrix for support vector machine by applying 40 components PCA . . . 51
5.21 Confusion matrix for support vector machine by applying 150 components PCA . . . 51
5.22 Confusion matrix for support vector machine by applying SVD . . . 51
5.23 Confusion matrix for support vector machine by applying LDA . . . 52
5.24 Confusion matrix for support vector machine by applying 20 components ICA . . . 53
5.25 Confusion matrix for support vector machine by applying 150 components ICA . . . 53
5.26 Result analysis of support vector machine (training set) . . . 54
5.27 Result analysis of support vector machine (testing set) . . . 54
5.28 Confusion matrix forK-Nearest Neighbors . . . 55
5.29 Error rate for k value . . . 55
5.30 Confusion matrix for KNN by applying 40 components PCA . . . 56
5.31 Confusion matrix for KNN by applying 150 components PCA . . . 56
5.32 Confusion matrix for KNN by applying SVD . . . 57
5.33 Confusion matrix for KNN by applying LDA . . . 57
5.34 Confusion matrix for KNN by applying 20 components ICA . . . 58
5.35 Confusion matrix for KNN by applying 150 components ICA . . . 58
5.36 Result analysis of K-Nearest Neighbors . . . 58
5.37 Confusion matrix for Naive Bayes . . . 59
5.38 Confusion matrix for Naive Bayes by applying 40 components PCA . . . 60
5.39 Confusion matrix for Naive Bayes by applying 150 components PCA . . . 60
5.40 Confusion matrix for Naive Bayes by applying SVD . . . 61
5.41 Confusion matrix for Naive Bayes by applying LDA . . . 61

5.42 Confusion matrix for Naive Bayes by applying 20 components ICA . . 62
5.43 Confusion matrix for Naive Bayes by applying 150 components ICA . 62
5.44 Result analysis of Naive Bayes 62
5.45 Confusion matrix for decision tree using gini index 63
5.46 Confusion matrix for decision tree using entropy 63
5.47 Confusion matrix for Multilayer perceptron (MLP) 64
5.48 Accuracy plot for MLP training set 65
5.49 Accuracy plot for MLP testing set 65
5.50 Loss plot for MLP training set 65
5.51 Loss plot for MLP testing set 65
5.52 Confusion matrix for Multilayer perceptron (MLP) for 150 components of PCA 66
5.53 Accuracy plot for MLP 150 PCA components training set 66
5.54 Loss plot for MLP 150 PCA components training set 66
5.55 Confusion matrix for Multilayer perceptron (MLP) of SVD 67
5.56 Accuracy plot for MLP (SVD) training set 67
5.57 Loss plot for MLP (SVD) training set 67
5.58 Confusion matrix for LSTM 68
5.59 Training session's progress over iterations for LSTM 68
5.60 Confusion matrix for Random forest classifier 69
5.61 Confusion matrix for KNN 70
5.62 Error for K value in KNN 70
5.63 Confusion matrix for SVM 71

6.1 Result Analysis Graph 73
6.2 Result Analysis Graph for PCA 73
6.3 Result Analysis Graph for SVD 74
6.4 Result Analysis Graph for LDA 74
6.5 Result Analysis Graph for ICA 75
6.6 Result Analysis Graph for new dataset 76

List of Tables

3.1 Time and frequency domain signals obtained from the smartphone sensors 11
3.2 Time of six activities of Human through Waist Mounted Device . . . 19

6.1 List of accuracies of all classifiers 75

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ICA Independent Component Analysis

KNN K-Nearest Neighbors

LDA Linear Discriminant Analysis

$LSTM$ Long Short$_t erm Memory$

MLP Multilayer perceptron

NB Naïve Bayes

PCA Principle Components Analysis

RNN Recurrent Neural Network

SVD Singular Value Decomposition

SVM Support Vector Machine

Chapter 1

Introduction

One of the first important milestones was the first Human Activity Recognition(HAR) solutions which emerged in 2006 that actually used smartphones[1]. At that time, the first experiments were conducted using data analysis of global system for mobile communication(GSM) sensors and accelerometer to measure mobility of devices. During this time, due to limited smartphones, all data processing was carried out on a computer (offline storage). In developing the first collaborative solutions since 2007, the literature has made progress[2][3][4]. The communication and processing model adopted by these solutions follows the following flow: data was collected from smartphones and sent to a server on the Internet where information was shared with users.These data was commonly used to improve the accuracy of machine learning algorithm classification models. It was only until 2008, with the evolution of processing and storage technology in smartphones, that approaches began to appear in which data collection and processing took place on the smartphone itself[5].
In 2009, smartphones were fitted with more sensors, allowing users to identify new activities, such as the detection of human voice. Sound Sense's works[6] represent this scenario very well with recognition of environmental sound-based activities, such as group conversations. At the same time, more studies concentrated on developing healthcare applications as chronic disease detection based on user locomotion issues[7][8][9]. Later on, in 2010, researchers focused on enhancing HAR recognition system specifics such as efficient data collection by continuous sensing to boost battery energy efficiency[10], (ii) enhancing classification models using a series of classifiers[11], and (iii) detecting intermediate intervals between activities[12]. Furthermore, Berchtold et al.[3] suggested the development of a HAR cloud service that would enable the classification models embedded in smartphones to be constantly updated via user feedback. In the beginning of 2011, more specific solutions have emerged, such as the first discussions on the impact of the position and orientation of the smartphone on the user body[13][14]. Henpraserttae et al.[14], for example, found that the mobile in an individual's hand and waist emits different signals and therefore requires different analyzes to identify the same operation. At the same time, as the WISDM server, the first public database was published[15]. Public libraries help validate and compare new HAR methods with existing ones.
The first experiments were published in 2012 concerning the identification of more complex behaviors using smartphones. For example, Dernbach et al.[16] and Khan et al.[17] combined inertial sensor data to recognize everyday (e.g. cooking) and physical activity. Das et al.[18] combined mobile sensor data with smart environ-

ment sensors to recognize the daily activities of users. Many research focused on improving digital applications to ensure that the entire data process takes place on the mobile device[11][12]. In the feature extraction step[19], data fusion techniques for multiple sensors have been applied since 2013. Furthermore, more detailed studies have been conducted to detect accurate lightweight features[20]. Some work in 2014 centered on the data segmentation phase with the goal of measuring the impact of time window size on classification model accuracy[21]. In 2015, the HAR area approached to converge with the application of algorithms for deep learning classification, with the first work being published by Alsheik et al.[22]. From there, the classification models created by the methods of deep learning became the state-of - the-art classification activities. New HAR scientists, based on data streaming[23], appeared in 2016 . At this point, such an issue was viewed as an online learning problem to mitigate the need to store the historically required training of the classification models. However, this approach is based on novel detection with the intention of mapping new activities that are not, by chance, depicted in the models of classification. In addition, in a step to elimination of model classification errors, new studies[24] related to transition-between-activity recognition (e.g. sit-to-stand) have emerged as data referring to transitions are considered noise in the database. Multiple studies[25][26][27][17] conducted comparative analysis among the various HAR solutions of the literature in 2017. Some analyzes attempted to evaluate the efficacy of the features in the classification models generated by machine learning algorithms. Frameworks to encapsulate all the steps and processes previously studied in a generic architecture have been recently introduced in 2018. From there, new API (Application Programming Interface) HAR implementations can emerge to facilitate the creation of HAR applications.

1.1 Motivation

For Human Activity Prediction, smartphone is a globally available key tool for approaching towards data collection of different human activities. The reason behind this are several; such as: smartphone is a portable device which is easy to operate having different embedded sensors (e.g. accelerometer and gyroscope), it has computational features and also the ability to communicate. Smartphone is a convenient option for extracting human motions' data from various types of real-world environments. It is a budget-friendly programmable embedded device which has brought together hardware and software sensors[28]. These sensors can sense different human activities such as: laying, sitting, standing, walking, walking downstairs and walking upstairs irrespective of age of the participant and his/her surroundings. We have developed a waist mounted device for our research using smartphone sensors for collecting data of our targeted six daily activities of human. Reason of choosing waist for the position of our device is waist can be considered as a center point of our body which can percept the position and angular variations of body while we lay down, sit, stand and walk straight, upstairs or downstairs. We have used accelerometer and gyroscope sensors in our device for observing the activities' data. We have also used memory card module for storing data and clock module for recording definite seconds which we took in collecting data of each person.

1.2 Problem Statement

This research aims to predict human daily activities by analyzing data which are being collected through a body worn device containing smartphone sensors. Smartphone sensors are easily available and convenient to record data of different people. Machine learning classifiers and neural network are used for modeling the data and thus predicting the accuracy of the possibility that which data indicates to which activity. In this case complexity of calculation and correctness of results depend on the type of data in the dataset. Some datasets may have lots of features and some features may have high correlation between them. Detecting the type of activity among large number of observations is very challenging. Results may vary in terms of accuracy because of these type of dataset. In this situation we can apply different types of dataset reduction techniques, which analyzes the dataset, identifies the features which are similar, then eliminates one of those after ensuring that the rest of the data can describe the whole process like before. Application of these techniques fastens the computation as well as gives more accurate prediction of activities.

1.3 Thesis Statement

Human activity prediction identifies human activities analyzing a set of observations retrieved from environment or sensors. Sensors can be body worn used in different body parts such as the waist, wrist, chest, thighs etc. Though these sensors are uncomfortable to use but they provide noticeable performance.
Dataset being used in our research consists of data collected through a wearable device in human waist. The device containing smartphone sensors extracts a large set of observations of human activities from the environment. These smartphone sensors are very responsive in terms of recording data. Smartphone is a flexible sensing tool with built-in sensors such as accelerometers, gyroscopes, dual cameras and microphones. All of these provide flexibility while monitoring Activities of Daily Living (ADL).
Activities of daily living are routine activities people do every day without assistance. There are six basic ADLs: eating, bathing, getting dressed, toileting, transferring and continence. The performance of these ADLs is important in determining what type of long-term care and health coverage, such as Medicare, Medicaid or long-term care insurance, a person will need as he or she ages. The body worn device takes data of six human activities such as: Laying, Sitting, Standing, Walking, Walking Downstairs and Walking Upstairs.
These wide range of data are analyzed using dataset reduction techniques which are Principal Component Analysis (PCA), Singular Value Decomposition (SVD), Linear Discriminant Analysis (LDA) and Independent Component Analysis (ICA) whether they have related features or not and then modeled using machine learning classifiers such as: SVM, KNN, Linear Regression, Logistic Regression, Random Forest, Decision tree, Naive Bayes and neural network algorithms which are LSTM and MLP.
Apart from the existing dataset we have developed another dataset with a smaller set of observations. We have developed a waist mounted device using gyroscope and accelerometer for recording data of human movements. Memory card does the task of storing and retrieving data from our device and clock module records the

designated amount of time taken in taking data of each individual. We do not have so many features in our dataset except the three axes of accelerometer and gyroscope. That's why we did not need to apply reduction techniques in our dataset rather we have modeled those data using the same machine learning classifiers and neural network algorithms.

Chapter 2

Literature Review

From the paper ,the research team introduced a new publicly data set For Human Activity Recognition using smartphones and acknowledged some results using a multi class Support Vector Machine approach. They also improved the classification performance of the learned model using the data set[29].
In this paper, performance of classification is observed using three machine learning algorithms which are SVM, HMM and ANN. Body activity recognition are obtained using numerical data collected from wearable sensors as well as the classification algorithms. Apart from activity motions other features like age, weight, acceleration statistics, physiological measurements are also taken into consideration. In future, they intend to make SVM approach for finding optimal parameters more efficient, decreasing the consumed time and computational complexity. They also plan to make HMM approach more convenient in producing accurate results. Lastly, they will try to recognize body activities from the perspective of sparse representation and random projections[30].
In this paper, human activities are recognised using LSTM network by evaluating four fusion methods for combining convolutional neural network outputs. Performance improvement is obtained by adding a third LSTM layer. Final illustrated results represents an attention mechanism to direct the LSTM through the noticeable fields of convolutional feature sequence[31].
In the paper, the research team developed a complete system included data acquisition system, features extraction, data processing, training and human activity recognition . Support vector machine is used to classify and identify action and recognised the system on windows , android platforms and operate in real time[32].
In the paper a database of more than 100 subjects was analyzed regarding human activity recognition using smartphone sensors. They obtained 98% accuracy through evaluating performance in terms of confusion matrices, effects of time-window sizes and different feature vectors. In future they plan to detect other complex activities to observe how the recognition rates scale for such sequences. Moreover, they expect to utilize their results for automatic social networking feeds, health related issues, calculating daily calories along with entertainment, sports, daily tasks or occupational tasks[33].
In the paper, it is analyzed that activities can be detected using a single triaxial accelerometer providing high accuracy. Meta-classifier usage is feasible to detect activities related to hands or mouth which are hard to detect using single accelerometer. They found that plurality voting classifier is the best classifier for activity

recognition. They would like to extend their research recognizing short activities such as opening door using swipe card from accelerometer data. Also there motto is to study the effect of ontology of activities in terms of classifying activities which are hard to recognize[34].
In this research, smartphone sensors including accelerometer and gyroscope are validated for activity recognition. Gyroscope provide extra benefits here as a data source by classifying activities. Heart rate monitor sensor has improved the intensity of identifying the activities. This study discusses the advantages and disadvantages of using a smartphone as a data collector in human activity recognition. It ensures effective ecological validity as it requires people to carry only one device that they carry usually[35].
In the paper , the research group introduced a comprehensive survey of the recent advances in activity recognition with smartphone sensors. They review the core data mining techniques behind the mainstream activity recognition algorithms, analyse their major challenges and introduced a variety of real applications enabled by the activity recognition[36].
In this research, smartphones incorporated with triaxial accelerometers are used to recognize human activities in a desirable way. Smartphones are easy to carry in many ways from which same activity can have variation in acceleration data. The system implements kernel discriminant analysis for balancing the class variances. The technique is implemented using the activity data collected from five body positions using a smartphone. The system increases the applicability of activity classification systems. By using an accelerometer enabled smartphone,which could be placed in any pocket without firm attachment to a specific body part, activities could be monitored throughout a longer period of time[37].
In this paper, they focused on healthcare applications and introduced MC-HF-SVM approach to use in AmI system for this purpose. Also they have used integer parameters to build multiclass SVM. Moreover, they assume MC-HF-SVM approach involving fixed-point calculations maybe used AR that ensures less memory,time and power consumption and also AR gives compatible accuracy levels to MC-SVM approach. Their experiment dictates that standard MC-SVM can be substituted from learned MC-HF-SVM model with reduction of 6 bits. This result has positive impacts on smartphones as it helps in releasing system and reduces energy consumption. In future they are hoping to introduce a public AR dataset to enable other researchers to test and compare among various learning models[38].
This paper proposes a modern and reliable method for ADL inference involving human motion and object recognition. The system uses 3 accelerometer sensor fusion and RFID reader. Among those two accelerometers are used to classify five human states using decision tree and detected hand moving RFID marked objects includes instrumental object-related activities. This centralized approach to instrumental operations in the identification of the human body system offers a reliable and high accuracy recognition level. This system is compact and smaller but provides recognition rate of 95% for 18 ADL. The program tests whether the elderly can live normally or not after wearing small mobile gadgets like accelerometers and RFID detector, and then the results are sent to their remote relatives. In future they are planning to check the health status of users by calculating calorie intake and expending calorie[39].
This paper represents Human Activity Recognition(HAR) area based on smart-

phones with inertial sensors. They discussed the history of smartphone based human activity recognition area involving the gradual evolution of HAR. The motto of exploring the noticeable historical events of HAR is to stimulate the future plans regarding HAR bringing revolutionary turning points of this area. Also, they described detailed dataset along with the necessary features for modeling the data. According to them feature extraction can be done in two ways; either manually or automatically. They have used shallow and deep machine learning algorithms for classification of data. Moreover, they have discussed the areas of data fusion, energy efficiency and data dimensionality reduction techniques and also highlighted the major findings of the best fit methods. Furthermore, they mentioned several challenges and future plans regarding smartphone based HAR area. Finally, they have provided a set of real time HAR applications which will definitely help in recognizing physical activities of humans[28].

In this paper, realtime human activity is monitored and thus predicted using multiple sensors. They analyzed several sets of features and sampling rates for finding a convenient classifier and also showed how that model performed on different body parts which are used in wearable device for collecting data. In future they will be extending their activity classifier for other activities to recognize individual's location using the activity classification[40].

This paper introduces eWatch which is a wearable computing platform existing in the form of a wrist watch, has sensing and notifying capability. It is easily viewable, traced and available for the sensors. It can sense light, motion, audio, temperature via bluetooth communication and also provides visual, audio and tactile information. This dynamic wearable computing platform provides realistic user observations. It consists of CPU, sensors, power control, notification mechanisms, and wireless communication. In addition,on the built-in display, it has the eWatch shell and the Graphical User Interface (GUI) . The sensed data by this eWatch is less power consuming and make efficient use of memory. The platform is suitable for classifying data using machine learning algorithms where sufficient battery capacity is ensured. In Future they will focus on the recognition of activity and the combination of location information with activity data. To enable the eWatch to function as a mobile node in a sensor network they want to integrate an 802.15.4 radio. Besides adding flexibility, integration of the eWatch into its environment will be made possible covering a wider network. It will surely bring a robust evolution in Human Activity Recognition (HAR) area[41].

In this paper, Ambient Assisted Living (AAL) is being emphasized for fall detection of elderly person which is a critical problem to focus on. They present an Activity Recognition (AR) and Fall Detection (FD) system to record real time observation using two accelerometers making it a mature technology for this purpose. For AR system, an architecture containing a set of combined rules is built to recognize postures ensuring that the system is well behaved. The classifiers of machine learning provides maximum accuracies wherever the rules of the architecture fail. In addition, for FD system emphasis is put on high accelerations and detected horizontal orientation as falling incorporates lying too. The F-measure of AR was 99% being tested on different persons and the F-measure of FD was 78% due to failure of recognizing events that lack of real time performance. So the system works well with postures or static activities but rules needed to recognize dynamic activities are quite difficult. Another issue is considering long term lying down recognized as

fall which might not always true. In future they are planning to sewing the sensors into users' clothing by ensuring that it does not decrease the accuracy significantly. As FD accuracy is not upto the mark due to the use of only two accelerometers they want to use additional sensors and recognize the field of potential fall and thus providing better health monitoring system of the elderly citizens[42].

Chapter 3

Dataset

In our research, we have used two sets of data sets. One is existing data set which we collected from [29]. Another one is new data set which we developed by collecting data through a waist mounted device. For our existing dataset and new dataset the workflow diagram is given below where we have mentioned the individual work process.

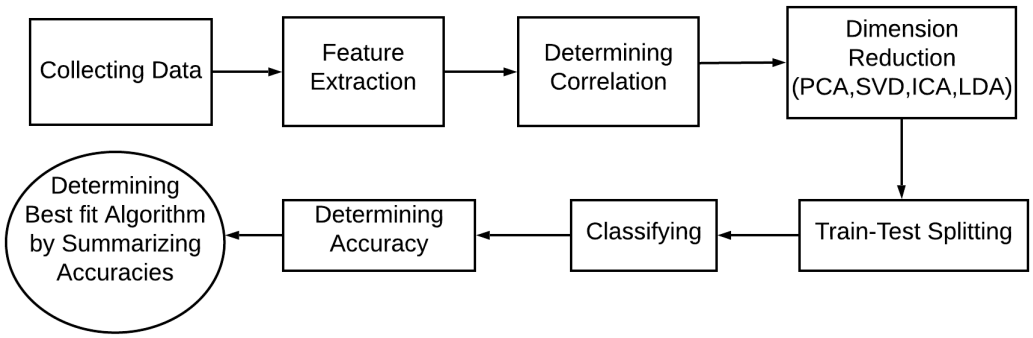


Figure 3.1: Workflow of existing dataset

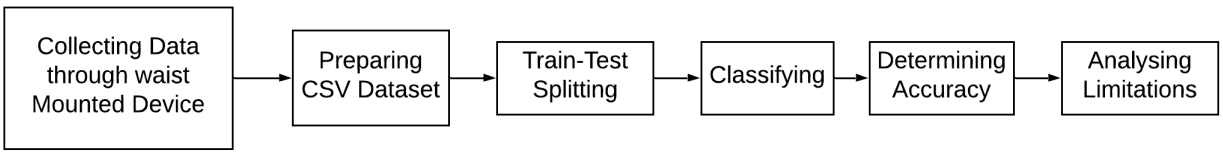


Figure 3.2: Workflow of new dataset

Here in Figure 3.1 shows that, after collecting ta dataset from UCI we extract the features of dataset as there is a huge number of features. After that we use multiple dimension reduction techniques and split the dataset. Then we use machine learning and neural network algorithms and find out the best fit algorithm for human activity prediction. In figure 3.2, the workflow diagram shows that, at first we collect the dataset with waist mounted device for six human activity. After getting the gyroscope and accelerometer data for per activity we arranged that in CSV file and classify that dataset and find out the accuracies for per classifier.
Now here we will describe the elaborate description of both dataset-

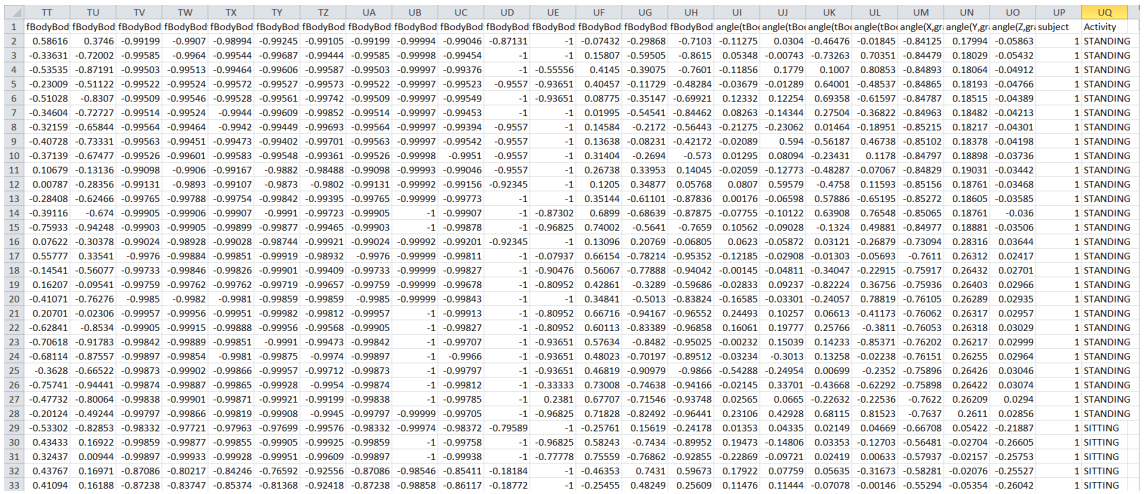
3.1 Existing Dataset

3.1.1 Dataset Description

We get the publicly data set for human activity recognition using smartphones from UCI Machine Learning Repository. In this data set, they use waist mounted device having smartphone sensors: gyroscope and accelerometer value to determine multiple activity of people. They focus on six activities of human like lying, sitting, standing, walking, walking upstairs and walking downstairs. Among them three of them like standing, sitting, lying are static activities and walking, walking downstairs , walking upstairs are dynamic activities. They have also included postural transitions that occurred between the static activities which are stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, and lie-to-stand . in this dataset, they collected data of 30 people aged between 19 to 48 in a rich sensor responsive environment using 72 environment and body sensors. Each person performed six activities wearing a smartphone (Samsung Galaxy S II). This dataset has 7353 rows and 563 columns. There are 561 features bearing the frequency and time domain variable where the calculated triaxial acceleration from the accelerometer (total acceleration) and the estimated body acceleration and also triaxial Angular velocity from the gyroscope are given through signal processing. The other two features an identifier of the subject whom carried out the experiment and the activity level of six activities for each subject. shown in Publicly available datasets provide a freely available source of data across various disciplines and researchers in the eld. That's why, they present a new dataset that has been created using inertial data from smartphone accelerometers and gyroscopes, targeting the recognition of six different human activities. Therefore, they have obtained results by exploiting a multi class Support Vector Machine (SVM) classier.

3.1.2 Dataset Preprocessing

The performance of all six activities took total 192 seconds for each person . They have started with standing position from zero second and then simultaneously took static activities value in order to standing, sitting, laying where each static activities took 15 seconds. After that they took the value of dynamic activities(walking, walking downstairs , walking upstairs) for 12 seconds each. There is also a separation of 5 seconds between each activities. They used an integrated handheld accelerometer and gyroscope sensor to monitor 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz. These signals are pre-processed with a median



	TT	TU	TV	TW	TX	TY	TZ	UA	UB	UC	UD	UE	UF	UG	UH	UI	UJ	UK	UL	UM	UN	UO	UP	UQ
1	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	fBodyBod	angle(tBo	angle(tBo	angle(tBo	angle(tBo	angle(X,gr	angle(Y,gr	angle(Z,gr	subject	Activity
2	0.58616	0.3746	-0.99199	-0.9907	-0.98994	-0.99245	-0.99105	-0.99199	-0.99994	-0.99046	-0.87131	-1	-0.07432	-0.29868	-0.7103	-0.11275	0.0304	-0.46476	-0.01845	-0.84125	0.17994	-0.05863	1	STANDING
3	-0.33631	-0.72002	-0.99585	-0.9964	-0.99544	-0.99687	-0.99444	-0.99585	-0.99998	-0.99454	-1	-1	0.15807	-0.59505	-0.8615	0.05348	-0.00743	-0.73263	0.70351	-0.84479	0.18029	-0.05432	1	STANDING
4	-0.53535	-0.87191	-0.99503	-0.99513	-0.99464	-0.99606	-0.99587	-0.99503	-0.99997	-0.99376	-1	-0.55556	0.4145	-0.39075	-0.7601	-0.11856	0.1779	0.1007	0.80853	-0.84893	0.18064	-0.04912	1	STANDING
5	-0.23009	-0.51122	-0.99522	-0.99524	-0.99572	-0.99527	-0.99573	-0.99522	-0.99997	-0.99523	-0.9557	-0.93651	0.40457	-0.11729	-0.48284	-0.03679	-0.01289	0.64001	-0.48537	-0.84865	0.18193	-0.04766	1	STANDING
6	-0.51028	-0.8307	-0.99509	-0.99546	-0.99528	-0.99561	-0.99742	-0.99509	-0.99997	-0.99549	-1	-0.93651	0.08775	-0.35147	-0.69921	0.12332	0.12254	0.69358	-0.61597	-0.84787	0.18515	-0.04389	1	STANDING
7	-0.34604	-0.72727	-0.99514	-0.99524	-0.9944	-0.99609	-0.99852	-0.99514	-0.99997	-0.99453	-1	-1	0.01995	-0.54541	-0.84462	0.08263	-0.14344	0.27504	-0.36822	-0.84963	0.18482	-0.04213	1	STANDING
8	-0.32159	-0.65844	-0.99564	-0.99464	-0.9942	-0.99449	-0.99693	-0.99564	-0.99997	-0.99394	-0.9557	-1	0.14584	-0.2172	-0.56443	-0.21275	-0.23062	0.01464	-0.18951	-0.85215	0.18217	-0.04301	1	STANDING
9	-0.40728	-0.73331	-0.99563	-0.99451	-0.99473	-0.99402	-0.99701	-0.99563	-0.99997	-0.99542	-0.9557	-1	0.13638	-0.08231	-0.42172	-0.02089	0.594	-0.56187	0.46738	-0.85102	0.18378	-0.04198	1	STANDING
10	-0.37139	-0.67477	-0.99526	-0.99601	-0.99583	-0.99548	-0.99361	-0.99526	-0.99998	-0.9951	-0.9557	-1	0.31404	-0.2694	-0.573	0.01295	0.08094	-0.23431	0.1178	-0.84797	0.18898	-0.03736	1	STANDING
11	0.10679	-0.13136	-0.99098	-0.9906	-0.99167	-0.9882	-0.98488	-0.99098	-0.99993	-0.99046	-0.9557	-1	0.26738	0.33953	0.14045	-0.02059	-0.12773	-0.48287	-0.07067	-0.84829	0.19031	-0.03442	1	STANDING
12	0.00787	-0.28356	-0.99131	-0.9893	-0.99107	-0.9873	-0.9802	-0.99131	-0.99992	-0.99156	-0.92345	-1	0.1205	0.34877	0.05768	0.0807	0.59579	-0.4758	0.11593	-0.85156	0.18761	-0.03468	1	STANDING
13	-0.28408	-0.62466	-0.99765	-0.99788	-0.99754	-0.99842	-0.99395	-0.99765	-0.99999	-0.99773	-1	-1	0.35144	-0.61101	-0.87836	0.00176	-0.06598	0.57886	-0.65195	-0.85272	0.18605	-0.03585	1	STANDING
14	-0.39116	-0.674	-0.99905	-0.99906	-0.99907	-0.9991	-0.99723	-0.99905	-1	-0.99907	-1	-0.87302	0.6899	-0.68639	-0.87875	-0.07755	-0.10122	0.63908	0.76548	-0.85065	0.18761	-0.036	1	STANDING
15	-0.75933	-0.94248	-0.99903	-0.99905	-0.99899	-0.99877	-0.99465	-0.99903	-1	-0.99878	-1	-0.96825	0.74002	-0.5641	-0.7659	0.10562	-0.09028	-0.1324	0.49881	-0.84977	0.18881	-0.03506	1	STANDING
16	0.07622	-0.30378	-0.99024	-0.98928	-0.99028	-0.98744	-0.99921	-0.99024	-0.99992	-0.99201	-0.92345	-1	0.13096	0.20769	-0.06805	0.0623	-0.05872	0.03121	-0.26879	-0.73094	0.28316	0.03644	1	STANDING
17	0.55777	0.33541	-0.9976	-0.99884	-0.99851	-0.99919	-0.98932	-0.9976	-0.99999	-0.99811	-1	-0.07937	0.66154	-0.78214	-0.95352	-0.12185	-0.02908	-0.01303	-0.05693	-0.7611	0.26312	0.02417	1	STANDING
18	-0.14541	-0.56077	-0.99733	-0.99846	-0.99826	-0.99901	-0.99409	-0.99733	-0.99999	-0.99827	-1	-0.90476	0.56067	-0.77888	-0.94042	-0.00145	-0.04811	-0.34047	-0.22915	-0.75917	0.26432	0.02701	1	STANDING
19	0.16207	-0.09541	-0.99759	-0.99762	-0.99762	-0.99719	-0.99657	-0.99759	-0.99999	-0.99678	-1	-0.80952	0.42861	-0.3289	-0.59686	-0.02833	0.09237	-0.82224	0.36756	-0.75936	0.26403	0.02966	1	STANDING
20	-0.41071	-0.76276	-0.9985	-0.9982	-0.9981	-0.99859	-0.99859	-0.9985	-0.99999	-0.99843	-1	-1	0.34841	-0.5013	-0.83824	-0.16585	-0.03301	-0.24057	0.78819	-0.76105	0.26289	0.02935	1	STANDING
21	0.20701	-0.02306	-0.99957	-0.99956	-0.99951	-0.99982	-0.99812	-0.99957	-1	-0.99913	-1	-0.80952	0.66716	-0.94167	-0.96552	0.24493	0.10257	0.06613	-0.41173	-0.76062	0.26317	0.02957	1	STANDING
22	-0.62841	-0.8534	-0.99905	-0.99915	-0.99888	-0.99956	-0.99568	-0.99905	-1	-0.99827	-1	-0.80952	0.60113	-0.83389	-0.96858	0.16061	0.19777	0.25766	-0.3811	-0.76053	0.26318	0.03029	1	STANDING
23	-0.70618	-0.91783	-0.99842	-0.99889	-0.99851	-0.9991	-0.99473	-0.99842	-1	-0.99707	-1	-0.93651	0.57634	-0.8482	-0.95025	-0.00232	0.15039	0.14233	-0.85371	-0.76202	0.26217	0.02999	1	STANDING
24	-0.68114	-0.87557	-0.99897	-0.99854	-0.9981	-0.99875	-0.9974	-0.99897	-1	-0.9966	-1	-0.93651	0.48023	-0.70197	-0.89512	-0.03234	-0.3013	0.13258	-0.02238	-0.76151	0.26255	0.02964	1	STANDING
25	-0.3628	-0.66522	-0.99873	-0.99902	-0.99866	-0.99957	-0.99712	-0.99873	-1	-0.99797	-1	-0.93651	0.46819	-0.90979	-0.9866	-0.54288	-0.24954	0.00699	-0.2352	-0.75896	0.26426	0.03046	1	STANDING
26	-0.75741	-0.94441	-0.99874	-0.99887	-0.99865	-0.99928	-0.9954	-0.99874	-1	-0.99812	-1	-0.33333	0.73008	-0.74638	-0.94166	-0.02145	0.33701	-0.43668	-0.62292	-0.75898	0.26422	0.03074	1	STANDING
27	-0.47732	-0.80064	-0.99838	-0.99901	-0.99871	-0.99921	-0.99199	-0.99838	-1	-0.99785	-1	0.2381	0.67707	-0.71546	-0.93748	0.02565	0.0665	-0.22632	-0.22536	-0.7622	0.26209	0.0294	1	STANDING
28	-0.20124	-0.49244	-0.99797	-0.99866	-0.99819	-0.99908	-0.9945	-0.99797	-0.99999	-0.99705	-1	-0.96825	0.71828	-0.82492	-0.96441	0.23106	0.42928	0.68115	0.81523	-0.7637	0.2611	0.02856	1	STANDING
29	-0.53302	-0.82853	-0.98332	-0.97721	-0.97963	-0.97699	-0.99576	-0.98332	-0.99974	-0.98372	-0.79589	-1	-0.25761	0.15619	-0.24178	0.01353	0.04335	0.02149	0.04669	-0.66708	0.05422	-0.21887	1	SITTING
30	0.43433	0.16922	-0.99859	-0.99877	-0.99855	-0.99905	-0.99925	-0.99859	-1	-0.99758	-1	-0.96825	0.58243	-0.7434	-0.89952	0.19473	-0.14806	0.03353	-0.12703	-0.56481	-0.02704	-0.26605	1	SITTING
31	0.32437	0.00944	-0.99897	-0.99933	-0.99928	-0.99951	-0.99609	-0.99897	-1	-0.99938	-1	-0.77778	0.75559	-0.76862	-0.92855	-0.22869	-0.09721	0.02419	0.00633	-0.57937	-0.02157	-0.25753	1	SITTING
32	0.43767	0.16971	-0.87086	-0.80217	-0.84246	-0.76592	-0.92556	-0.87086	-0.98546	-0.85411	-0.18184	-1	-0.46353	0.7431	0.59673	0.17922	0.07759	0.05635	-0.31673	-0.58281	-0.02076	-0.25527	1	SITTING
33	0.41094	0.16188	-0.87238	-0.83747	-0.85374	-0.81368	-0.92418	-0.87238	-0.98858	-0.86117	-0.18772	-1	-0.25455	0.48249	0.25609	0.11476	0.11444	-0.07078	-0.00146	-0.55294	-0.05354	-0.26042	1	SITTING

Figure 3.3: CSV file of existing dataset

filter and a 3rd order low-pass Butterworth filter with a cutoff frequency of 20Hz. This rate is sufficient to capture the movement of the human body as 99% of its energy is[43]. Using butterworth low-pass filter into body acceleration and inertia, the acceleration signal, which has gravitational and body movement components, was isolated. The gravitational force is believed to have only low frequency components, so they found the experiments that for a constant gravity signal, 0.3 Hz was an ideal corner frequency. Additional time signals were obtained when the euclidean magnitude and time derivatives (jerk and angular acceleration) were calculated from the triaxial signals. From the smartphone sensor, 17 measures (like mean,max, std, mad, entropy and so on) applied to the time and frequency domain signals and a total of 561 features were extracted to describe each activity window. Table 3.1 shows the time and frequency domain signals obtained from the smartphone sensors.

Namr	Time	Frequency
Body Acc	1	1
Gravity Acc	1	0
Body Acc Jerk	1	1
Body Angular Speed	1	1
Body Angular Acc	1	0
Body Acc Magnitude	1	1
Gravity Acc Mag	1	0
Body Acc Jerk Mag	1	1
Body Angular Speed Mag	1	1
Body Angular Acc Mag	1	1

Table 3.1: Time and frequency domain signals obtained from the smartphone sensors

After processing of data the dataset has also been randomly partitioned into two independent sets, where 70% of data were selected for training and the remaining 30% for testing by using train-test splitting method. After splitting the dataset they have 7352 rows and 561 columns in training set and 2947 rows and 561 columns in testing set. X_train, X_test, Y_train Y_test is determined with the splitting of data where X denotes the all 561 features and Y denotes the activity and subject.

3.1.3 Dimension Reduction Techniques

As the dataset has a big number of features and it is a higher dimensional data so that dimension reduction techniques is highly recommended for this dataset. We have used 4 dimension reduction techniques by using different size of components. These techniques help us to get better results for each algorithm.

PCA

Principal component analysis (PCA) extracts a set of features from a large data set. By comparing the dataset we used 40 components of data and apply PCA on it. After comparing the result we find that, the maximization of components helps us to get higher accuracy. So we increase the number of component to 150. After applying PCA we have split the dataset by 70% of training and 30% of testing dataset where we get 150 features. The distribution of PCA explained variance for 150 components is given below. Here figure 3.4 shows the distribution plot of PCA explained variance where x and y axis shows the value of decomposed value. This plot denotes 3 components scatter plot from o to 2.

```
array([3.48236304e+01, 2.73504627e+00, 2.29439284e+00, 1.04377529e+00,
       9.43517003e-01, 7.08152304e-01, 6.55052596e-01, 5.95090075e-01,
       5.39647116e-01, 4.77652868e-01, 4.24368278e-01, 3.74345916e-01,
       3.22558940e-01, 3.10568349e-01, 2.77748554e-01, 2.64394540e-01,
       2.60087117e-01, 2.40278721e-01, 2.36992739e-01, 2.28464121e-01,
       2.19338452e-01, 2.08932138e-01, 1.96193981e-01, 1.88919079e-01,
       1.84899516e-01, 1.77823828e-01, 1.67998396e-01, 1.62433371e-01,
       1.61201621e-01, 1.56947282e-01, 1.54039313e-01, 1.46630230e-01,
       1.42127492e-01, 1.31304929e-01, 1.28402565e-01, 1.24503411e-01,
       1.24113706e-01, 1.17059207e-01, 1.16010916e-01, 1.13996152e-01,
       1.07361875e-01, 1.05147999e-01, 1.00861815e-01, 9.99112269e-02,
       9.59336403e-02, 8.91833276e-02, 8.74228171e-02, 8.71933876e-02,
       8.51572724e-02, 8.36995317e-02, 8.07860434e-02, 7.82934139e-02,
       7.61695833e-02, 7.06209289e-02, 6.95900521e-02, 6.66596553e-02,
       6.57564908e-02, 6.40168717e-02, 6.29522510e-02, 6.10522902e-02,
       5.99581212e-02, 5.83972646e-02, 5.78371461e-02, 5.64901049e-02,
       5.51970702e-02, 5.40131564e-02, 5.31140500e-02, 5.09165244e-02,
       4.94158593e-02, 4.87665414e-02, 4.74709958e-02, 4.61522143e-02,
       4.57467554e-02, 4.48880901e-02, 4.35300854e-02, 4.33772847e-02,
       4.29765626e-02, 4.15378210e-02, 4.07378767e-02, 4.02221191e-02,
       3.92285987e-02, 3.84024922e-02, 3.76875022e-02, 3.68841945e-02,
       3.63282590e-02, 3.61617524e-02, 3.54561174e-02, 3.50899524e-02,
       3.41889427e-02, 3.35945992e-02, 3.34256729e-02, 3.27123550e-02,
       3.17095382e-02, 3.09556493e-02, 3.05520891e-02, 3.02911609e-02,
       2.99032552e-02, 2.93217664e-02, 2.92342624e-02, 2.85826969e-02,
       2.78376089e-02, 2.73058203e-02, 2.71579950e-02, 2.65568021e-02,
       2.60744014e-02, 2.56990547e-02, 2.51498922e-02, 2.48692632e-02,
       2.45277287e-02, 2.41807886e-02, 2.35430392e-02, 2.33561865e-02,
       2.18266903e-02, 2.15983688e-02, 2.11993442e-02, 2.07079667e-02,
       2.05003451e-02, 2.01082922e-02, 1.94520578e-02, 1.89246911e-02,
       1.86604505e-02, 1.85633107e-02, 1.80991811e-02, 1.76258787e-02,
       1.71725865e-02, 1.69223476e-02, 1.69062769e-02, 1.65383852e-02,
       1.59523315e-02, 1.56400927e-02, 1.54626788e-02, 1.49224192e-02,
       1.45200323e-02, 1.43523304e-02, 1.39519677e-02, 1.35970593e-02,
       1.35312343e-02, 1.33478126e-02, 1.31833230e-02, 1.30524372e-02,
       1.27313040e-02, 1.23539130e-02, 1.22295771e-02, 1.17948478e-02,
       1.16631280e-02, 1.14810063e-02, 1.12128852e-02, 1.11470565e-02,
       1.09350208e-02, 1.06615664e-02])
```

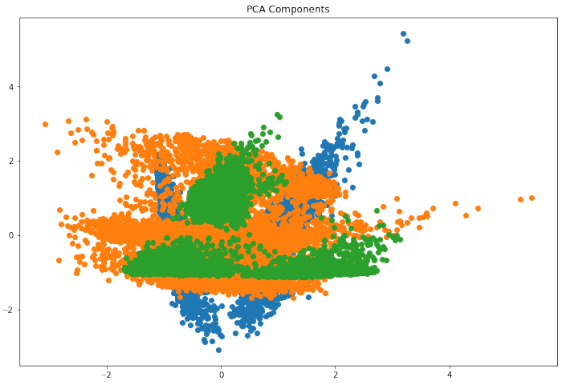


Figure 3.4: Distribution of PCA components

SVD

Singular Value Decomposition is used to reduce the dimensionality like the number of columns, of a data set. We have used 100 components to reduce the extra features. The explained variances of SVD is given below. Figure 3.5 shows the decomposed value of 100 components of SVD where the decomposed components are indicated very clearly by the scatter plot and there is not noticeable correlation among the components. This plot denotes 3 components scatter plot from o to 2.

```
array([15.81435622, 19.89542722,  2.46440779,  2.074266  ,  0.96149788,
       0.83862051,  0.6757628 ,  0.64728367,  0.55605685,  0.53488731,
       0.47352632,  0.41316063,  0.3485229 ,  0.32193186,  0.30815253,
       0.27628144,  0.26285824,  0.24525423,  0.23931494,  0.2368935 ,
       0.22819231,  0.21856582,  0.20822338,  0.19542522,  0.18494218,
       0.1822536 ,  0.1746681 ,  0.1679705 ,  0.16212413,  0.16017619,
       0.15683081,  0.15379591,  0.14518502,  0.14208205,  0.13026747,
       0.12806285,  0.12419342,  0.12341213,  0.11702718,  0.11550953,
       0.11285474,  0.10664554,  0.10443699,  0.10074032,  0.09975222,
       0.0908902 ,  0.08905822,  0.08734564,  0.0871333 ,  0.08477161,
       0.08264854,  0.07858029,  0.07646582,  0.07078369,  0.06999507,
       0.06748912,  0.0660801 ,  0.06404907,  0.06296528,  0.06165746,
       0.06097722,  0.05942244,  0.05811484,  0.05753156,  0.05618978,
       0.05500732,  0.0540023 ,  0.05170941,  0.05081611,  0.04932823,
       0.04858468,  0.04726333,  0.04600205,  0.044998  ,  0.04415053,
       0.04342728,  0.04292128,  0.04270415,  0.04133279,  0.04064115,
       0.04003368,  0.03917396,  0.03833682,  0.03743294,  0.03667586,
       0.03588283,  0.03570817,  0.03509513,  0.03476603,  0.03386199,
       0.03336774,  0.03289277,  0.03219746,  0.0308713 ,  0.03064032,
       0.0301606 ,  0.02947134,  0.02904779,  0.02883708,  0.0283806 ])
```

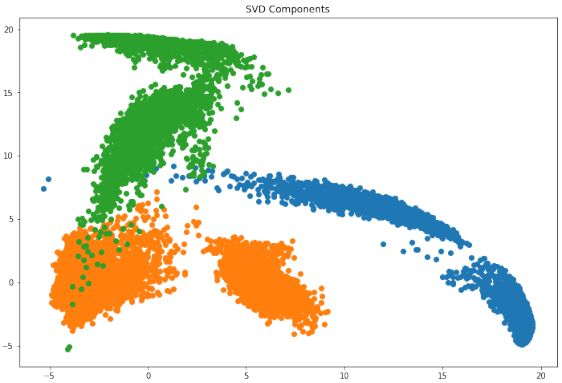


Figure 3.5: Distribution of SVD components

LDA

Linear discriminant analysis computes the directions of linear discriminants that will represent the axes. It maximizes the separation between multiple classes. As we have multiple classes of features that's why we use 40 components of features to calculate LDA. After applying the LDA on X_train and X_test we get 5881 rows and 5 columns in the training set and 1471 rows and 5 columns in the testing set where LDA reduces a high number features and the new feature calculates the higher accuracy value for each algorithm. Figure 3.6 shows the decomposed values of x and y axis where it shows that, LDA maximizes the separation between classes and the plot denotes 3 components scatter plot from o to 2.

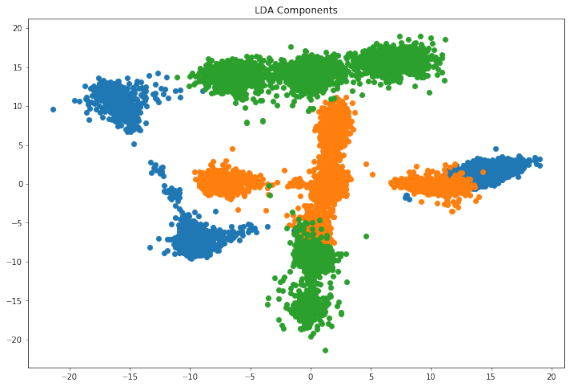


Figure 3.6: Distribution of LDA components

ICA

Independent Component Analysis finds the independent variables and here we use Non-Gaussianity method to measure independent components. We have used both 40 and 150 components to determine the independent variables in X_test and X_train. After applying ICA of 150 components we got 150 features in both test and train set. Here the value of decomposed independent components are shown in the figure 3.7 by two axis x and y. This plot denotes 3 components scatter plot from o to 2.

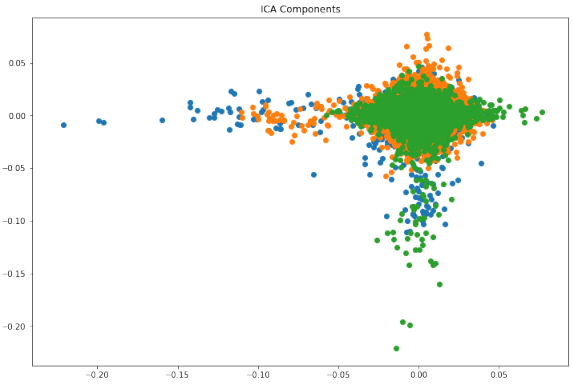


Figure 3.7: Distribution of ICA components

3.1.4 Data Visualization

In order to work with any dataset it is very important to visualize it, visualization of graphs helps us to understand the dataset and the correlation between data more precisely . With the help of Python librarieswe can easily visualize the data. We used python libraries like seaborn , mathplotlib , scikitplot etc to plot heatmap , correlation matrix , scatter and density , countplot , pieplot etc. Here, the visualization plots of existing data is given below:

Scatter and Density Plot

Here the density plot visualises the distribution of data over time period and continuous interval where the time interval of existing data is 5 second. Figure 3.8 also shows the scatter plot to denote how much one variable is affected by another.

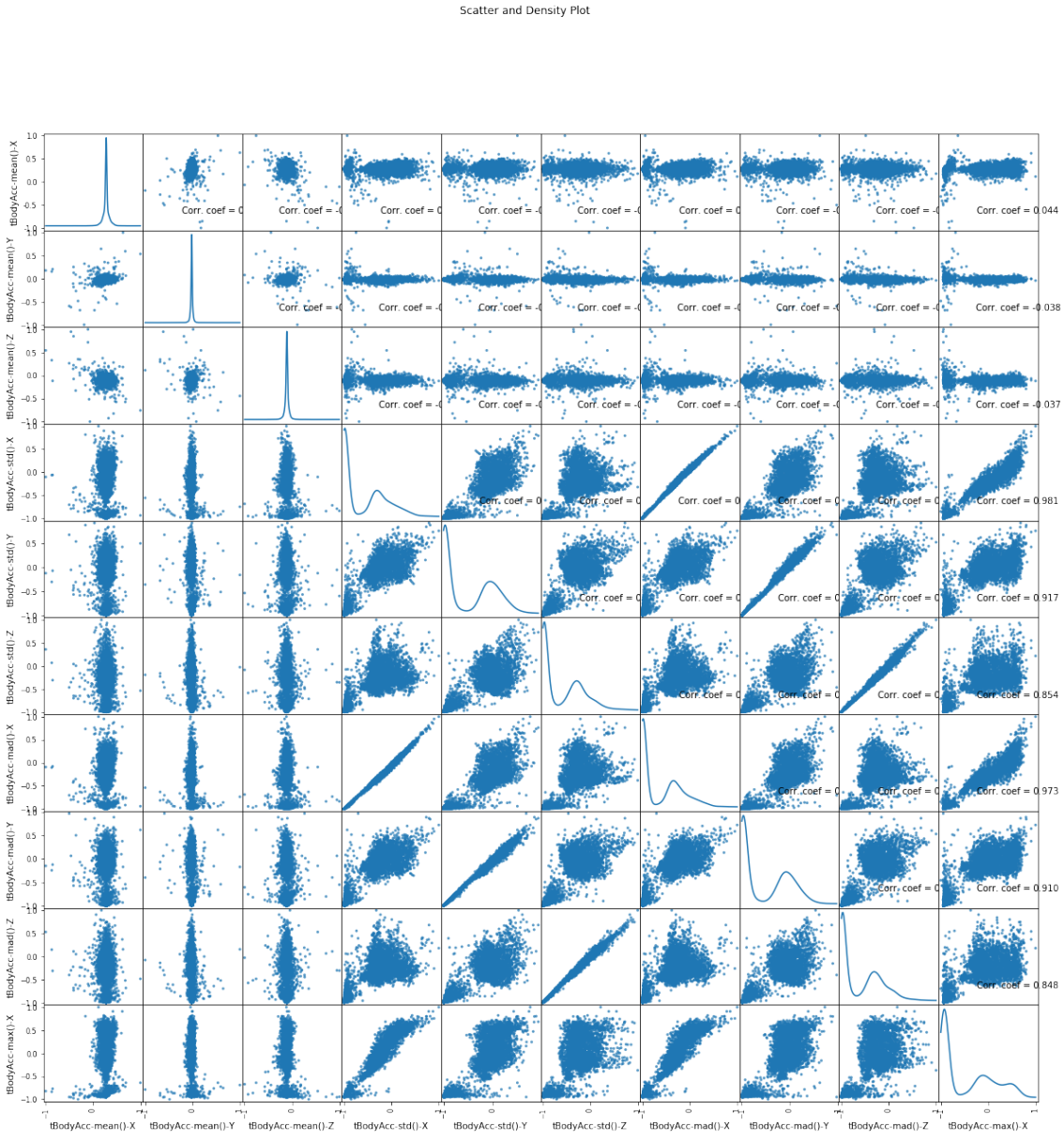


Figure 3.8: Scatter and Density Plot of existing dataset

Histogram plot

Histogram diagram consisting of rectangles whose region is proportional to a variable's frequency and whose size is equivalent to the interval of class. Here in existing dataset, the range of activity is 0.00-0.40 which is shown in figure 3.9.

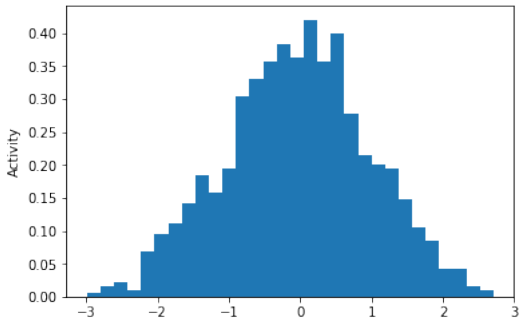


Figure 3.9: Histogram of existing dataset

Pie Plot

Pie plot is one of the most commonly used form of data visualization, it is used to plot the pie chart. In the pie chart the percentage of data of each category is represented as a slice of the circle. Here figure 3.10 shows the percentage of six activities in a pie chart form of existing dataset.

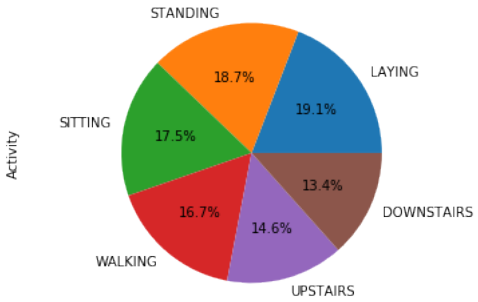


Figure 3.10: Pie chart of existing dataset

Count Plot

Countplot is a barplot where the dependent variable is the number of instances of each instance of the independent variables like human activities. We used seaborn visualization libraries count plot to visualize our data. Figure 3.11 shows each activity count range in a barplot form of existing dataset.

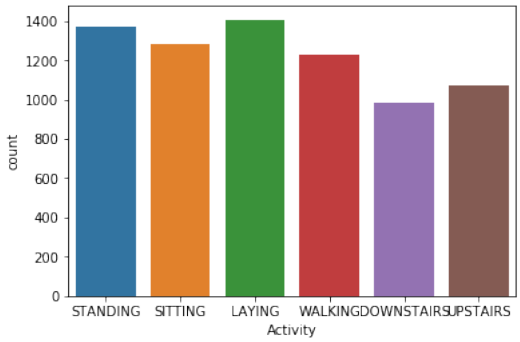


Figure 3.11: Count plot of existing dataset

Heatmap Plot

Heatmap is the graphical representation of each column and their relation between them. It is in reality a colored tabular matrix where each cell's color depends on the information this contains. We used python seaborn library to plot the heatmap. Here existing dataset has 563 column and the graphical representation of each column is shown in figure 3.12 below.

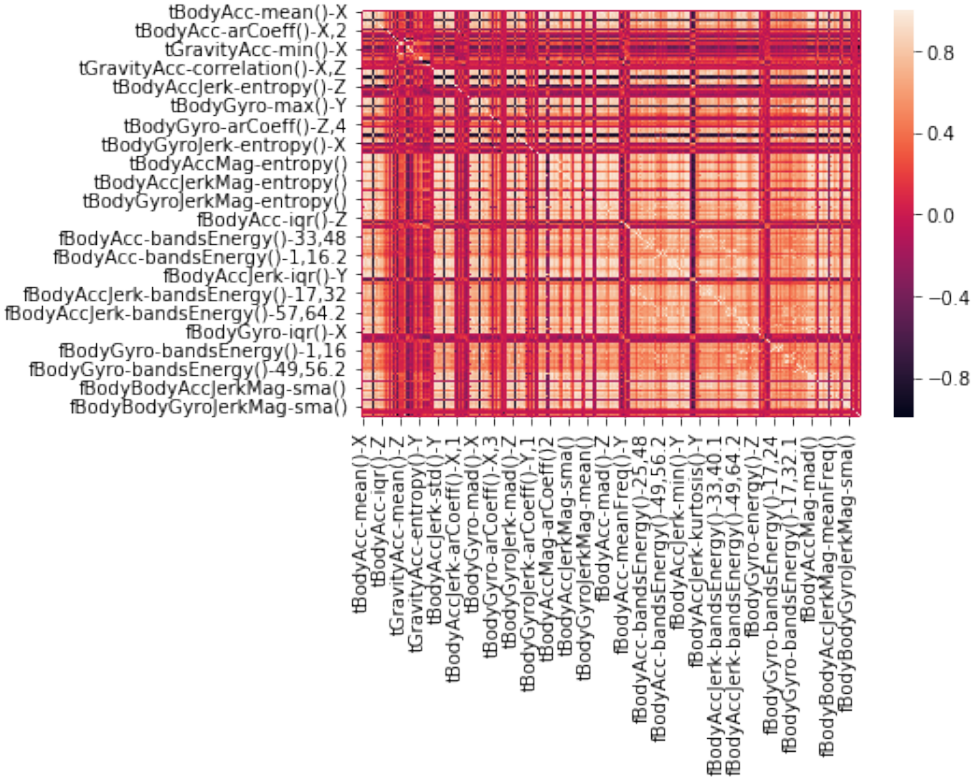


Figure 3.12: Heatmap plot of existing dataset

Correlation Matrix

Correlation matrix is a table of coefficients of correlation among variables. The relationship between two variables is shown by each cell in the table. As a way to summarize data, as an input into a more advanced analysis, and as a diagnosis for advanced analysis, a correlation matrix is used. We plotted the data correlation matrix by using python's matplotlib library to plot the matrix in the figure 3.13.

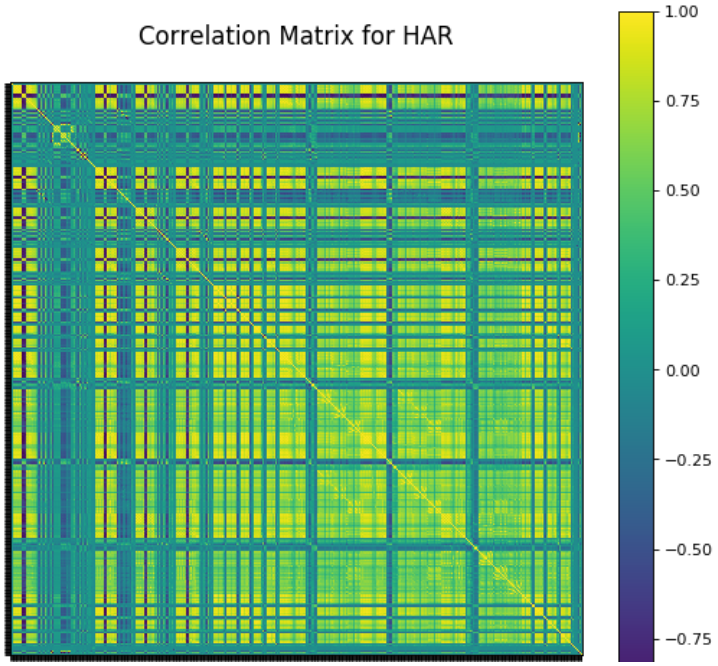


Figure 3.13: Correlation matrix of existing dataset

3.2 New Dataset

3.2.1 Device Description

We have developed a device for recording accelerometer and gyroscope value of human different activities movements. Here we implemented a waist mounted device because human bodies different movements value can be detected accurately thorough waist as it is the main masspoint of human body. To implement the device the main controller is arduino microcontroller. The main sensor of this device is 6DOF Accelerometer Gyroscope GY-521 which contains a MEMS accelerometer and a MEMS gyro in a single chip. We can capture 3 dimensional value accurately through this sensor as it contains 16-bits analog to digital conversion hardware for each channel. Therefore it captures the x, y, and z channel at the same time. To record the data, we have used Micro SD TF Card Module And DS1307 I2C Real Time Clock Module . Clock module is used for recording the time for each data. Table 3.2 shows the total time we have count for each person. The performances of all six activities took total 156 seconds for each person. We have started with lying position from zero second and then simultaneously took static activities value in order to standing, sitting, laying where each static activities took 20 seconds. Then, we took the value of dynamic activities like walking, walking downstairs, walking

upstairs for 25 seconds each. There is also a separation of 5 seconds between static activities and 2 seconds separation between dynamic activities. On the other hand, to record all those data we have used the SD card module. Here, we have used 16GB micro sd card in the module and the value of x,y,z of both accelerometer and gyroscope were stored in the card with the time and date while we were taking the data for each person. Figure 3.14 shows the device setup which we used for recording the value of human daily activities.

No	Activity	Activity Time(sec)	Pause time(sec)
1	Laying	20	5
2	Sitting	20	5
3	Standing	20	5
4	Walking	25	2
5	Walking Upstairs	25	2
6	Walking Downstairs	25	2
			Total= 156 sec

Table 3.2: Time of six activities of Human through Waist Mounted Device

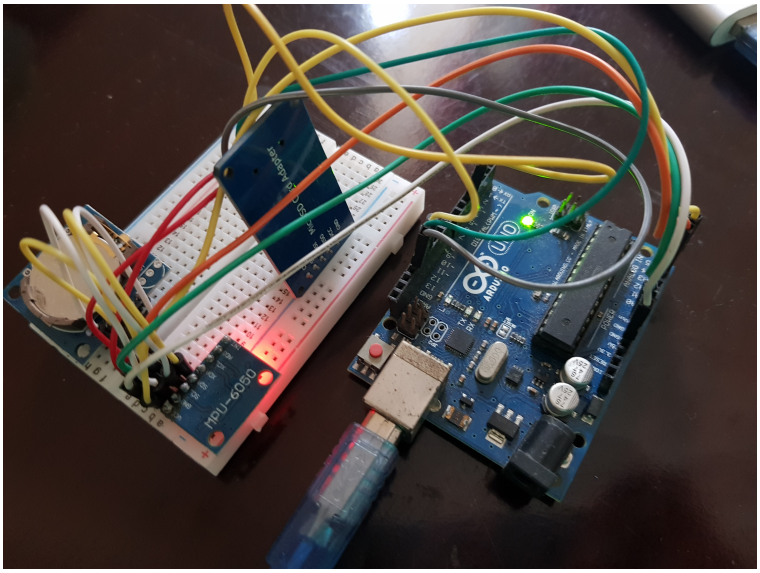


Figure 3.14: Device setup for data collection

3.2.2 Data Description

In our research, we have made a new dataset for human activity recognition using a wearable device developed with smartphone sensors: gyroscope and accelerometer. These sensors sense our targeted six human activities such as: Laying, Sitting, Standing, Walking, Walking downstairs and Walking upstairs. We have taken data of total 30 people aged between 55-65, 30-35 and 21-26 years. We collected data in a homely environment such as varsity cafeteria as well as house. After collecting the data through the waist mounted device, we placed the recorded value in CSV file where dataset has 2822 rows and 8 columns. Figure 3.15 shows the csv format of new dataset of 30 people with their six activities' value. The dataset has also been randomly partitioned into two independent sets, where 70% of data were selected for training and the remaining 30% for testing by using train-test splitting method. After splitting the dataset we have 2257 rows and 6 columns in training set and 565 rows and 6 columns in testing set. X_train, X_test, Y_train Y_test are determined with the splitting of data where X denotes 3 axis accelerometer (ax,ay,az) and 3 axis gyroscope(gx,gy,gz) value and Y denotes the activity and subject.

	A	B	C	D	E	F	G	H	I	J
1	acc_x	acc_y	acc_z	gyro_x	gyro_y	gyro_z	Subject	Activity		
2	-30180	4744	-28368	8	2072	202	1	LAYING		
3	-31472	3840	-28412	0	2048	80	1	LAYING		
4	-30924	4736	-28640	8	2072	217	1	LAYING		
5	-30956	4736	-28368	0	2064	2	1	LAYING		
6	-31228	4872	-28384	512	2064	16	1	LAYING		
7	3840	7816	-32696	520	2048	8	1	SITTING		
8	3900	7960	-32768	520	2049	8	1	SITTING		
9	3892	7864	-32512	520	2073	203	1	SITTING		
10	3840	7688	-32512	8	24	72	1	SITTING		
11	3888	7568	-32512	8	2064	152	1	SITTING		
12	552	6448	260	8	0	195	1	STANDING		
13	292	6536	0	8	1	82	1	STANDING		
14	280	6064	-28408	0	2048	90	1	STANDING		
15	288	6296	-28408	8	2064	129	1	STANDING		
16	48	6296	0	8	9	17	1	STANDING		
17	2848	8064	-28636	8	2064	66	1	WALKING		
18	3896	6704	256	520	2049	131	1	WALKING		
19	1584	5944	48	0	0	209	1	WALKING		
20	512	4408	16	520	25	137	1	WALKING		
21	3588	6680	-28672	512	16	25	1	WALKING		
22	276	6680	256	520	2073	17	1	WALKING		
23	1048	5768	48	512	8	18	1	WALKING		
24	1596	5256	-28392	520	17	202	1	WALKING_DOWNSTAIRS		
25	2576	5784	-28384	0	2056	146	1	WALKING_DOWNSTAIRS		
26	1548	5544	0	512	17	154	1	WALKING_DOWNSTAIRS		
27	3384	7832	0	8	16	192	1	WALKING_DOWNSTAIRS		
28	820	7864	-28608	0	24	203	1	WALKING_DOWNSTAIRS		
29	3340	1080	0	0	2064	3	1	WALKING_DOWNSTAIRS		
30	1812	4512	-28604	8	2064	152	1	WALKING_DOWNSTAIRS		
31	1284	7200	320	512	2056	128	1	WALKING_UPSTAIRS		
32	48	6552	336	512	9	209	1	WALKING_UPSTAIRS		
33	560	5280	260	0	24	138	1	WALKING_UPSTAIRS		
34	3352	144	0	0	2057	89	1	WALKING_UPSTAIRS		
35	1560	6328	340	512	25	75	1	WALKING_UPSTAIRS		
36	2356	6840	-28332	8	2057	27	1	WALKING_UPSTAIRS		

Figure 3.15: CSV file of new dataset

3.2.3 Data Visualization

Scatter and Density Plot

Here the figure 3.16 visualizes the time and continuous interval distribution of data where the time interval of new data is 5 seconds and 2 seconds. These statistics often include the scatter plot to show how much is affected by another variable.

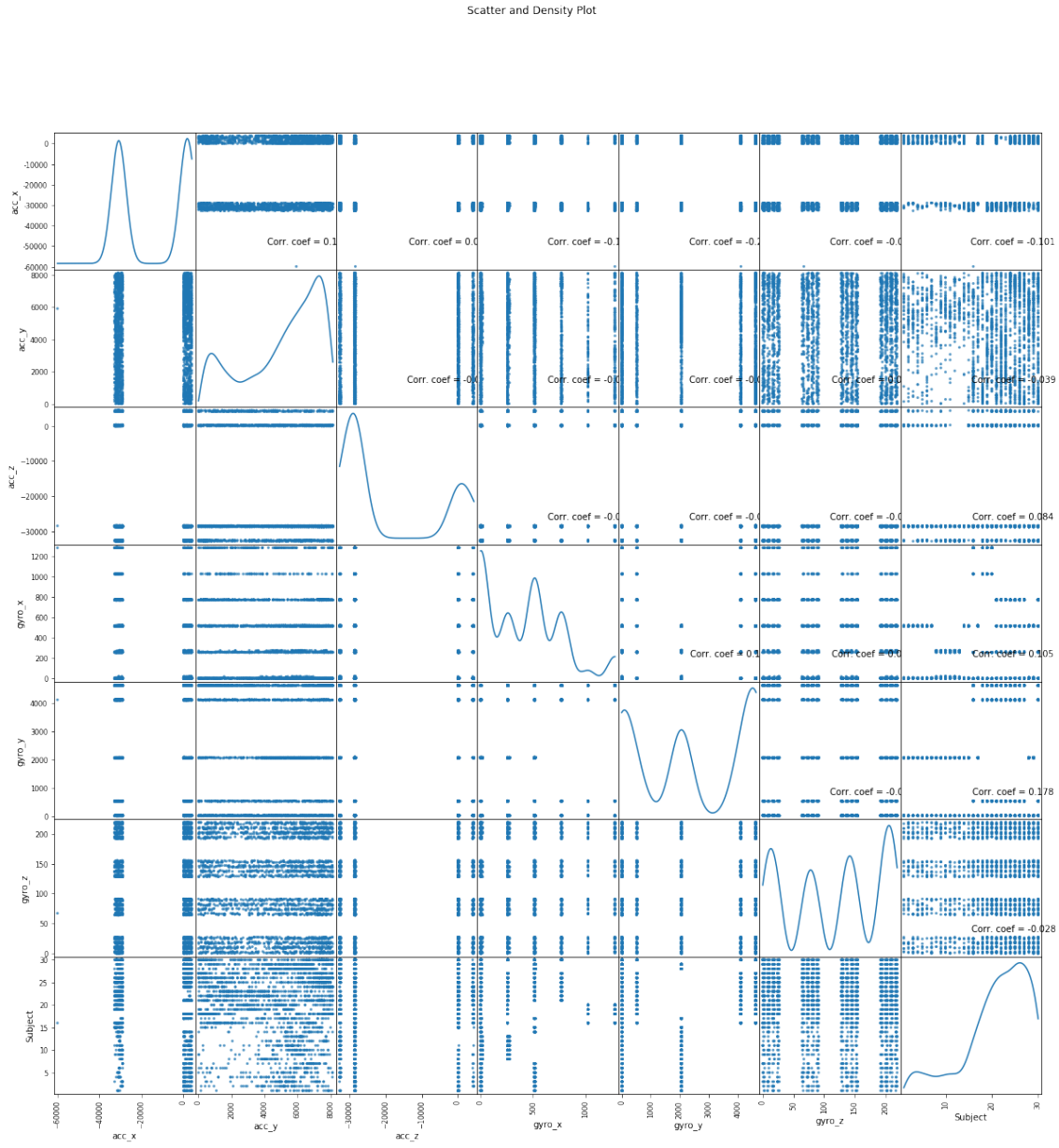


Figure 3.16: Scatter and Density Plot of new dataset

Histogram plot

Histogram diagram shown in figure 3.17 consists of rectangles whose area is proportional to a variable's frequency and whose size is equal to the distance between groups. The range of operation in this new dataset is 0.00-0.35.

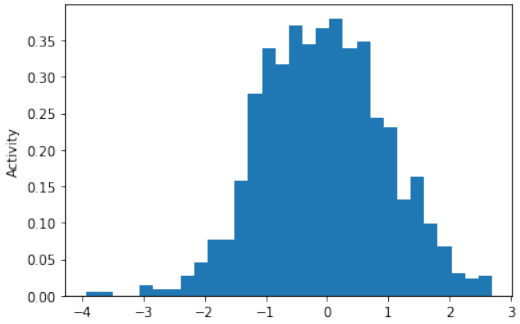


Figure 3.17: Histogram of new dataset

Pie Plot

In the pie chart, the percentage of information in each class is represented as a slice of the ring. Here figure 3.18 shows the pie plot of the new dataset.

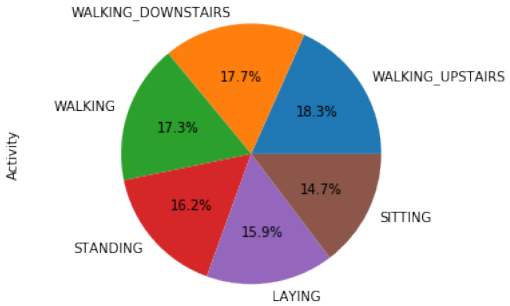


Figure 3.18: Pie chart of new dataset

Count Plot

Figure 3.19 shows the barplot of new dataset where the dependent variable is the number of instances of each instance of the independent variables like human activities.

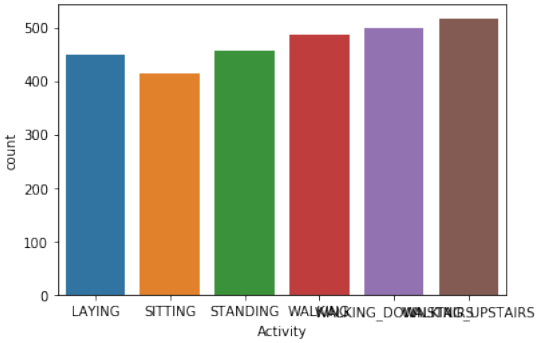


Figure 3.19: Count plot of new dataset

Heatmap Plot

Here new dataset has 8 columns and the graphical representation (heatmap) of each column is shown in figure 3.20 below.

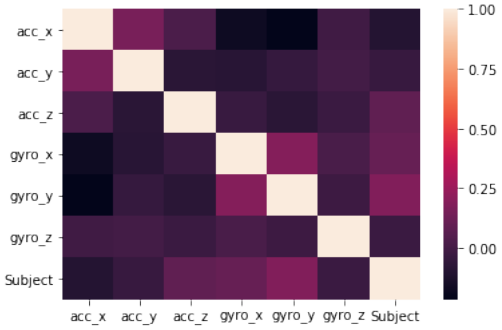


Figure 3.20: Heatmap plot of new dataset

Correlation Matrix

Figure 3.21 shows the correlation matrix of new dataset where it shows the correlation of each columns.

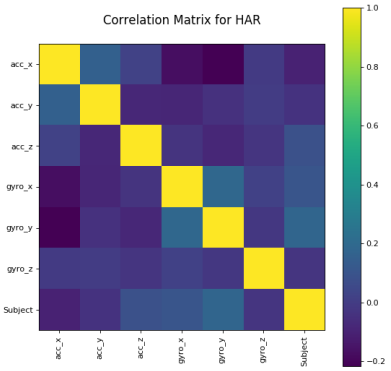


Figure 3.21: Correlation matrix of new dataset

Chapter 4

Methodology

4.1 Machine Learning

4.1.1 Linear regression

Linear regression is a statistical method for modeling the most generalized linear fit (linear formula, using the least squared approach) between a dependent variable and one or more explanatory variables (or independent variables). When regressors, represent inputs or causes, a statistical background is an independent variable. This input variable, essentially mapped to predict the potential outcome in linear line equation, is referred to as dependent variables where dependent variables represent the result or outcome from which the variance is being studied. Figure 4.1 shows the connection between dependent and independent variable where the drawn line is called regression line.

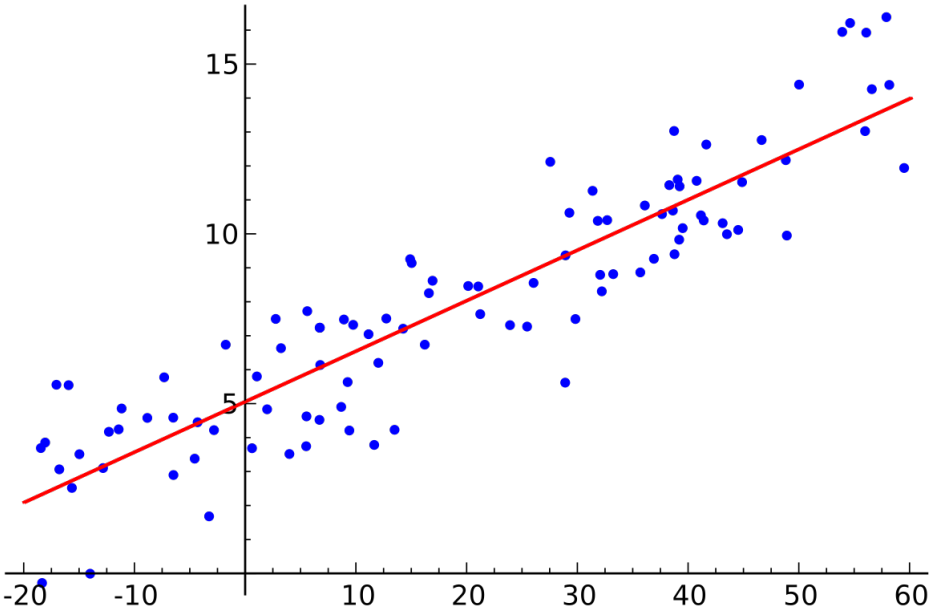


Figure 4.1: Linear regression with regression line scattering [44]

Types of linear regression

There are two types of linear regression-

- Simple linear regression: Simple linear regression is used to determine the relationship between two continuous variables. This searches for a correlation that is quantitative but not deterministic. It is said that the relationship between two variables is deterministic if the other can express one variable accurately. The formula of linear regression is-

$$y = \beta_0 + \beta_1 x \tag{4.1}$$

Here,
x=explanatory variables
y=dependent variable
$\beta_0 = y - intercept(constantterm)$
$\beta_1 = slopecoefficientsfortheexplanatoryvariable$

- Multiple linear regression: There is more than one explanatory variable for multiple linear regression. Multiple correlated dependent variables are predicted here, rather than a single scalar variable. It is a line fit between multiple inputs and one output. The Formula for Multiple Linear Regression is-

$$y_i = \beta_0 + \beta_1 x_i 1 + \beta_2 x_i 2 + .. + \beta_p x_i p + \epsilon \tag{4.2}$$

where, for i=n observations: yi=dependent variable
Here,
$x_i = explanatoryvariables$
$\beta_0 = y - intercept(constantterm)$
$\beta_p = slopecoefficientsforeachexplanatoryvariable$
$\epsilon = themodel'serrorterm(alsoknownastheresiduals)$

Multiple regression, in fact, is the extension of ordinary minus squares (OLS) regression involving more than one explanatory factor.

Basic concepts of linear regression

- Correlation (r): Explains the relationship between two variables, possible values -1 to +1.
- Variance (σ_2) : $Measuresthedissemination of yourdata.$

- Standard Deviation (σ) : $Measureofspreadinyourdata(SquarerootofVariance)$.
- Normal distribution: Normal distribution, also known as the Gaussian distribution, is a symmetrical distribution of probability over the mean, indicating that data close to the mean occurs more often than data far from the mean. Normal distribution appears as a bell curve in graph form.

- Residual (error term): Actual value (Which we have) Minus Predicted value (Which came using linear regression).

Linear Regression Learning Model Type

1. Simple Linear Regression: Simple linear regression where one input can be used to estimate the coefficients using statistics. This involves the calculation of statistical data properties such as averages, standard deviations, correlations, and co variance.

2. Ordinary Least Squares: Ordinary Least Squares are used to approximate the coefficient values when more than one source is present. The procedure of the Ordinary Least Squares seeks to minimize the squared residuals.

3. Gradient Descent: For each coefficient, it works by starting with random values. For each pair of input and output values, the sum of squared errors is calculated. As a scale factor, a learning rate is used and the coefficients are updated to minimize the error. The process is repeated until a minimum sum squared error is achieved or no further improvement is possible.

4. Regularization: It is an extension of the linear model where it minimizes the sum of the square error of the variable on the training data by using ordinary least squares. It also reduces the complexity of the equation, such as the number of all model coefficients or the total size of the sum.

4.1.2 Logistic Regression

For classification problems, logistic regression is used. It is an algorithm based on a predictive analysis theory of probability. It uses a more complex cost range defined as the ' Sigmoid function ' instead of a linear function. It tends to limit the function of costs from 0 to 1. It is therefore not represented by linear functions as it can have a value greater than 1 or less than 0, which is not possible due to the logistic regression hypothesis. The logistic regression algorithm additionally makes use of a linear equation with unbiased predictors to predict a value. The expected value can be anywhere, from negative infinity to positive infinity. We want the algorithm's output to be a variable of class type. For instance, 0 as no 1 as yes. Thus, the linear equation's output is squashed into a range of [0,1]. The sigmoid function is used to smash the expected value between 0 and 1.

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + .. \tag{4.3}$$

$$g(x) = \frac{1}{1 + e^{-}x} \tag{4.4}$$

So Linear equation and sigmoid function is-

$$h = g(z) = \frac{1}{1 + e^{-}z} \tag{4.5}$$

Here,
h= Squashes output
z= output of the linear equation
g(x)= function which returns a squashed value h, where the value h will lie in the range of 0 to 1.
Here the figure 4.2 underneath shows the chart of the sigmoid function where the sigmoid function becomes asymptote to y=1 for positive estimations of x and becomes asymptote to y=0 for negative estimations of x.

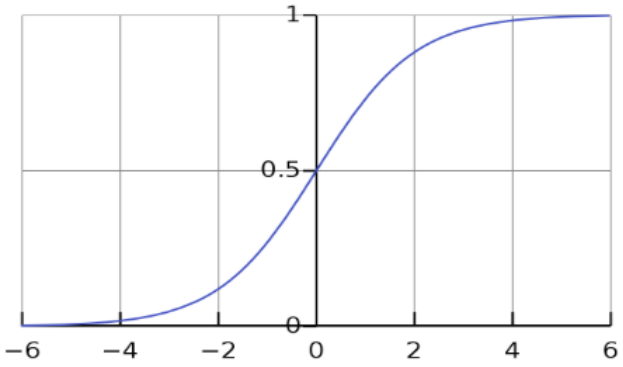


Figure 4.2: Sigmoid function graph
[45]

4.1.3 Random Forest Classifier

Random forest is a supervised algorithm for machine learning that uses a decision tree approach to forest creation. It develops a lot of decision trees based on random data selection and random variables selection that provides the class of dependent variables based on a lot of trees. The higher the chance of getting accurate results, the greater the number of trees. Random forest results in high precision and understanding very quickly. On large data sets, it is very effective. Random forest uses the gini index to evaluate each tree's final rank.If data set T contains examples from n classes' gini index, Gini (T) is defined as-

$$Gini(T) = 1 - \sum (PJ)^2 \tag{4.6}$$

4.1.4 Decision Tree

Decision Tree is an algorithm used to identify and regress supervised learning. This works by dividing the information into two or more sub-sets based on the values of the input factor. A cost function or splitting criteria is used to determine the best split (one with the lowest cost) between all the split points. The data is divided into groups recursively until there is only one sample in the leaves. For implement the Decision Tree classifier using Scikit-Learn, an improved version of the CART (Classification and Regression Trees) algorithm is used in this model. Gini impurity is used to calculate the uncertainty as the dividing criterion. The other is ID3 (Iterative Dichotomiser 3) which uses Entropy Function and Knowledge Gain as metrics to implement the classifier.

4.1.5 Naïve Bayes (NB)

Naïve Bayes is a supervised learning algorithm that relies on the classification theorem of Bayes. The theorem of Bayes uses conditional probability, which in effect uses prior knowledge to measure the likelihood of a future event. The formula for Bayes' Theorem is [46]-

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.7}$$

Here,
P(A)= prior probability, i.e., the probability of the hypothesis being true
P(A| $B) = posterior probability, the probability that a hypothesis (A) is true given some evidence (B)$
$P(B) = probability of the evidence, irrespective of the hypothesis$
$P(B|A) = probability of the evidence when hypothesis is is true$

The input variables (features) are considered to be independent of each other in the Naïve Bayes classifier and all features must contribute independently to the target variable's likelihood. By having a conditional independence statement over the training dataset, it reduces uncertainty. This reduces the complexity of the above problem to just 2n drastically. The principle of conditional independence states that X is conditionally independent of Y given Z and given random variables X, Y and Z, if and only if the distribution of probability governing X is independent of Y given Z. In other words, X and Y are conditionally independent given Z if and only if, given the knowledge that Z occurs, knowledge of whether X occurs does not provide information on the likelihood that Y occurs, and knowledge of whether Y occurs does not provide information on the likelihood that X occurs; this assumption makes the Bayes algorithm naive. Given, n different attribute values, the likelihood now can be written as-

$$P(X_1...X_n|Y) = \prod_{i=1}^{n} P(X_i|Y) \tag{4.8}$$

Here,
X = the attributes or features
Y = the response variable
Now, P(X| Y) = $the products of probability distribution of each attribute X given Y$.
There are several different types of Naïve Bayes classifiers, among which was used in this model the Gaussian Naïve Bayes classifier. The classification of Gaussian Naïve Bayes assumes that the function values are continuous and that the values of belonging to each group are normally distributed.

4.1.6 Support vector machine(SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both SVC (Support Vector Classification) and SVR (Support Vector Regression) classification and regression. The main goal behind SVM is to find a hyperplane in an N-dimensional space (N — the number of features) that categorizes the data points as clearly as possible, separating features into different domains or regions. There are many potential hyperplanes that could be chosen to distinguish the two groups of data points. Maximizing the margin gap offers some support in order to be able to distinguish future data points with greater confidence.

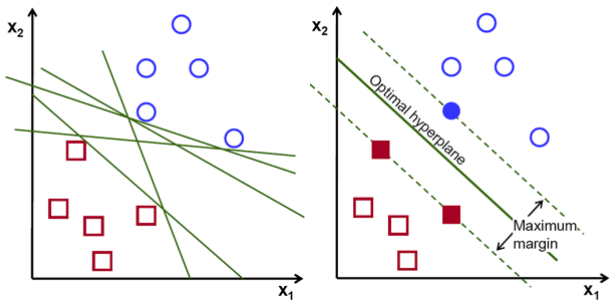


Figure 4.3: Hyperplane of support vector machine
[47]

Hyperplanes are decision boundaries that help distinguish points of data. Data points that fall on either side of the hyperplane can be assigned to different classes. Often, the dimension of the hyperplane depends on the number of features. If the number of input characteristics is 2, the hyperplane is just a row. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of apps exceeds 3. There are two types of SVM, one of which is linear SVM and the other is non-linear SVM. Linear SVM is mainly used to classify linear data sets. It has linear decision boundaries and is efficient in dealing with large multi-class multi-class datasets. It is more effective than non-linear SVM. Nevertheless, it is not appropriate for non-linear complex datasets. Non-linear boundaries, on the other hand, are used to deal with such issues. This is called SVM's kernel trick. Such functions transform low-dimensional input space into larger dimensional space to distinguish the problem of non-linear separation. It transforms extremely complex data and then separates data based on the labels or outputs that are described. Figure 4.3 shows the sample of placing hyperplane in SVM.

4.1.7 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a directed classifier which records every considerable case and describes retrieved cases that depend on an estimation of closeness, for example, separation capacities. It is an algorithm used to classify as well as to regress. However, it is more commonly used to solve classification problems. K is referred to as the number of neighbors closest or nearest to KNN. K is this algorithm's primary determining factor. In cases where K=1 is simply referred to as the nearest neighbor algorithm. Typically, K is an odd number. In figure 4.4 , when k=1 then the testing data are in the same level as the closest example of the training set . Similarly, when k=3 the the level of 3 closest class are checked and the most common level is assigned to the testing data.

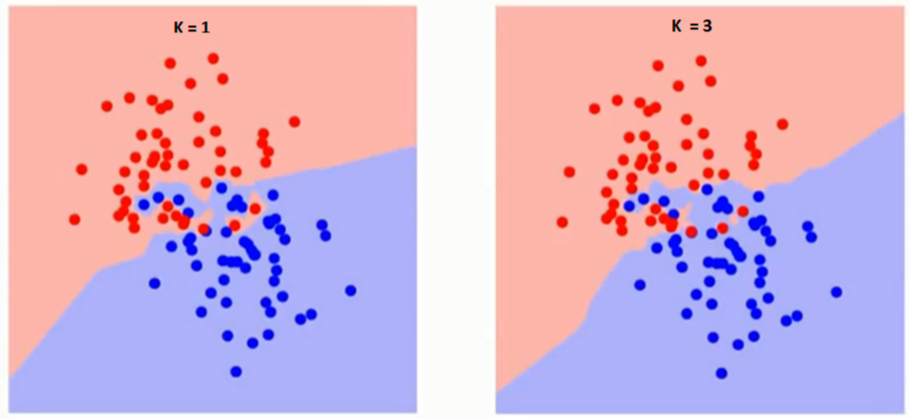


Figure 4.4: Visual representation of K factor
[48]

4.2 Neural Network

4.2.1 Multilayer perceptron (MLP)

Data is transmitted in one direction in MLPs across multiple interconnected layers. MLPs are usually made up of one or more neuron layers. Information is fed into the input layer, one or more layers may exist between where the information is finely tuned, and finally the prediction is given by the output layer, also called the visible layer. MLPs are effective when it comes to classification issues, i.e. where a category or label is input data data set. They also do well in estimation of regression. Because of a set of input data in tabular form in a spreadsheet or in a CSV format file, they estimate a real world value quantity in regression problems. MLPs are machine learning algorithms that are supervised. They are trained on sets of pairs of input-outputs and eventually learn to model the pairs ' interdependence. It is important to change the data parameters or biases to learn so that the percentage of errors remains small. "Backpropagation" is the technique used for these adjustments.
For ' Rectifier linear unit, ' ReLU is short form. It is the most common activation

function of a neural network's hidden layers. ReLU is ideal for non-linear datasets and that's why we used it as it can allow us to use multiple layers and help us to spread errors backwards. The equation for ReLU is:

$$A(z0) = max(0, z) \tag{4.9}$$

For an output the value of z has to be positive otherwise it will give zero.

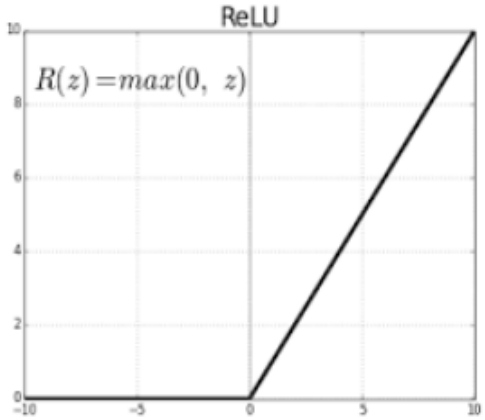


Figure 4.5: ReLU Activation Function Graph

Softmax is a type of sigmoid function that is really useful for multi-class dataset classification like ours. In classification, it is helpful as it would give us production 24 as 0 and 1 for each category. That's why we choose the Softmax function. The equation of the Softmax function is shown below:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{k} e^{z_k}} for j = 1, ..., k \tag{4.10}$$

4.2.2 Long short-term memory (LSTM)

LSTMs have an accomplishment over ordinary feed-forward neural networks and RNN in numerous respects. Reason behind this is the fact of remembering patterns for long time interval. LSTMs have been located as the most extraordinary arrangement for prediction problems. Succession expectation inconveniences are seen as one of the hardest issues to clear up in the records science industry. These comprise of a big scope of issues; from anticipating salary to finding designs in stock market data, from recognition film plots to perceiving your method for talking, from language translations to foresee your consequent word on the console of your iPhone[49].

An Overview of Recurrent Neural Networks (RNN)

Think about a case of financial exchange's information for a specific stock as a successive information. A basic model of AI or an Artificial Neural Network can

figure out how to anticipate stock costs dependent on various highlights: stock volume, opening worth, and so forth cost of the stock relies upon these highlights just as on the stock qualities in the earlier days. For a broker, these qualities are one of the major prescient factors in the prior days (or the pattern).
All experiments are viewed as autonomous in customary feed-forward neural systems. While planning the model for a specific day, there is no worry at stock expenses in the previous days.
This time dependence is worked on utilizing Recurrent Neural Networks. Figure 4.6 shows how an average RNN seems:

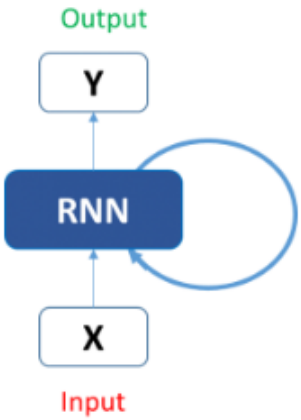


Figure 4.6: Folded pattern of RNN

it looks a lot simpler in figure 4.7 once it has unfolded:

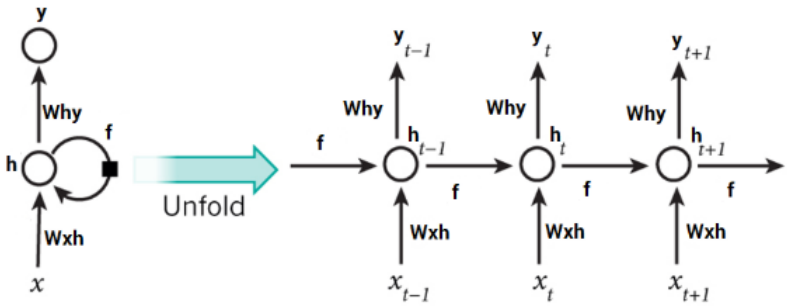


Figure 4.7: Unfolded pattern of RNN

Presently it's simpler to perceive how these systems take a gander at the pattern of stock costs, before foreseeing stock costs for now. Here each estimate at the time t (h t) depends upon all desired outcomes and the data they have aggregated.
To an enormous degree, yet not so much, RNNs can take care of our succession dealing with issue. Presently RNNs are incredible with regards to short sentences, yet to have the option to comprehend and recollect the importance behind the groupings, much the same as a human mind, we need our models to have the option to fabricate and recall a story. It's impractical With a basic RNN.

Limitations of Recurrent Neural Networks (RNN)

When managing transient dependence, intermittent neural systems perform well. For instance, The issue which has nothing to do with the setting of the announcement and hence RNN need not recollect information disclosed before this, or what was its importance, all they have to know is that the perception much of the time. Vanilla RNNs, in any case, don't comprehend the setting of an information. The RNN needs to recall this setting so as to make a right expectation. The applicable data can be confined from the point where it is required by a gigantic measure of superfluous information. There remains the disappointment of Recurrent Neural Network! A marginally changed adaptation of RNNs – the Long Short-Term Memory Networks – can take care of this issue.

Improved version of RNN: LSTM (Long Short-Term Memory) Networks

As far as including new data in RNN, it changes the current data totally by applying a capacity. All the data is changed along these lines, i.e. There is no remittance for ' significant ' information and ' not all that significant ' data. LSTMs, then again, use increases and augmentations to roll out little improvements to the data. The data courses through a system called cell states with LSTMs. It encourages LSTMs to review or overlook things specifically. The data has three unique conditions at a specific cell state. Relationship with transport lines is another significant element of LSTM. LSTMs utilize the data development strategy which can be recipient for ventures moving items for various purposes. As it moves through the various layers, we may have some expansion, adjustment or evacuation of data similarly as an item can be shaped, painted or pressed on a transport line. The accompanying outline clarifies the cozy relationship of LSTMs and transport lines.

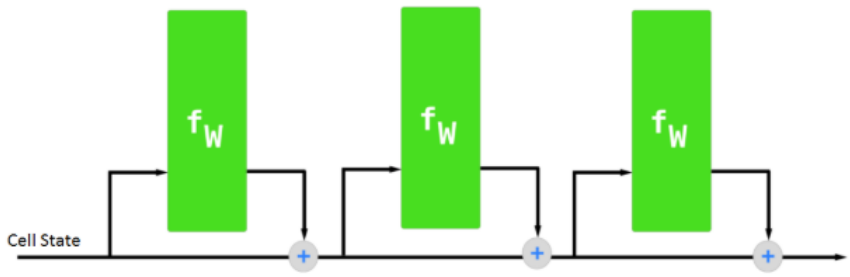


Figure 4.8: Outline of LSTM

As LSTMs don't control the whole data but instead alter them somewhat, they can overlook and recall things specifically.
LSTMs are an exceptionally encouraging answer for issues identified with grouping and time arrangement. The one drawback, be that as it may, is the test of preparing them. Indeed, even a straightforward model goes into preparing a great deal of time and machine cash. Be that as it may, this is only a limitation on equipment.

4.3 Dimension Reduction Techniques

For machine learning classification problems, on the basis of which the final classification is made, there are often too many variables. Such considerations are basically so-called features variables. The higher the number of features, the more difficult it becomes to envision and function on the training set. Most of these features are sometimes linked and therefore redundant. This is where algorithms for the reduction of dimensionality come into play. Reduction of dimensionality is the method of reducing the number of random variables being considered by obtaining a collection of key variables. It can be split into the collection of features and extraction of features[50].
Since we have a higher number of features, we have used 4 dimension reduction techniques. Which are PCA, SVD, LDA ICA.

4.3.1 Principle Component Analysis(PCA)

The key component examination approach is to infer significant factors (as parts) from a wide scope of factors in a dataset. This gathers highlights from a high-dimensional data set with a mission to gather expected amount of data. Visualization seems significantly reasonable with less factors. While managing 3 or higher dimensional data, it is progressively helpful. It is constantly accomplished on a network with balance or covariance. This guarantees the framework ought to be straight and the information ought to be structured.A head segment is a direct mix of the first factors which are extricated so that the primary head segment clarifies most extreme difference in the dataset and the second head part attempts to clarify the rest of the change in the dataset and is uncorrelated to the main head segment third head segment attempts to clarify the fluctuation which isn't clarified by the initial two head segments, etc. Here in the figure 4.9 below PC1 and PC2 are the principal components.

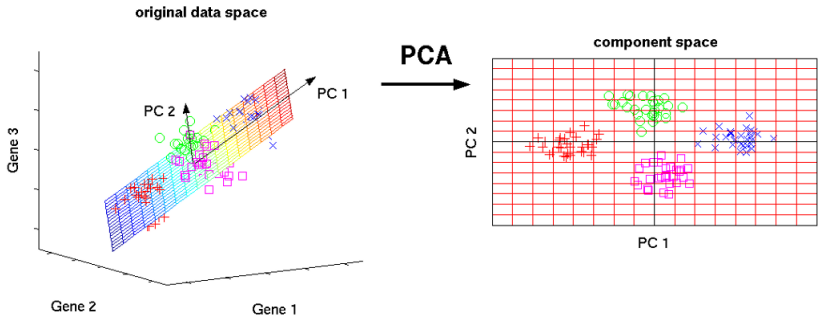


Figure 4.9: Principle Component Analysis
[51]

4.3.2 Singular Value Decomposition(SVD)

SVD(Singular Value Decomposition) is a technique of dimension reduction that can be used to reduce the dimensionality of a data set, such as the number of columns. It is often used for noise reduction, image compression and other areas of digital signal processing. This divides the first factors into three matrices. It is principally

used to wipe out the repetitive highlights of the informational collection. This uses the idea of Eigenvalues and Eigenvectors to test these three grids.
SVD is an algorithm that makes a m x n matrix, M, of real or complex values into three element matrices where factorization has the USV* form as shown in figure 4.10.
Here,
U = m x p matrix.
S = p x p diagonal matrix.
V = n x p matrix, For V * transposing V, p x n matrix, or transposing the conjugate if M comprises complex values.
The value p is called the rank. The diagonal entries of S are considered the unique values of M. U columns are usually called left-singular vectors of M, and columns of V are considered right-singular vectors of M.

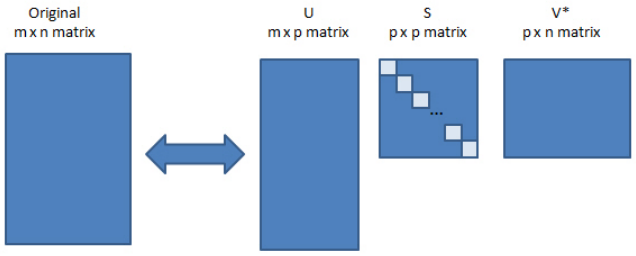


Figure 4.10: Singular Value Decomposition matrices
[52]

4.3.3 Linear discriminant analysis (LDA)

Linear discriminant analysis (LDA) is an algorithm for dimension reduction, meaning that it reduces the number of dimensions from the original to C— 1 number of characteristics where C is the number of groups. LDA is a supervised learning algorithm where it determines the directions (linear discriminants) that will represent the axes that maximize multiple class separation. Although for a multi-class identification function in which category labels are identified, LDA is superior to PCA. It helps to find the boundaries among category clusters. LDA projects data points on a line to distinguish clusters as much as possible, with each cluster having a relative (close) distance to a centroid. Essentially LDA seeks the centroid of each group data points.

4.3.4 Independent Component Analysis (ICA)

Independent Component Analysis (ICA) is an information theory-oriented dimension reduction technique. This technique expects that the factors given are direct blends of certain obscure dormant factors which are commonly free of the idle factors. To keep the elements as distinct as possible, ICA is searching for a non-mixing matrix.Non-Gaussianity is the most common method of measuring element independence. To evaluate individual elements, ICA uses non-Gussian methods. The maximized kurtosis would make the non-Gaussian's distribution and therefore individual components can be found by independent analysis of the components.

Chapter 5

Result Analysis

5.1 Existing Dataset

We have applied seven machine learning algorithm and two neural network method to analyse the existing dataset. The results of each dataset are given below-

5.1.1 Linear Regression

In Linear Regression, we have multiple explanatory (independent) variables which are the accelerometer and gyroscope values and also the values obtained from these by applying fast fourier, euclidean mathematical laws. Here, the dependent variables are the six human activities (Laying, Sitting, Standing, Walking, Walking downstairs, Walking Upstairs) to be predicted. We have kept the encoded value of dependent variables of train and test dataset into Y_train and Y_test variable respectively. Then the independent variables' values of train and test dataset are kept in X_train and X_test accordingly. Thus we have two independent variables X_train and X_test and two dependent variables Y_train and Y_test. Therefore, we have fit simple linear regression on X_train, Y_train and X_test, Y_test. The accuracy we have obtained is 98% for X_test, Y_test and 97% for X_train, Y_train. The results are given below:

```
#LINEAR REGRESSION
#test
reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9828347818906217
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.9740562043782035
```

The variance score of linear regression is 95%. The Coefficients' array are given below. Here figure 5.1 shows residual error of train and test data.

```
Coefficients:
 [ 1.42070326e+00 -3.68594145e-01 -4.86513445e-01  1.31427416e-01
  1.31832761e-02 -3.38707039e-01 -1.46723546e-01 -1.66944222e-01
 -2.03491686e-02  6.69791893e-02  1.59775259e-01 -1.34184654e-01
 -2.85319766e-02 -4.22590142e-02  3.26559073e-02 -9.71590896e-02
  3.93021556e-03 -2.39490255e-03  2.83331453e-02  8.08754640e-02
 -1.41720769e-02  2.40603049e-02 -1.31980487e-01 -3.32924166e-02
  2.33687803e-02  3.91815502e-02  1.02613261e-01  1.63694068e-02
  4.97102096e-02  1.05471134e-02 -3.32863741e-03 -5.69159542e-02
  7.16325504e-03 -4.17457374e-02 -1.35218351e-02  1.99293413e-02
 -9.44898353e-03 -7.05710951e-02 -1.27959226e-02 -4.91333114e-02
 -3.64430624e-02 -1.14615999e-02  2.72473865e-02 -5.54298150e-02
 -1.09309999e-02  2.61115753e-02 -1.17509134e-02 -1.11584471e-03
 -7.80173307e-04  2.40422035e-02  9.81875584e-03  5.26106146e-03
 -2.59714995e-03  3.03227323e-02 -2.96492647e-03  3.82597483e-02
  5.04250521e-02 -1.01454275e-01 -4.72243221e-02  3.81460922e-03
  5.35443429e-03 -7.21886376e-03  7.86596092e-02 -5.61696904e-02
 -5.10481116e-02 -9.84051872e-03  6.15696136e-02  3.06751704e-02
  5.16955662e-02 -4.53125251e-03 -9.04735694e-03  4.34885653e-02
 -7.93466925e-02 -4.92178568e-02  3.33901502e-02  3.55103040e-02
  5.76101035e-02  4.28550484e-02 -2.26808034e-02  2.83602284e-03
  7.51659710e-03  3.90849292e-02 -8.65985612e-03 -6.02138120e-02
  1.69167563e-02  9.63132408e-03 -7.03194496e-02  1.23155805e-02
  3.28978866e-03 -1.74537165e-02  8.45533641e-04 -2.66973898e-02
 -2.70795037e-02 -2.82710307e-02 -2.80161756e-02  1.98300748e-03
  1.43579791e-02 -9.40755356e-03  3.77236001e-03  1.08003377e-02
  2.11757854e-02  1.98034182e-02 -4.08991870e-03  1.13898753e-03
  3.32065778e-03 -1.11991613e-03  3.52792458e-02  1.92432852e-02
  2.57421055e-02 -2.48837166e-02  5.95428015e-03 -1.43524641e-02
  1.06105646e-02 -7.41594518e-03 -8.63801776e-03  6.67764433e-02
 -4.99174376e-03  1.34053224e-04  2.87742502e-03  1.49936186e-02
  2.47485860e-02 -3.24703925e-02 -1.88066559e-02 -1.90041309e-02
 -1.48636860e-03 -2.73182869e-03 -9.35034874e-03  1.40716449e-02
 -9.35548249e-03  8.60129858e-03 -4.05709960e-02 -2.66686114e-02
  3.93786259e-02 -8.13144007e-03  2.70629180e-02  4.10596713e-03
 -1.14800171e-02  3.20161606e-02  1.41178267e-02 -1.89560948e-03
  5.38143596e-02  4.40184824e-03 -3.24186929e-03  5.63394379e-03
  4.12136857e-03 -3.09192890e-02  2.55311897e-02 -1.82320492e-03
  1.52349476e-02 -1.75939460e-03]
Variance score: 0.9551218408717504
```

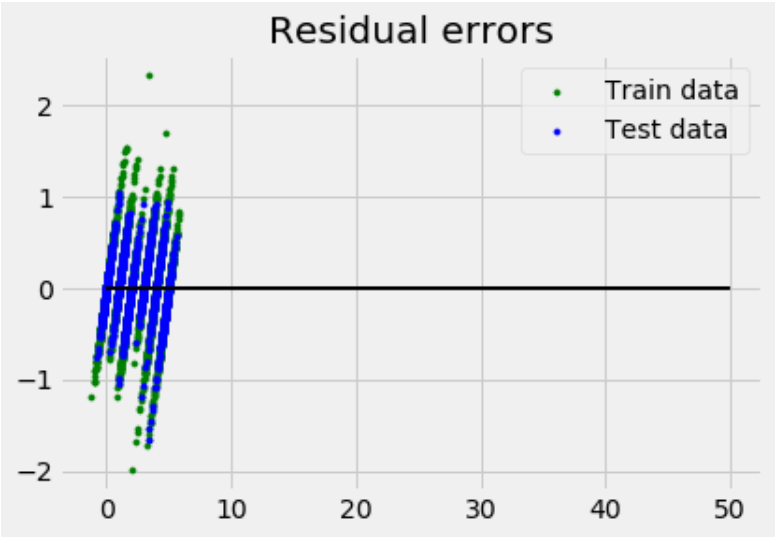


Figure 5.1: Residual error of linear regression

Then we moved to first dimension reduction technique Principle Component Analysis (PCA). In PCA obtained accuracy for both sets of variables is 92% while number of components were 40. But the accuracy noticeably increased to 95% for both sets of variables while the component numbers were increased to 150.

```
#LINEAR REGRESSION
#test
reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9205937672285065
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.9186877571205487
```

```
#LINEAR REGRESSION
#test
reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9550085950028344
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.9523277930422857
```

Next technique is Singular Value Decomposition (SVD). In SVD obtained accuracy for both sets of variables is 94% while the number of components were 100.

```
#LINEAR REGRESSION

reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.948174812170856
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.944498371678085
```

Third technique is Linear Discriminant Analysis (LDA) where the accuracy for X_train,Y_train is 97% and X_test, Y_test is 96.8% for 40 components.

```
#linear Regression
from sklearn import linear_model

reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9688724036598746
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.9736610431429796
```

Last reduction technique is Independent Component Analysis (ICA). Here for 20 components we got 90% accuracy for both sets of variables. When we increased components to 150, we got 95% accuracy for both sets of variables train and test.

```
#LINEAR REGRESSION
#test
reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9031352540900438
```

```
#linear Regression
from sklearn import linear_model

reg= linear_model.LinearRegression()
reg.fit(X_test,Y_test)
reg.score(X_test,Y_test)
```

```
0.9552782622082658
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.9052362996292423
```

```
#train
reg.fit(X_train,Y_train)
reg.score(X_train,Y_train)
```

```
0.952804678225469
```

Graphical representation of all the accuracy of linear regression is given below in figure 5.2. It shows us that maximum accuracy is 98.47% which is obtained by simple linear regression without applying reduction techniques.

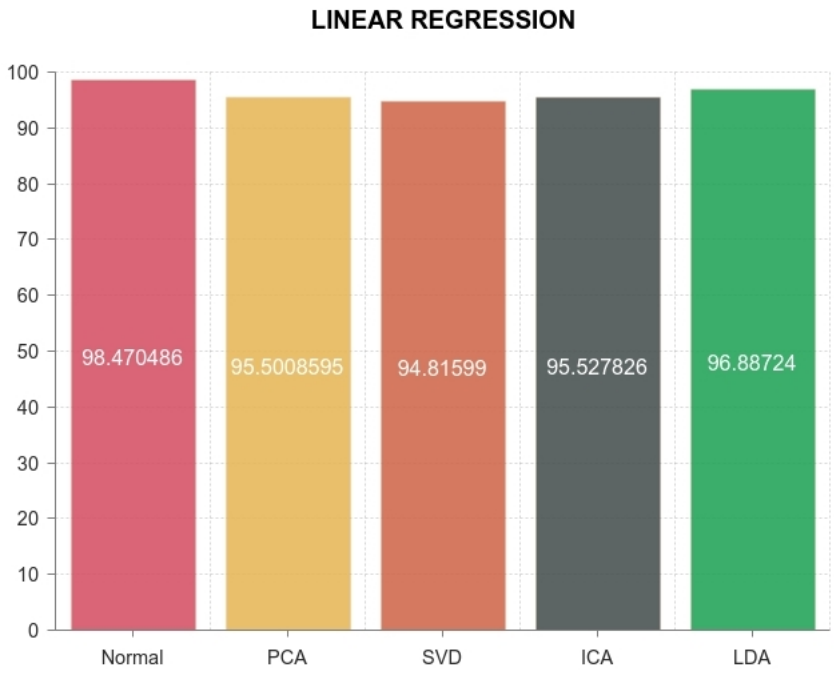


Figure 5.2: Result analysis of linear regression

5.1.2 Logistic Regression

In logistic regression we have fit a set of independent and dependent variable which are X_train and Y_train into the classifier. Then we did the predictive analysis between classifier and X_test which leads to 96% accuracy. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.85-1.00. Support values refers to 537, 491, 532, 496, 420 and 471 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively. The accuracy score, confusion matrix, classification report and plot for normalized confusion matrix (figure 5.3) for logistic regression are given below:

```
For Logistic Regression accuracy score is  0.9619952494061758
For  Logistic Regression confusion_matrix is:

 [[537   0   0   0   0   0]
 [  0 432  55   0   0   4]
 [  0  13 517   2   0   0]
 [  0   0   0 494   2   0]
 [  0   0   0   4 407   9]
 [  0   0   0  23   0 448]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00       537
           1       0.97      0.88      0.92       491
           2       0.90      0.97      0.94       532
           3       0.94      1.00      0.97       496
           4       1.00      0.97      0.98       420
           5       0.97      0.95      0.96       471

    accuracy                           0.96      2947
   macro avg       0.96      0.96      0.96      2947
weighted avg       0.96      0.96      0.96      2947
```

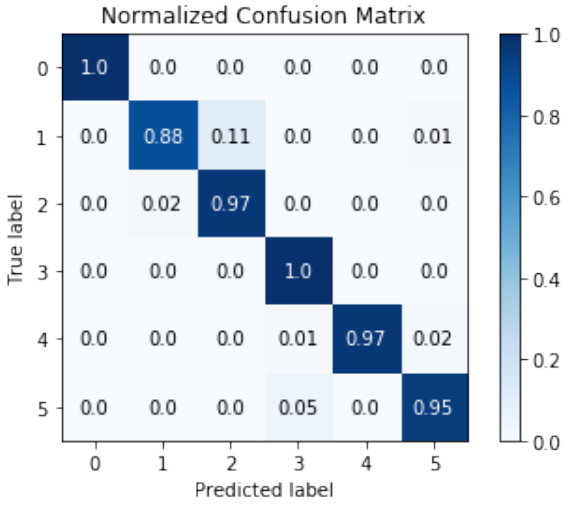


Figure 5.3: Confusion matrix for logistic regression

In PCA obtained accuracy is 93% while number of components were 40. But the accuracy noticeably increased to 98% while the component numbers were increased to 150. In both cases the number of total occurrences of the activities is 1471.

```
For Logistic Regression accuracy score is  0.9354180829367778
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [  3 241  40   0   0   0]
 [  0  37 239   0   0   0]
 [  0   0   0 238   0   2]
 [  0   0   0   0 175   3]
 [  0   0   0   4   6 221]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       0.99      1.00      0.99       262
           1       0.87      0.85      0.86       284
           2       0.86      0.87      0.86       276
           3       0.98      0.99      0.99       240
           4       0.97      0.98      0.97       178
           5       0.98      0.96      0.97       231

    accuracy                           0.94      1471
   macro avg       0.94      0.94      0.94      1471
weighted avg       0.94      0.94      0.94      1471
```

```
For Logistic Regression accuracy score is  0.982324949014276
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [  0 277   7   0   0   0]
 [  0  13 262   0   0   1]
 [  0   0   0 239   0   1]
 [  0   0   0   0 177   1]
 [  0   0   0   1   2 228]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.96      0.98      0.97       284
           2       0.97      0.95      0.96       276
           3       1.00      1.00      1.00       240
           4       0.99      0.99      0.99       178
           5       0.99      0.99      0.99       231

    accuracy                           0.98      1471
   macro avg       0.98      0.98      0.98      1471
weighted avg       0.98      0.98      0.98      1471
```

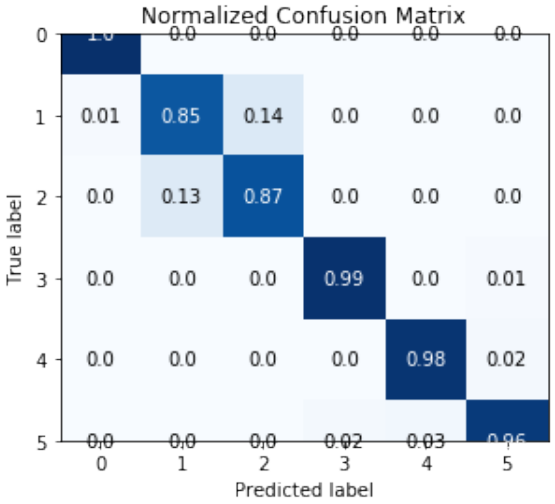


Figure 5.4: Confusion matrix for logistic regression by applying 40 components PCA

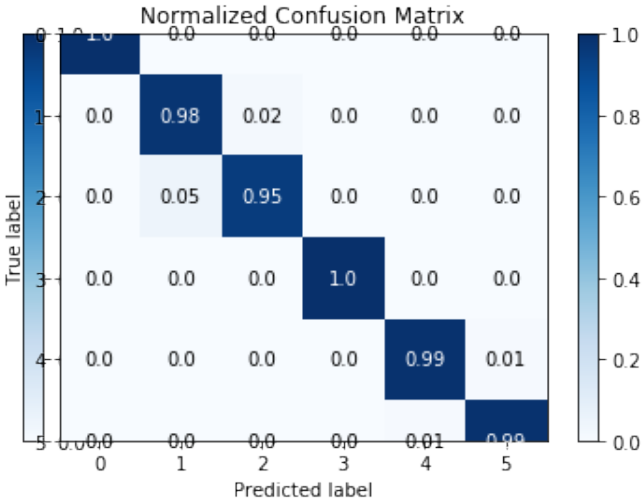


Figure 5.5: Confusion matrix for logistic regression by applying 150 components PCA

In SVD we got 97.82% accuracy while number of components were 100 and total occurrences of six activities were 1471.

```
For Logistic Regression accuracy score is  0.9782460910944936
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [  1 269  14   0   0   0]
 [  0  10 265   0   0   1]
 [  0   0   0 239   0   1]
 [  0   0   0   0 175   3]
 [  0   0   0   2   0 229]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.96      0.95      0.96       284
           2       0.95      0.96      0.95       276
           3       0.99      1.00      0.99       240
           4       1.00      0.98      0.99       178
           5       0.98      0.99      0.98       231

    accuracy                           0.98      1471
   macro avg       0.98      0.98      0.98      1471
weighted avg       0.98      0.98      0.98      1471
```

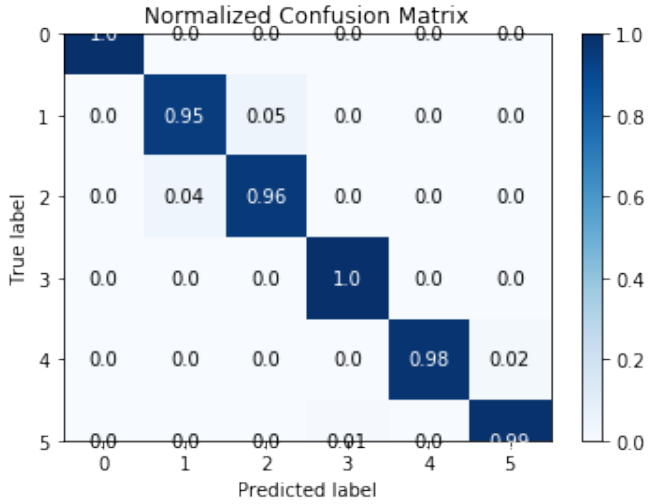


Figure 5.6: Confusion matrix for logistic regression by applying SVD

In LDA accuracy is 98% for 40 components and 1471 occurrences of the activities.

```
  this warning.', FutureWarning)
For Logistic Regression accuracy score is  0.9809653297076818
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [  2 272  10   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   0 178   0]
 [  0   0   0   1   1 229]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       0.99      1.00      1.00       262
           1       0.95      0.96      0.96       284
           2       0.96      0.95      0.96       276
           3       1.00      1.00      1.00       240
           4       0.99      1.00      1.00       178
           5       1.00      0.99      0.99       231

    accuracy                           0.98      1471
   macro avg       0.98      0.98      0.98      1471
weighted avg       0.98      0.98      0.98      1471
```

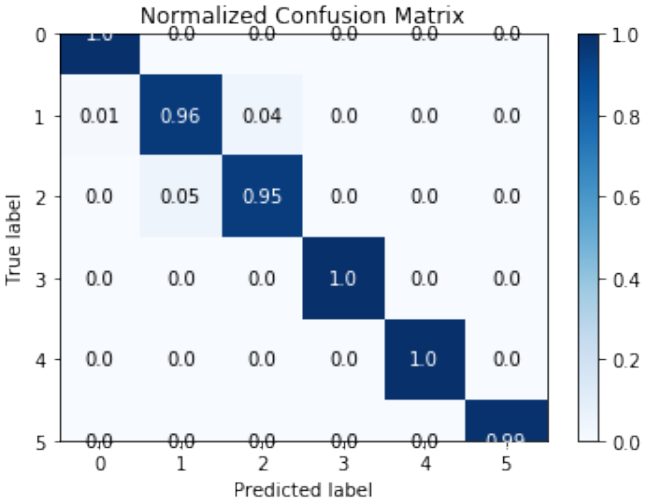


Figure 5.7: Confusion matrix for logistic regression by applying LDA

In ICA our obtained accuracy is 83% while number of components were 20. But the accuracy increased to 94.76% while the component numbers were increased to 150. In both cases the number of total occurrences of the activities is 1471.

```
  "this warning.", FutureWarning)
For Logistic Regression accuracy score is  0.831407205982325
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [ 24 145 113   2   0   0]
 [  0  31 245   0   0   0]
 [  0   0   1 235   0   4]
 [  3   0   2  22 149   2]
 [  1   0   5  37   1 187]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       0.90      1.00      0.95       262
           1       0.82      0.51      0.63       284
           2       0.67      0.89      0.76       276
           3       0.79      0.98      0.88       240
           4       0.99      0.84      0.91       178
           5       0.97      0.81      0.88       231

    accuracy                           0.83      1471
   macro avg       0.86      0.84      0.84      1471
weighted avg       0.85      0.83      0.83      1471
```

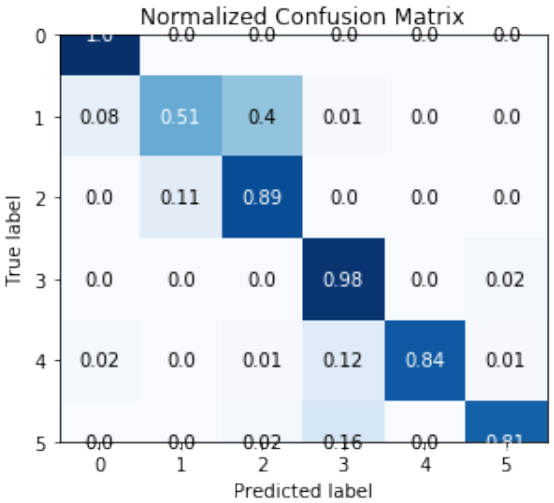


Figure 5.8: Confusion matrix for logistic regression by applying 20 components ICA

```
For Logistic Regression accuracy score is  0.947654656696125
For  Logistic Regression confusion_matrix is:

 [[262   0   0   0   0   0]
 [  3 244  37   0   0   0]
 [  0   9 267   0   0   0]
 [  0   0   0 239   0   1]
 [  1   0   2   4 170   1]
 [  0   1   0  17   1 212]]
For  Logistic Regression Classification Report:

              precision    recall  f1-score   support

           0       0.98      1.00      0.99       262
           1       0.96      0.86      0.91       284
           2       0.87      0.97      0.92       276
           3       0.92      1.00      0.96       240
           4       0.99      0.96      0.97       178
           5       0.99      0.92      0.95       231

    accuracy                           0.95      1471
   macro avg       0.95      0.95      0.95      1471
weighted avg       0.95      0.95      0.95      1471
```

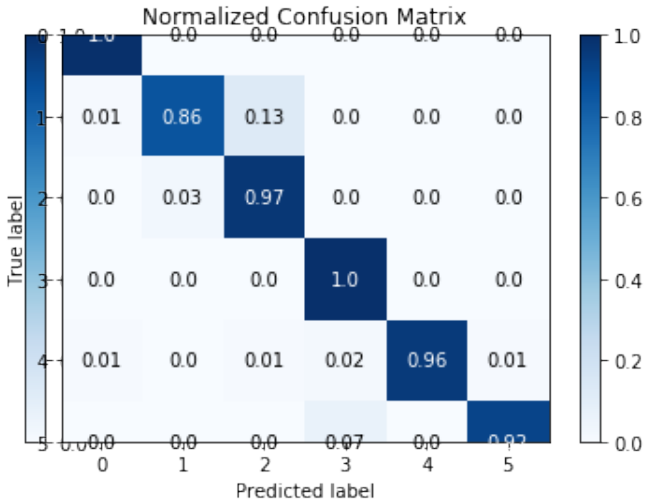


Figure 5.9: Confusion matrix for logistic regression by applying 150 components ICA

Lastly, the overall summarized accuracy for logistic regression in both cases; which are without applying reduction techniques and by applying reduction techniques are given below. In figure 5.10, we can see that the maximum accuracy is 98.23249% which is obtained by PCA where the number of components is 150.

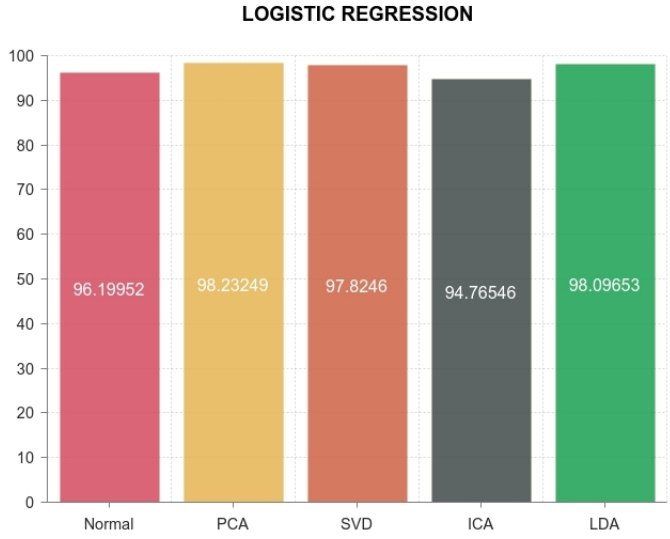


Figure 5.10: Result analysis of logistic regression

5.1.3 Random Forest Classifier

For random forest classifier we have applied imputation method for detecting null values and thus generating trees. After applying transform method on X_test we get the X_test_imp variable on which we applied random forest classifier and got the y_predicted variable. Thus we got 97.42% accuracy by fitting classifier on y_predicted and Y_test variables. The accuracy score, confusion matrix, classification report and normalised confusion matrix (figure 5.11) figures are given below:

```
97.42 %
[[261   0   0   0   0   1]
 [  0 275   9   0   0   0]
 [  0  16 260   0   0   0]
 [  0   0   0 238   0   2]
 [  0   0   0   3 172   3]
 [  0   0   0   0   4 227]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.95      0.97      0.96       284
           2       0.97      0.94      0.95       276
           3       0.99      0.99      0.99       240
           4       0.98      0.97      0.97       178
           5       0.97      0.98      0.98       231

    accuracy                           0.97      1471
   macro avg       0.98      0.97      0.97      1471
weighted avg       0.97      0.97      0.97      1471
```

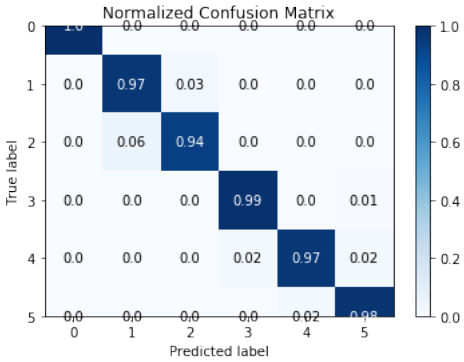


Figure 5.11: Confusion matrix for random forest classifier

In PCA obtained accuracy is 90.75% while number of components were 40. But the accuracy decreased a bit to 89.6% while the component numbers were increased to 150. In both cases the number of total occurrences of the activities is 1471.

```
90.75 %
[[262   0   0   0   0   0]
 [  5 227  52   0   0   0]
 [  0  49 227   0   0   0]
 [  0   0   0 233   3   4]
 [  0   0   0   6 170   2]
 [  0   0   0  10   5 216]]
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       262
           1       0.82      0.80      0.81       284
           2       0.81      0.82      0.82       276
           3       0.94      0.97      0.95       240
           4       0.96      0.96      0.96       178
           5       0.97      0.94      0.95       231

    accuracy                           0.91      1471
   macro avg       0.91      0.91      0.91      1471
weighted avg       0.91      0.91      0.91      1471
```

```
89.6 %
[[261   1   0   0   0   0]
 [  9 228  47   0   0   0]
 [  1  46 229   0   0   0]
 [  0   1   0 229   5   5]
 [  0   0   0  10 167   1]
 [  1   0   0  12  14 204]]
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       262
           1       0.83      0.80      0.81       284
           2       0.83      0.83      0.83       276
           3       0.91      0.95      0.93       240
           4       0.90      0.94      0.92       178
           5       0.97      0.88      0.93       231

    accuracy                           0.90      1471
   macro avg       0.90      0.90      0.90      1471
weighted avg       0.90      0.90      0.90      1471
```

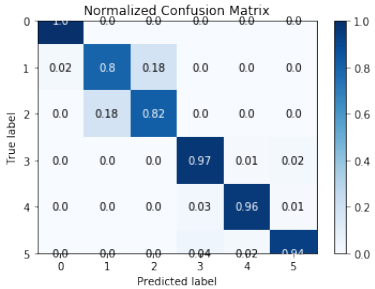


Figure 5.12: Confusion matrix for random forest classifier by applying 40 components PCA

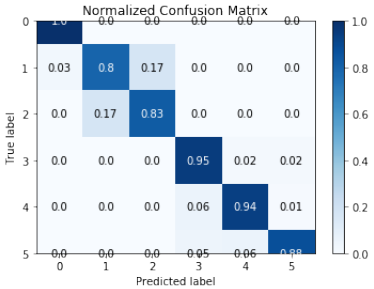


Figure 5.13: Confusion matrix for random forest classifier by applying 150 components PCA

In SVD, accuracy is 89.6% for 100 components and 1471 occurrences of activities.

```
89.6 %
[[261   0   0   0   0   1]
 [  4 230  49   0   0   1]
 [  7  53 216   0   0   0]
 [  0   0   0 234   3   3]
 [  0   0   0   9 169   0]
 [  0   0   0  11  12 208]]
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       262
           1       0.81      0.81      0.81       284
           2       0.82      0.78      0.80       276
           3       0.92      0.97      0.95       240
           4       0.92      0.95      0.93       178
           5       0.98      0.90      0.94       231

    accuracy                           0.90      1471
   macro avg       0.90      0.90      0.90      1471
weighted avg       0.90      0.90      0.90      1471
```

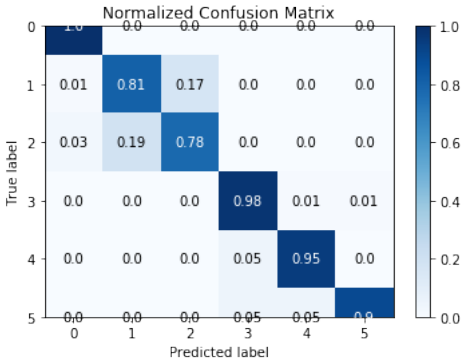


Figure 5.14: Confusion matrix for random forest classifier by applying SVD

In LDA, we got 98.03% accuracy for 40 components and 1471 occurrences of activities.

```
98.03 %
[[262   0   0   0   0   0]
 [  0 271  13   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   2   1 228]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.95      0.95      0.95       284
           2       0.95      0.95      0.95       276
           3       0.99      1.00      1.00       240
           4       0.99      1.00      1.00       178
           5       1.00      0.99      0.99       231

    accuracy                           0.98      1471
   macro avg       0.98      0.98      0.98      1471
weighted avg       0.98      0.98      0.98      1471
```

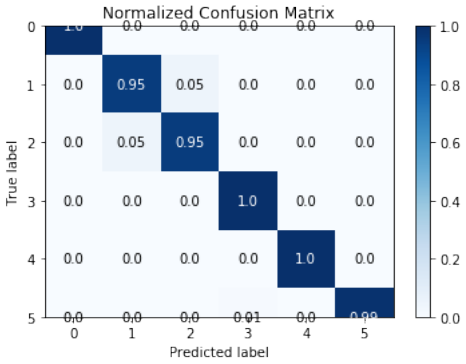


Figure 5.15: Confusion matrix for random forest classifier by applying LDA

In ICA, for 20 and 150 components we got 89.53% and 75.66% accuracies accordingly. In both cases the sum of support values or occurrences is 1471.

```
75.66 %
[[242   8   9   0   2   1]
 [ 25 198  52   6   0   3]
 [ 28  38 197   8   2   3]
 [  1   1   0 190  27  21]
 [  2   1   1  27 134  13]
 [  0   4   3  42  30 152]]
              precision    recall  f1-score   support

           0       0.81      0.92      0.86       262
           1       0.79      0.70      0.74       284
           2       0.75      0.71      0.73       276
           3       0.70      0.79      0.74       240
           4       0.69      0.75      0.72       178
           5       0.79      0.66      0.72       231

    accuracy                           0.76      1471
   macro avg       0.75      0.76      0.75      1471
weighted avg       0.76      0.76      0.75      1471
```

```
89.53 %
[[260   0   1   0   1   0]
 [  7 230  47   0   0   0]
 [  2  65 209   0   0   0]
 [  0   0   0 234   1   5]
 [  0   0   0   3 174   1]
 [  0   0   0   8  13 210]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       262
           1       0.78      0.81      0.79       284
           2       0.81      0.76      0.78       276
           3       0.96      0.97      0.96       240
           4       0.92      0.98      0.95       178
           5       0.97      0.91      0.94       231

    accuracy                           0.90      1471
   macro avg       0.90      0.90      0.90      1471
weighted avg       0.90      0.90      0.89      1471
```

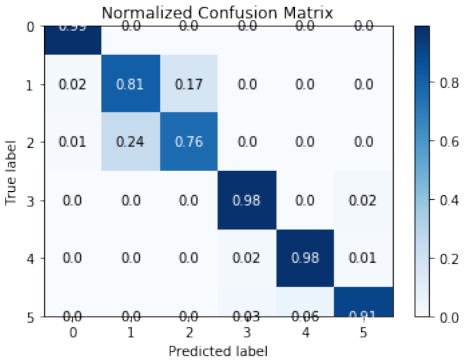


Figure 5.16: Confusion matrix for random forest classifier by applying 20 components ICA

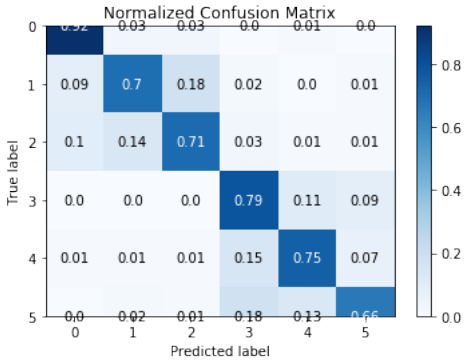


Figure 5.17: Confusion matrix for random forest classifier by applying 150 components ICA

Lastly, the overall summarized accuracy for Random forest classifier in both cases; which are without applying reduction techniques and by applying reduction techniques are given below. Figure 5.18 shows that the maximum accuracy is 98.03% which is obtained by LDA where the number of components is 40.

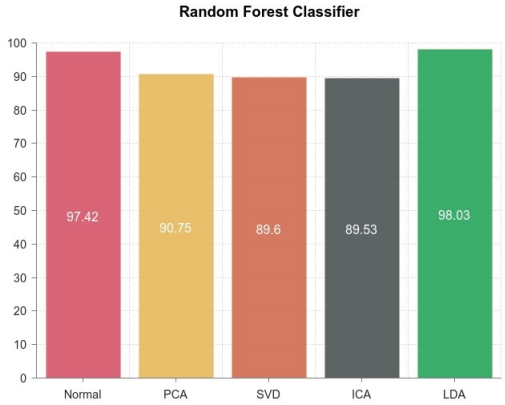


Figure 5.18: Result analysis of Random forest classifier

5.1.4 Support Vector Machine (SVM)

Firstly, we scaled the data of X_train and X_test using standardscaler. As we know modeling based on repetition of the same samples can predict the seen data but fails to analyze any unseen data which is called over fitting. That's why while using supervised machine learning algorithm cross validation (CV) is performed to tune parameters which helps to avoid over fitting. Here we used grid-search technique to perform CV for best fit SVM. C, gamma are parameters of Radial Basis Function (RBF) kernel SVM. Gamma parameter's value 0.0001 is very low which indicates that influence of single training set is far. Value of C parameter is 1000 which indicates that there remains smaller margin and the decision function is better at classifying all training points accurately. Accuracy for training and test data are 99.68% and 96.16%. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between

0.89-1.00. Support values refers to 537, 491, 532, 496, 420 and 471 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively. Total 2947 occurrences of six human activities are analyzed here.

```
[[537   0   0   0   0   0]
 [  0 436  54   0   0   1]
 [  0  15 517   0   0   0]
 [  0   0   0 493   3   0]
 [  0   0   0   5 398  17]
 [  0   0   0  16   2 453]]
```

	precision	recall	f1-score	support
LAYING	1.00	1.00	1.00	537
SITTING	0.97	0.89	0.93	491
STANDING	0.91	0.97	0.94	532
WALKING	0.96	0.99	0.98	496
WALKING_DOWNSTAIRS	0.99	0.95	0.97	420
WALKING_UPSTAIRS	0.96	0.96	0.96	471
accuracy			0.96	2947
macro avg	0.96	0.96	0.96	2947
weighted avg	0.96	0.96	0.96	2947

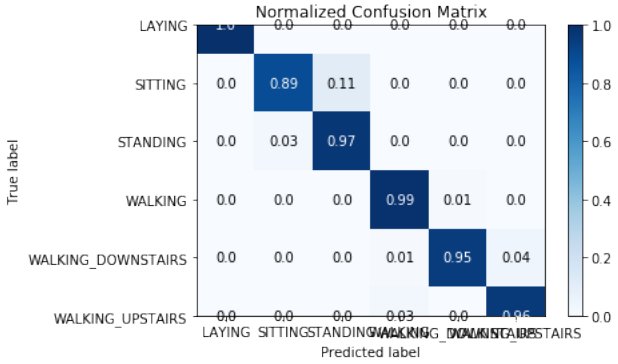


Figure 5.19: Confusion matrix for support vector machine

In PCA, we got 99.28% and 97.55% accuracies for train and test data where number of components were 40. When we increased the components to 150 we obtained 100% and 98.77% accuracies for train and test data.

```
[[262   0   0   0   0   0]
 [  0 266  18   0   0   0]
 [  0  15 261   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   3   0 228]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.95	0.94	0.94	284
2	0.94	0.95	0.94	276
3	0.99	1.00	0.99	240
4	1.00	1.00	1.00	178
5	1.00	0.99	0.99	231
accuracy			0.98	1471
macro avg	0.98	0.98	0.98	1471
weighted avg	0.98	0.98	0.98	1471

```
[[262   0   0   0   0   0]
 [  0 275   9   0   0   0]
 [  0   9 267   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   0   0 231]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.97	0.97	0.97	284
2	0.97	0.97	0.97	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.99	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.99	0.99	0.99	1471

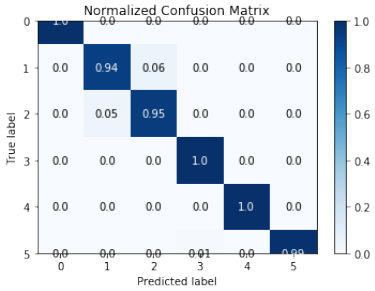


Figure 5.20: Confusion matrix for support vector machine by applying 40 components PCA

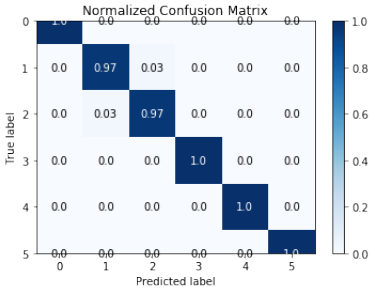


Figure 5.21: Confusion matrix for support vector machine by applying 150 components PCA

In SVD, for 100 components we got 100% and 98.16% accuracy for train and test data respectively.

```
[[262   0   0   0   0   0]
 [  0 274  10   0   0   0]
 [  0  16 260   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   0 178   0]
 [  0   0   0   0   0 231]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.94	0.96	0.95	284
2	0.96	0.94	0.95	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.98	1471
macro avg	0.98	0.98	0.98	1471
weighted avg	0.98	0.98	0.98	1471

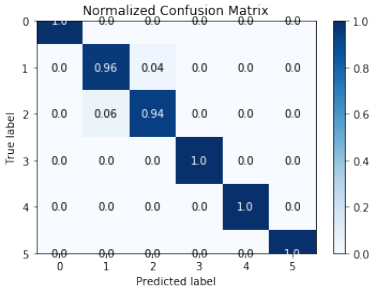


Figure 5.22: Confusion matrix for support vector machine by applying SVD

In LDA, we got 98.64% accuracy for train data and 98.23% accuracy for test data while the number of components were 40.

```
[[262   0   0   0   0   0]
 [  1 273  10   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   0 178   0]
 [  0   0   0   1   0 230]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.95	0.96	0.96	284
2	0.96	0.95	0.96	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.98	1471
macro avg	0.98	0.98	0.98	1471
weighted avg	0.98	0.98	0.98	1471

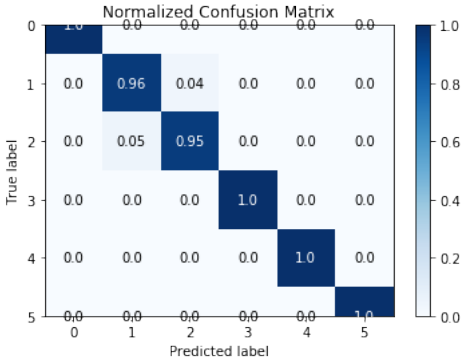


Figure 5.23: Confusion matrix for support vector machine by applying LDA

In ICA, we got 99.79% and 98.43% accuracy for train and test data for both cases of component numbers which are 20 and 150.

```
[[262   0   0   0   0   0]
 [  0 274  10   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   0   0 231]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.95	0.96	0.96	284
2	0.96	0.95	0.96	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.98	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.98	0.98	0.98	1471

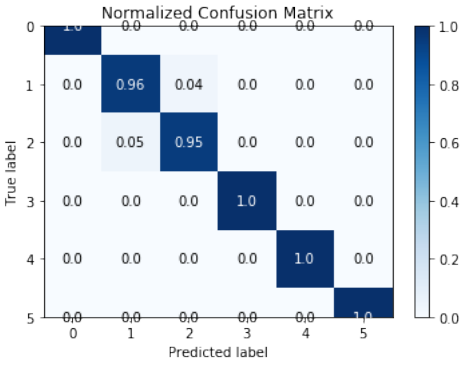


Figure 5.24: Confusion matrix for support vector machine by applying 20 components ICA

```
[[262   0   0   0   0   0]
 [  0 274  10   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   0   0 231]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.95	0.96	0.96	284
2	0.96	0.95	0.96	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.98	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.98	0.98	0.98	1471

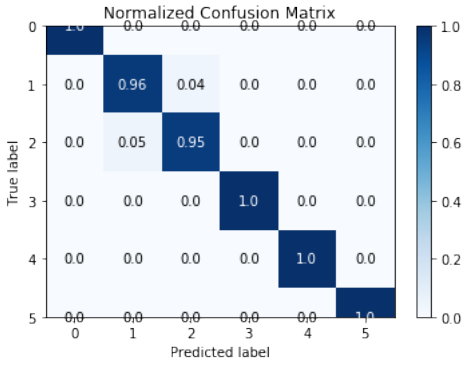


Figure 5.25: Confusion matrix for support vector machine by applying 150 components ICA

Lastly, in training set, figure 5.26 shows that , by applying PCA of 150 components and SVD we got 100% accuracy. On the other hand, figure 5.27 shows that, in the testing set, we got the highest accuracy by applying PCA , where the component is 100.

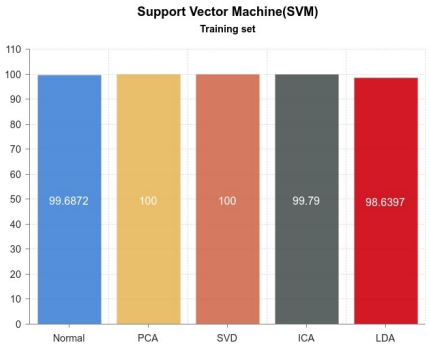


Figure 5.26: Result analysis of support vector machine (training set)

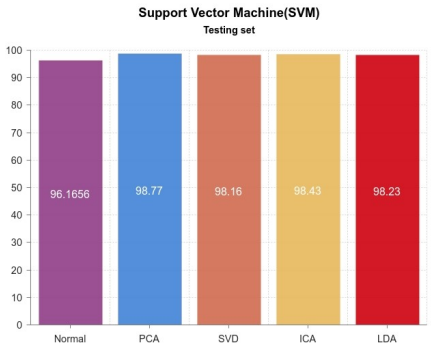


Figure 5.27: Result analysis of support vector machine (testing set)

5.1.5 K-Nearest Neighbors (KNN)

In KNN at first we scaled the X_train and X_test data which contains all the input features. Then we applied K-Neighbors Classifier on scaled X_train values and Y_train data which contains encoded variables of activities. Thus we got 85.34% accuracy from Y_test and Y-pred variable where Y_pred is obtained from scaled data of X_test variable. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.72-1.00. Support values refers to 537, 491, 532, 496, 420 and 471 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively. Total 2947 occurrences of six human activities are analyzed here. Here macro average and weighted average accuracy score is 86%.

```
85.34102477095351 %
[[497  28  12   0   0   0]
 [  2 354 134   0   0   1]
 [  0  68 464   0   0   0]
 [  0   0   0 474  17   5]
 [  0   0   0  64 305  51]
 [  0   0   0  39  11 421]]
              precision    recall  f1-score   support

           0       1.00      0.93      0.96       537
           1       0.79      0.72      0.75       491
           2       0.76      0.87      0.81       532
           3       0.82      0.96      0.88       496
           4       0.92      0.73      0.81       420
           5       0.88      0.89      0.89       471

    accuracy                           0.85      2947
   macro avg       0.86      0.85      0.85      2947
weighted avg       0.86      0.85      0.85      2947
```

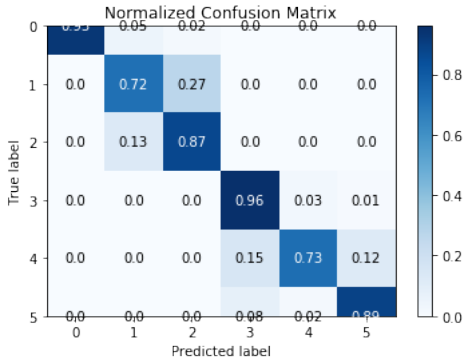


Figure 5.28: Confusion matrix forK-Nearest Neighbors

In the Error rate of K value graph in figure 5.29, we can see that we have the least error while the K value is 10 whereas we have the maximum error while the K value is 2.5 while number of components is 20.

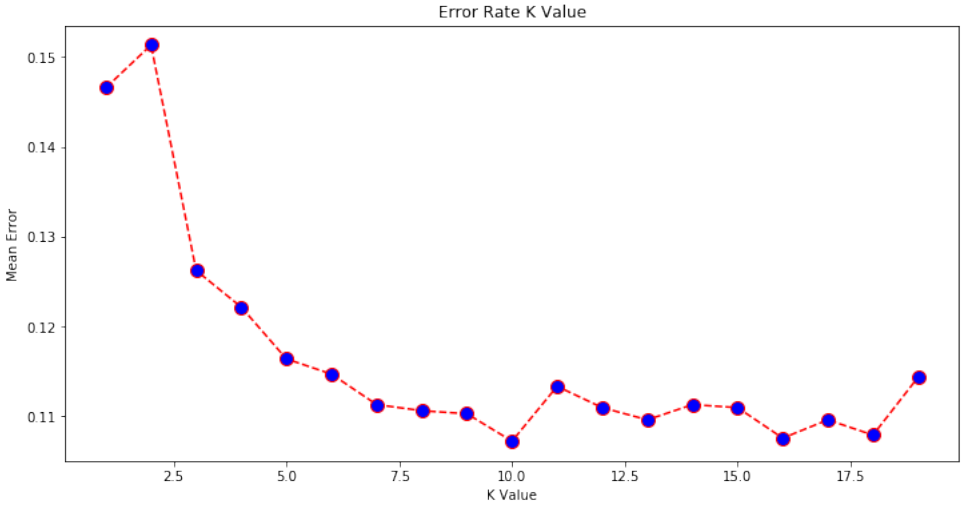


Figure 5.29: Error rate for k value

In PCA, we got 93.95% accuracy for 40 components and 93.74% accuracy for 150 components.

```
[[262   0   0   0   0   0]
 [  4 243  37   0   0   0]
 [  0  35 241   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   3 169   6]
 [  0   0   0   2   1 228]]
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       262
           1       0.87      0.86      0.86       284
           2       0.87      0.87      0.87       276
           3       0.98      1.00      0.99       240
           4       0.99      0.95      0.97       178
           5       0.97      0.99      0.98       231

    accuracy                           0.94      1471
   macro avg       0.94      0.94      0.94      1471
weighted avg       0.94      0.94      0.94      1471
```

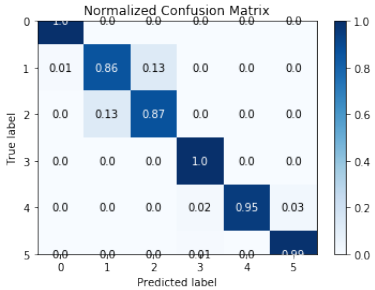


Figure 5.30: Confusion matrix for KNN by applying 40 components PCA

```
[[250   8   4   0   0   0]
 [  2 254  27   0   0   1]
 [  4  18 254   0   0   0]
 [  0   1   1 237   1   0]
 [  0   2   9   5 159   3]
 [  0   0   1   4   1 225]]
              precision    recall  f1-score   support

           0       0.98      0.95      0.97       262
           1       0.90      0.89      0.90       284
           2       0.86      0.92      0.89       276
           3       0.96      0.99      0.98       240
           4       0.99      0.89      0.94       178
           5       0.98      0.97      0.98       231

    accuracy                           0.94      1471
   macro avg       0.94      0.94      0.94      1471
weighted avg       0.94      0.94      0.94      1471
```

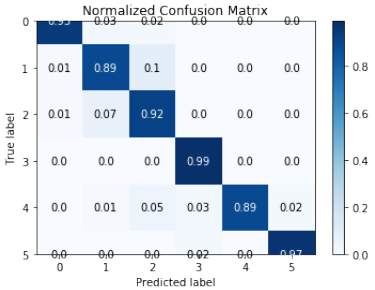


Figure 5.31: Confusion matrix for KNN by applying 150 components PCA

In SVD, we got 95.51% accuracy for 100 components.

```
[[250   7   5   0   0   0]
 [  2 250  32   0   0   0]
 [  1  15 260   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   1 176   1]
 [  0   0   0   2   0 229]]
              precision    recall  f1-score   support

           0       0.99      0.95      0.97       262
           1       0.92      0.88      0.90       284
           2       0.88      0.94      0.91       276
           3       0.99      1.00      0.99       240
           4       1.00      0.99      0.99       178
           5       1.00      0.99      0.99       231

    accuracy                           0.96      1471
   macro avg       0.96      0.96      0.96      1471
weighted avg       0.96      0.96      0.96      1471
```

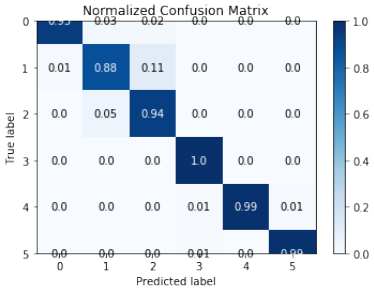


Figure 5.32: Confusion matrix for KNN by applying SVD

In LDA, we got 97.55% accuracy for 40 components.

```
[[262   0   0   0   0   0]
 [  0 263  21   0   0   0]
 [  0  14 262   0   0   0]
 [  0   0   0 240   0   0]
 [  0   0   0   0 178   0]
 [  0   0   0   0   1 230]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.95      0.93      0.94       284
           2       0.93      0.95      0.94       276
           3       1.00      1.00      1.00       240
           4       0.99      1.00      1.00       178
           5       1.00      1.00      1.00       231

    accuracy                           0.98      1471
   macro avg       0.98      0.98      0.98      1471
weighted avg       0.98      0.98      0.98      1471
```

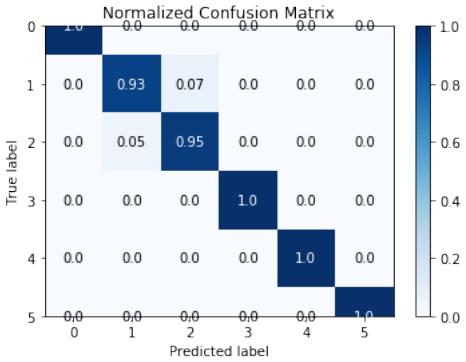


Figure 5.33: Confusion matrix for KNN by applying LDA

In ICA, we got 89.87% accuracy for 20 components and 93.67% accuracy for 150 components.

```
[[260   2   0   0   0   0]
 [  6 217  61   0   0   0]
 [  0  53 222   1   0   0]
 [  0   0   1 234   3   2]
 [  0   0   1   3 169   5]
 [  0   0   0   8   3 220]]
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       262
           1       0.80      0.76      0.78       284
           2       0.78      0.80      0.79       276
           3       0.95      0.97      0.96       240
           4       0.97      0.95      0.96       178
           5       0.97      0.95      0.96       231

    accuracy                           0.90      1471
   macro avg       0.91      0.91      0.91      1471
weighted avg       0.90      0.90      0.90      1471
```

```
[[248   7   7   0   0   0]
 [  3 255  26   0   0   0]
 [  3  19 254   0   0   0]
 [  0   2   1 236   1   0]
 [  0   1   8   5 160   4]
 [  0   0   2   3   1 225]]
              precision    recall  f1-score   support

           0       0.98      0.95      0.96       262
           1       0.90      0.90      0.90       284
           2       0.85      0.92      0.89       276
           3       0.97      0.98      0.98       240
           4       0.99      0.90      0.94       178
           5       0.98      0.97      0.98       231

    accuracy                           0.94      1471
   macro avg       0.94      0.94      0.94      1471
weighted avg       0.94      0.94      0.94      1471
```

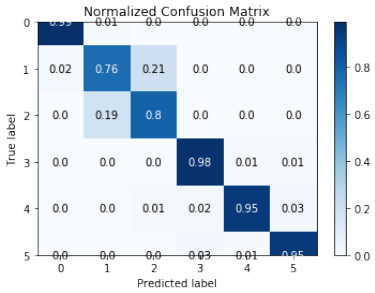


Figure 5.34: Confusion matrix for KNN by applying 20 components ICA

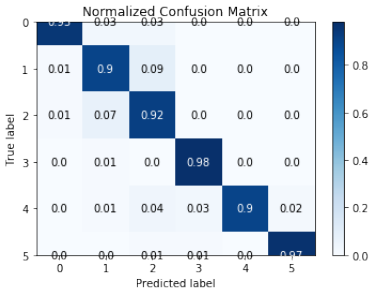


Figure 5.35: Confusion matrix for KNN by applying 150 components ICA

Lastly, in figure 5.36, the overall summarized accuracy for K-Nearest Neighbors (KNN) in both cases; where we can see that the maximum accuracy is 97.552685% which is obtained by LDA with 100 components.

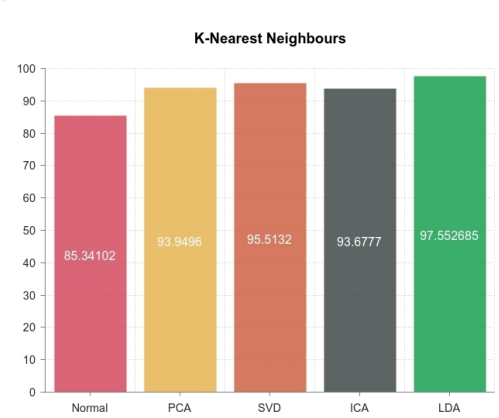


Figure 5.36: Result analysis of K-Nearest Neighbors

5.1.6 Naive Bayes classifier (NB)

We applied gaussian naive bayes classifier on independent variable X_train which has the input independent variables and dependent variable Y_train which has the numerical values from 0 to 5 for six activities. The accuracy score obtained from X_test and Y_test in naive bayes classifier is 77.0274%. he classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.58-0.96. Support values refers to 537, 491, 532, 496, 420 and 471 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively. Total 2947 occurrences of six human activities are analyzed here.

```
[[323 211   0   0   0   3]
 [  5 368 111   0   0   7]
 [  8  54 455   0   0  15]
 [  0   0   0 416  42  38]
 [  0   0   0  80 257  83]
 [  0   0   0   9  11 451]]
```

	precision	recall	f1-score	support
0	0.96	0.60	0.74	537
1	0.58	0.75	0.65	491
2	0.80	0.86	0.83	532
3	0.82	0.84	0.83	496
4	0.83	0.61	0.70	420
5	0.76	0.96	0.84	471
accuracy			0.77	2947
macro avg	0.79	0.77	0.77	2947
weighted avg	0.79	0.77	0.77	2947

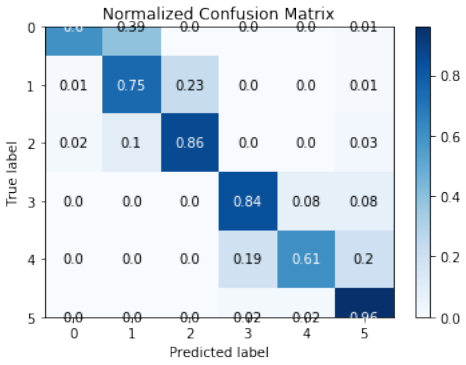


Figure 5.37: Confusion matrix for Naive Bayes

In PCA, we got 89.12% accuracy for 40 components and 86.33% accuracy for 150 components.

```
[[262   0   0   0   0   0]
 [  8 206  69   0   1   0]
 [  1  36 237   0   2   0]
 [  0   0   0 222   5  13]
 [  0   0   0   3 163  12]
 [  0   0   0   3   7 221]]
```

	precision	recall	f1-score	support
0	0.97	1.00	0.98	262
1	0.85	0.73	0.78	284
2	0.77	0.86	0.81	276
3	0.97	0.93	0.95	240
4	0.92	0.92	0.92	178
5	0.90	0.96	0.93	231
accuracy			0.89	1471
macro avg	0.90	0.90	0.90	1471
weighted avg	0.89	0.89	0.89	1471

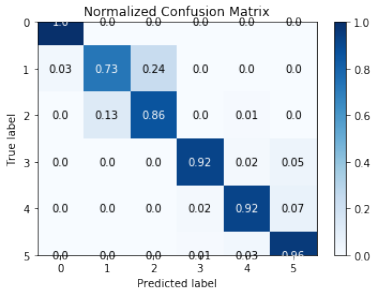


Figure 5.38: Confusion matrix for Naive Bayes by applying 40 components PCA

```
[[261   0   0   0   1   0]
 [  8 207  55   0  14   0]
 [  6  25 236   0   9   0]
 [  0   0   0 214  22   4]
 [  0   0   0  13 143  22]
 [  0   0   0   3  19 209]]
```

	precision	recall	f1-score	support
0	0.95	1.00	0.97	262
1	0.89	0.73	0.80	284
2	0.81	0.86	0.83	276
3	0.93	0.89	0.91	240
4	0.69	0.80	0.74	178
5	0.89	0.90	0.90	231
accuracy			0.86	1471
macro avg	0.86	0.86	0.86	1471
weighted avg	0.87	0.86	0.86	1471

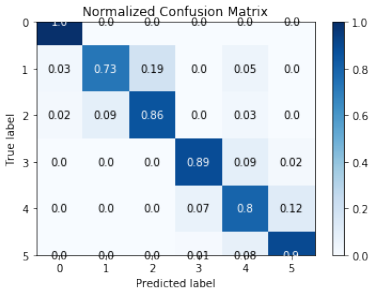


Figure 5.39: Confusion matrix for Naive Bayes by applying 150 components PCA

In SVD, for 100 components we got 88.64% accuracy.

```
[[261   0   0   0   1   0]
 [  8 219  57   0   0   0]
 [  9  29 237   0   0   1]
 [  0   0   0 219  14   7]
 [  0   0   0   8 155  15]
 [  0   0   0   3  15 213]]
```

	precision	recall	f1-score	support
0	0.94	1.00	0.97	262
1	0.88	0.77	0.82	284
2	0.81	0.86	0.83	276
3	0.95	0.91	0.93	240
4	0.84	0.87	0.85	178
5	0.90	0.92	0.91	231
accuracy			0.89	1471
macro avg	0.89	0.89	0.89	1471
weighted avg	0.89	0.89	0.89	1471

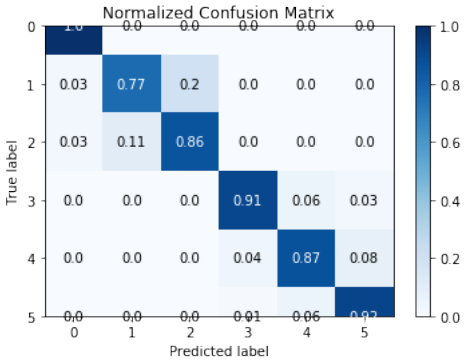


Figure 5.40: Confusion matrix for Naive Bayes by applying SVD

In LDA, for 40 components we got 98.16% accuracy.

```
[[262   0   0   0   0   0]
 [  0 271  13   0   0   0]
 [  0  13 263   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   0 178   0]
 [  0   0   0   0   0 231]]
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.95	0.95	0.95	284
2	0.95	0.95	0.95	276
3	1.00	1.00	1.00	240
4	1.00	1.00	1.00	178
5	1.00	1.00	1.00	231
accuracy			0.98	1471
macro avg	0.98	0.98	0.98	1471
weighted avg	0.98	0.98	0.98	1471

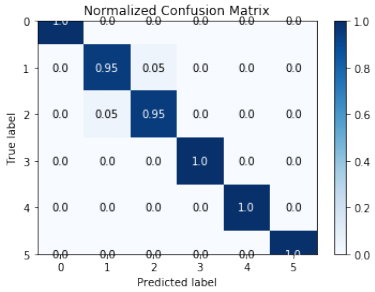


Figure 5.41: Confusion matrix for Naive Bayes by applying LDA

In ICA, we got 84.02% accuracy for 20 components and 81.91% accuracy for 150 components.

```
[[258   2   1   0   1   0]
 [ 18 191  73   0   2   0]
 [  9  49 213   0   5   0]
 [  0   0   0 213   6  21]
 [  0   0   0  11 160   7]
 [  0   0   0  18  12 201]]
```

	precision	recall	f1-score	support
0	0.91	0.98	0.94	262
1	0.79	0.67	0.73	284
2	0.74	0.77	0.76	276
3	0.88	0.89	0.88	240
4	0.86	0.90	0.88	178
5	0.88	0.87	0.87	231
accuracy			0.84	1471
macro avg	0.84	0.85	0.84	1471
weighted avg	0.84	0.84	0.84	1471

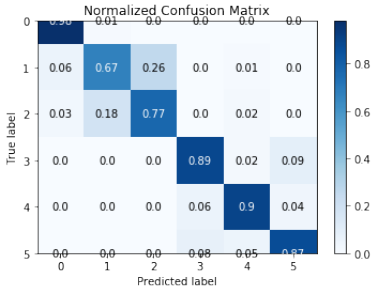


Figure 5.42: Confusion matrix for Naive Bayes by applying 20 components ICA

```
[[217   9  35   0   1   0]
 [ 20 169  93   0   2   0]
 [ 12  23 238   0   3   0]
 [  1   0   0 207  21  11]
 [  0   0   0   2 158  18]
 [  0   0   0   1  14 216]]
```

	precision	recall	f1-score	support
0	0.87	0.83	0.85	262
1	0.84	0.60	0.70	284
2	0.65	0.86	0.74	276
3	0.99	0.86	0.92	240
4	0.79	0.89	0.84	178
5	0.88	0.94	0.91	231
accuracy			0.82	1471
macro avg	0.84	0.83	0.83	1471
weighted avg	0.83	0.82	0.82	1471

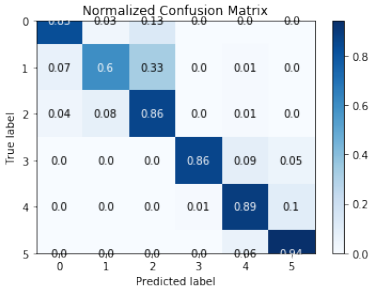


Figure 5.43: Confusion matrix for Naive Bayes by applying 150 components ICA

Lastly, the overall summarized accuracy for Naïve Bayes in both cases; where we got the highest accuracy 98.164% by applying LDA dimension reduction technique where the number of components is 40 shown in figure 5.44.

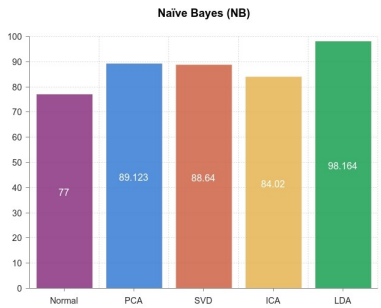


Figure 5.44: Result analysis of Naive Bayes

5.1.7 Decision Tree

In decision tree data is split into six subsets (X, Y ,X_train, Y_train, X_test, Y_test). Gini index is used to measure the uncertainty. Entropy Function and Information Gain are used as metrics to implement classifier function by ID3 (Iterative Dichotomiser 3). We got 78.18% accuracy using Gini index and 88.24% accuracy using Entropy.

```
Results Using Gini Index:
Predicted values:
[0 3 0 ... 0 3 3]
Confusion Matrix:  [[262   0   0   0   0   0]
 [  0 259  25   0   0   0]
 [  0  32 244   0   0   0]
 [  0   0   0 237   3   0]
 [  0   0   0  30 148   0]
 [  0   0   0 211  20   0]]
Accuracy :  78.17811012916384
Report :               precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.89      0.91      0.90       284
           2       0.91      0.88      0.90       276
           3       0.50      0.99      0.66       240
           4       0.87      0.83      0.85       178
           5       0.00      0.00      0.00       231

    accuracy                           0.78      1471
   macro avg       0.69      0.77      0.72      1471
weighted avg       0.71      0.78      0.73      1471
```

Figure 5.45: Confusion matrix for decision tree using gini index

```
Results Using Entropy:
Predicted values:
[0 5 0 ... 0 3 3]
Confusion Matrix:  [[261   0   0   0   0   1]
 [  0 259  25   0   0   0]
 [  0  32 244   0   0   0]
 [  0   0   0 211   4  25]
 [  0   0   0  11 135  32]
 [  0   0   0  31  12 188]]
Accuracy :  88.23929299796058
Report :               precision    recall  f1-score   support

           0       1.00      1.00      1.00       262
           1       0.89      0.91      0.90       284
           2       0.91      0.88      0.90       276
           3       0.83      0.88      0.86       240
           4       0.89      0.76      0.82       178
           5       0.76      0.81      0.79       231

    accuracy                           0.88      1471
   macro avg       0.88      0.87      0.88      1471
weighted avg       0.88      0.88      0.88      1471
```

Figure 5.46: Confusion matrix for decision tree using entropy

5.1.8 Multilayer perceptron (MLP)

As we have 561 features in the existing dataset regarding six daily activities of human , MLP algorithm is one of the most effective options among the supervised learning algorithms for classifying our data. Data are taken as real world estimations by the model which is convenient approach for classification and regression. Moderation of interdependencies among the data parameters leads to us to least percentage of error. The features are given in the input layers and we obtained 98.57% accuracy in the output layer or visible layer, along with the prediction of the designated activity. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.96-1.00. Support values refers to 262,284,276,240,178,231 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively. Total 1471 occurrences of six human activities are analyzed here.

98.57239972807615 %

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.96	0.98	0.97	284
2	0.98	0.96	0.97	276
3	1.00	1.00	1.00	240
4	1.00	0.99	1.00	178
5	0.99	1.00	1.00	231
micro avg	0.99	0.99	0.99	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.99	0.99	0.99	1471
samples avg	0.99	0.99	0.99	1471

```
[[262   0   0   0   0   0]
 [  1 277   6   0   0   0]
 [  0  12 264   0   0   0]
 [  0   0   0 239   0   1]
 [  0   0   0   0 177   1]
 [  0   0   0   0   0 231]]
```

Figure 5.47: Confusion matrix for Multilayer perceptron (MLP)

Regarding loss function graph for test and train datasets we have the following results:

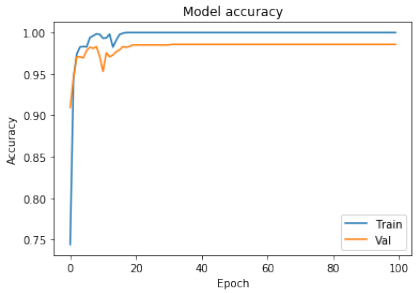


Figure 5.48: Accuracy plot for MLP training set

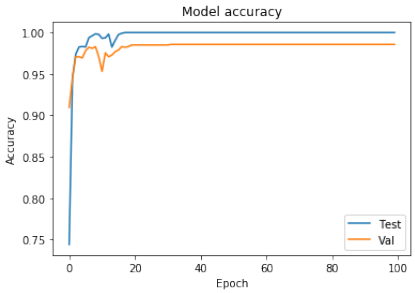


Figure 5.49: Accuracy plot for MLP testing set

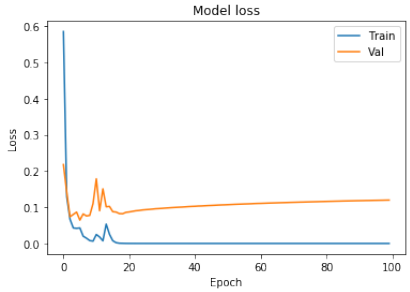


Figure 5.50: Loss plot for MLP training set

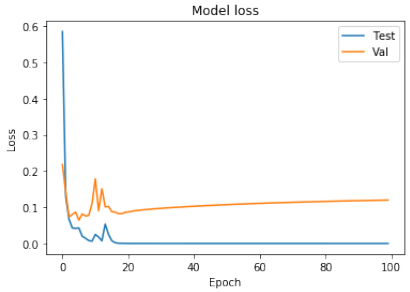


Figure 5.51: Loss plot for MLP testing set

In PCA, we got 98.70% accuracy for 150 components. Visualization of loss function result for PCA has straight lines for both train and value. Its because of the reduction technique of PCA which changes the interdependencies between data parameters and gives less error than modeling of original dataset.

98.77634262406526 %

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.97	0.99	0.98	284
2	0.98	0.96	0.97	276
3	1.00	1.00	1.00	240
4	1.00	0.99	0.99	178
5	0.99	1.00	1.00	231
micro avg	0.99	0.99	0.99	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.99	0.99	0.99	1471
samples avg	0.99	0.99	0.99	1471

```
[[262   0   0   0   0   0]
 [  0 280   4   0   0   0]
 [  0  10 265   0   0   1]
 [  0   0   0 239   0   1]
 [  0   0   1   1 176   0]
 [  0   0   0   0   0 231]]
```

Figure 5.52: Confusion matrix for Multilayer perceptron (MLP) for 150 components of PCA

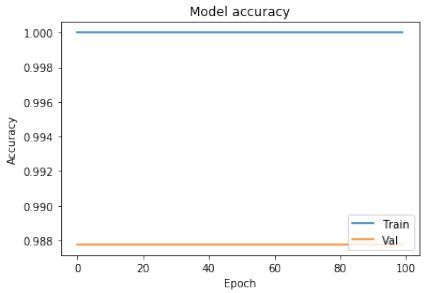


Figure 5.53: Accuracy plot for MLP 150 PCA components training set

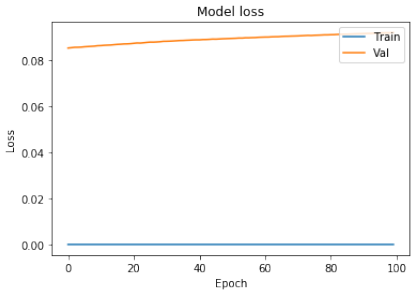


Figure 5.54: Loss plot for MLP 150 PCA components training set

In SVD, we have 98.64% accuracy for 100 components. Here, Loss function graph also gives straight lines for both train and value like PCA as its also have interdependency moderation mechanism for reduction formula.

98.64038069340585 %

	precision	recall	f1-score	support
0	1.00	1.00	1.00	262
1	0.97	0.98	0.97	284
2	0.98	0.97	0.97	276
3	0.99	0.99	0.99	240
4	0.99	0.99	0.99	178
5	1.00	0.99	0.99	231
micro avg	0.99	0.99	0.99	1471
macro avg	0.99	0.99	0.99	1471
weighted avg	0.99	0.99	0.99	1471
samples avg	0.99	0.99	0.99	1471

```
[[262   0   0   0   0   0]
 [  0 278   6   0   0   0]
 [  0   9 267   0   0   0]
 [  1   0   0 238   0   1]
 [  0   0   0   1 177   0]
 [  0   0   0   1   1 229]]
```

Figure 5.55: Confusion matrix for Multilayer perceptron (MLP) of SVD

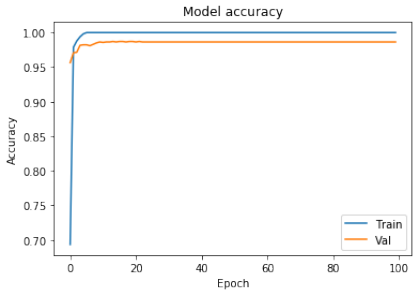


Figure 5.56: Accuracy plot for MLP (SVD) training set

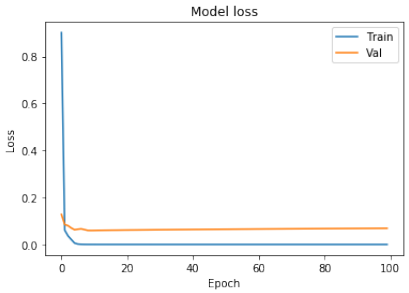


Figure 5.57: Loss plot for MLP (SVD) training set

5.1.9 Long short-term memory (LSTM)

Obtained accuracy in LSTM is 90.70%. It can be increased based on how the neural network weights got initialized at the beginning. Neural networks are quite good in accurate movement identification of humans. They used the device on the waist in the test and each classification sequence has only a 128 sample window with two internal sensors (a.k.a. 2.56 seconds at 50 FPS), so these predictions are extremely accurate considering this limited context window and raw data. there is no major bug, and the community tried to implement this code a lot. Expecting good results for guessing between the "SITTING" and "STANDING" labels is not as easy as those tasks appear to be almost the same from a waist-mounted device's point of view depending on how the original data set was collected. It can also be seen that the difference between "WALKING," "WALKING UPSTAIRS" and "WALKING DOWNSTAIRS" was somewhat difficult to make. Clearly, in terms of gestures, these behaviors are quite similar.

```
Testing Accuracy: 90.7024085521698%

Precision: 91.06470687433304%
Recall: 90.70240922972515%
f1_score: 90.6678839061672%

Confusion Matrix:
[[447  12  37   0   0   0]
 [ 12 453   4   1   1   0]
 [  2   1 417   0   0   0]
 [  0  24   0 428  39   0]
 [  6   0   0 118 408   0]
 [  0  17   0   0   0 520]]

Confusion matrix (normalised to % of total test data):
[[15.167968    0.40719375  1.2555141   0.          0.          0.        ]
 [ 0.40719375 15.371564    0.13573125  0.03393281  0.03393281  0.        ]
 [ 0.06786563  0.03393281 14.149983    0.          0.          0.        ]
 [ 0.          0.8143875   0.         14.523244    1.3233796   0.        ]
 [ 0.20359688  0.          0.          4.004072   13.844588    0.        ]
 [ 0.          0.5768578   0.          0.          0.         17.645063  ]]
Note: training and testing data is not equally distributed amongst classes,
so it is normal that more than a 6th of the data is correctly classifier in the last category.
```

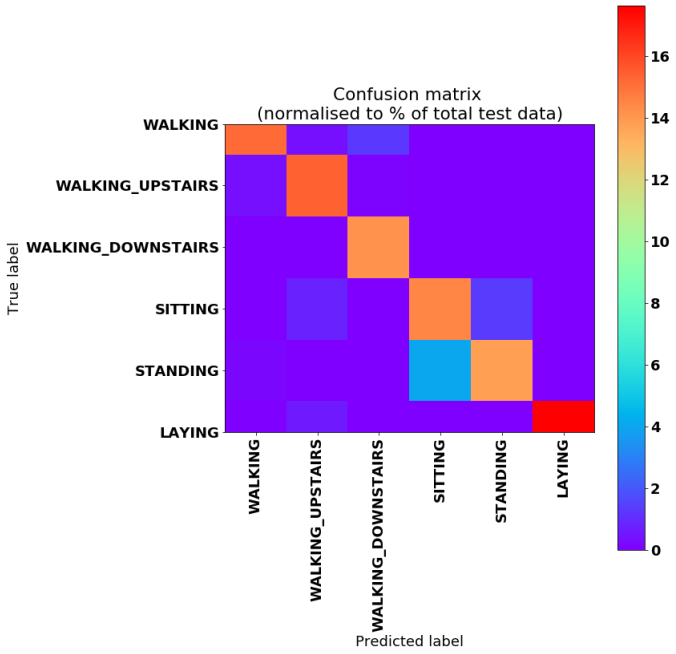


Figure 5.58: Confusion matrix for LSTM

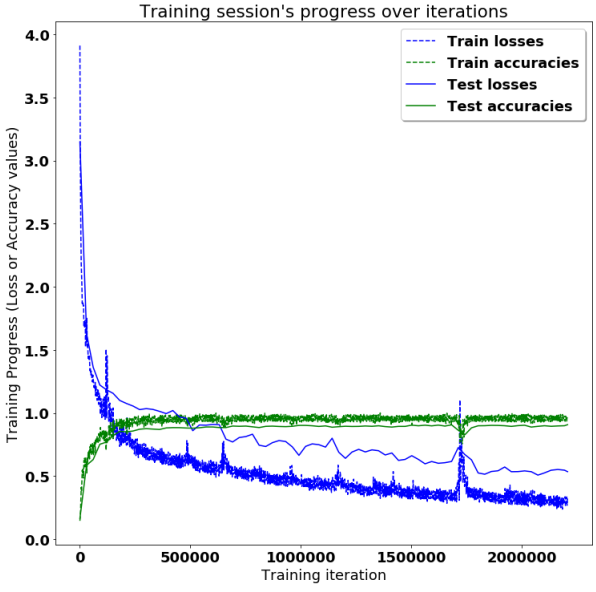


Figure 5.59: Training session's progress over iterations for LSTM

5.2 New Dataset

Here we have used three machine learning classifier to analyse the new dataset. the results are given below-

5.2.1 Random Forest Classifier

Here in random forest classifier we used imputation method to detect null values. Then after applying transform method on X_test we get the X_test_imp variable on which we applied random forest classifier and got the y_predicted variable. Thus we got 43.9% accuracy by fitting classifier on y_predicted and Y_test variables. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.25-0.78. Support values refers to 36,40,43,42,44 and 41 occurrences for standing, sitting, laying, walking, downstairs and upstairs activities respectively for 246 occurrences. The accuracy score, confusion matrix, classification report and plot for normalized confusion matrix for logistic regression are given below:

```
43.9 %
[[28  0  1  4  1  2]
 [ 8 17  4  1  5  5]
 [ 3  4 21  7  5  3]
 [ 2  2  4 20 10  4]
 [ 3  7  6  8 11  9]
 [ 4  4  4 10  8 11]]
              precision    recall  f1-score   support

           0       0.58      0.78      0.67        36
           1       0.50      0.42      0.46        40
           2       0.53      0.49      0.51        43
           3       0.40      0.48      0.43        42
           4       0.28      0.25      0.26        44
           5       0.32      0.27      0.29        41

    accuracy                           0.44       246
   macro avg       0.43      0.45      0.44       246
weighted avg       0.43      0.44      0.43       246
```

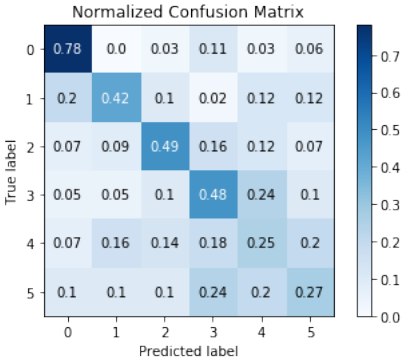


Figure 5.60: Confusion matrix for Random forest classifier

5.2.2 K-Nearest Neighbors(KNN)

In KNN classifier, at first we scaled X train and X test data with all the features of the input.Then we applied KNN classifier on scaled X_train values and Y_train data which contains encoded variables of activities. So we got 36.99% accuracy from the variable Y_test and Y_pred where Y_pred is obtained from X test variable scaled results.The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.28-0.57 for total 246 occurrences of six human activities are analyzed here.

```
[[20  8  1  1  5  1]
 [ 3 13  5  8  5  6]
 [ 3  9 18  5  3  5]
 [ 2  5  4 17  8  6]
 [ 3  3  5  7 11 15]
 [ 4  2  2 11 10 12]]
              precision    recall  f1-score   support

           0       0.57      0.56      0.56        36
           1       0.33      0.33      0.33        40
           2       0.51      0.42      0.46        43
           3       0.35      0.40      0.37        42
           4       0.26      0.25      0.26        44
           5       0.27      0.29      0.28        41

    accuracy                           0.37       246
   macro avg       0.38      0.37      0.38       246
weighted avg       0.38      0.37      0.37       246
```

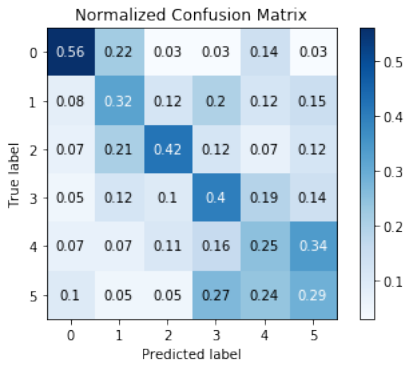


Figure 5.61: Confusion matrix for KNN

In the Error rate of K value graph in figure 5.62, we can see that we have the least error while the K value is 5 whereas we have the maximum error while the K value is 7.5 while number of components is 20.

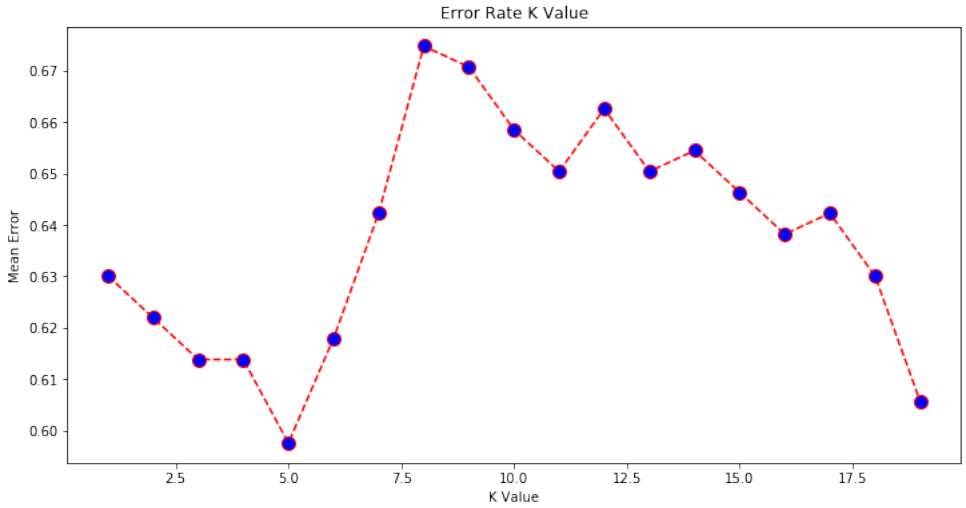


Figure 5.62: Error for K value in KNN

5.2.3 Support Vector Machine(SVM)

In SVM , after scalling the X_train and X_test we got the gamma parameter's value is 0.001 which indicates that influence of single training set is far. Value of C parameter is 1000 which indicates that there remains smaller margin and the decision function is better at classifying all training points accurately. Accuracy for training and test data are 35.27% and 36.17%. The classification report contains precision, recall, f1-score and support values for six activities. Precision, recall and f1-score all values are between 0.0-.83 for 246 occurrences of six human activities.

```
[13  5  3 13  0  6]
[ 6  4 12 16  0  5]
[ 4  2  2 29  0  5]
[ 7  3  2 21  0 11]
[ 6  1  3 18  0 13]]
```

	precision	recall	f1-score	support
0	0.45	0.83	0.59	36
1	0.33	0.12	0.18	40
2	0.55	0.28	0.37	43
3	0.28	0.69	0.40	42
4	0.00	0.00	0.00	44
5	0.32	0.32	0.32	41
accuracy			0.36	246
macro avg	0.32	0.37	0.31	246
weighted avg	0.32	0.36	0.30	246

	precision	recall	f1-score	support
0	0.45	0.83	0.59	36
1	0.33	0.12	0.18	40
2	0.55	0.28	0.37	43
3	0.28	0.69	0.40	42
4	0.00	0.00	0.00	44
5	0.32	0.32	0.32	41
accuracy			0.36	246
macro avg	0.32	0.37	0.31	246
weighted avg	0.32	0.36	0.30	246

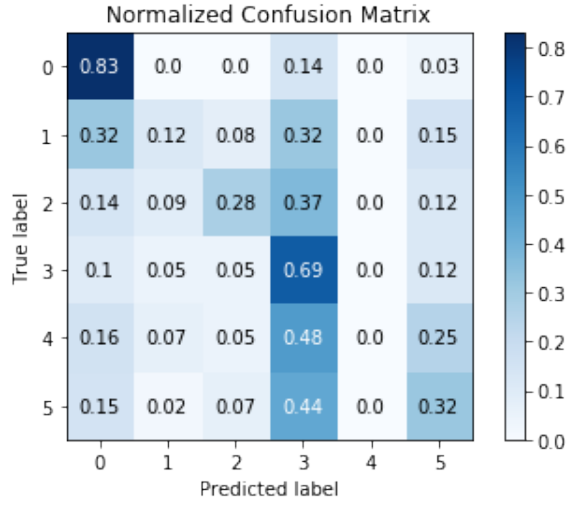


Figure 5.63: Confusion matrix for SVM

Chapter 6

Discussion

We exploited several works on HAR focused on machine learning and deep learning. We observed a huge usage of SVM for multiclass problems of HAR. The paper which mainly developed and classified the existing dataset we have used, has used SVM and obtained 96% accuracy for predicting six human activities[29]. In another research, for sitting, walking, jogging, and going upstairs and downstairs they used KNN classifier and obtained 52.3%–79.4% for up and down stair walking, 91.7% for jogging, 90.1%–94.1% for walking on a level ground, and 100% for sitting[35]. Smartphone is the most widely used device in almost all the research regarding human activity recognition which assists in recording activity data conveniently with its sensors. Another research also focused on activities like Sitting Standing Walking Upstair Downstair and Lying using smartphone sensors and for classification purpose they used SVM which gave 89.59% accuracy[32].
At first we tried to observe the accuracy of machine learning classifiers and neural network by fitting them on existing dataset without applying dimension reduction techniques on it. We applied 9 algorithms in total (Linear and logistic regression,SVM,KNN,Random forest classifier,naive bayes classifier,decision tree,LSTM and MLP). Among them we got maximum accuracy of 99% for SVM and minimum accuracy of 77% for naive bayes classifier. SVM works well for high dimensional spaces as it places clear seperation margin among multiple features. The existing dataset deals with higher number of features of six activities among which SVM creates the hyperplan of classification. That's why we got maximum accuracy for SVM. In Addition to, MLP gives the second highest accuracy of 98%. It is an efficient algorithm of neural network for gesture recognition. Therefore after SVM, MLP gives the second best performance for activity prediction. On the contrary, naive bayes gives minimum accuracy as shown in figure 6.1 because it estimates small train set properly but when it comes large set of train set containing real-time independent variables it is almost impossible to obtain predictors.

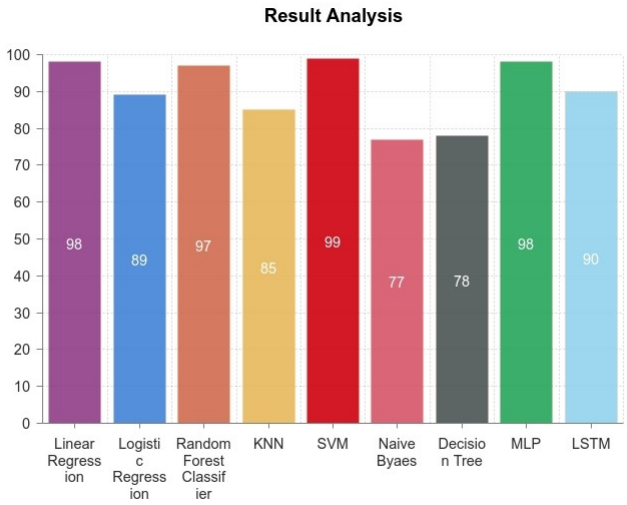


Figure 6.1: Result Analysis Graph

In PCA we got 100% accuracy for SVM where number of components was 150. Here the lowest accuracy is 89% which is obtained in naive bayes classifier for 40 components. As SVM is the most efficient algorithm for multiclass modeling it gives the maximum accuracy for PCA as well. The dataset contains 561 features, among which we took 150 principle components which can describe the dataset properly. That's why, for 150 components we get highest accuracy in SVM as shown in figure 6.2. On the other hand, naive bayes gives lowest accuracy for 40 components because of its prediction limitation for huge set of independent features which we have in the dataset.

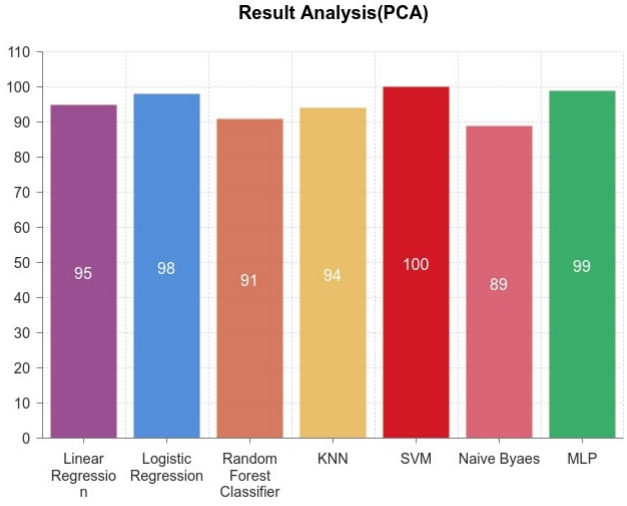


Figure 6.2: Result Analysis Graph for PCA

From figure 6.3, we can see that, in SVD highest accuracy is 100% for SVM and lowest accuracy is 89% for naive bayes classifier. For both cases number of components was 100.

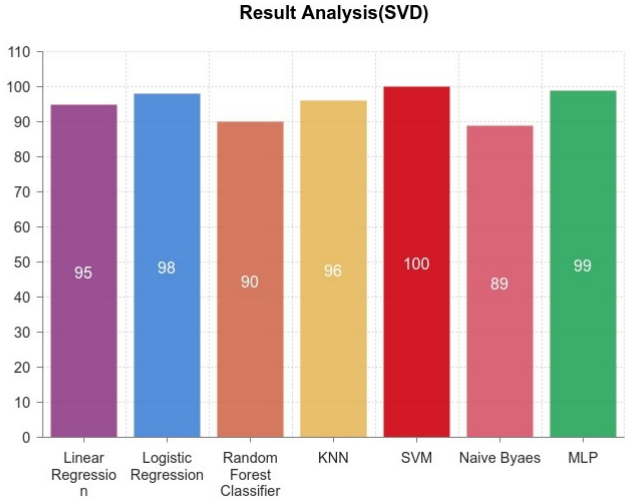


Figure 6.3: Result Analysis Graph for SVD

From figure 6.4, we can see that, in LDA, for 40 components, maximum accuracy is 99% and minimum accuracy is 97% which are observed for SVM and Linear Regression respectively.

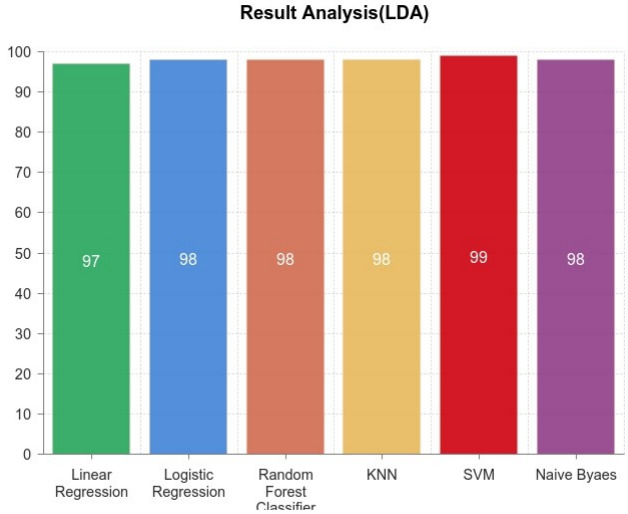


Figure 6.4: Result Analysis Graph for LDA

Figure 6.5 illustrates that, ICA we got highest accuracy 99.7% and lowest accuracy 84%, which are for SVM and Naive bayes classifier. For SVM we got same result for 20 and 150 components. But for naive bayes number of components was 20.

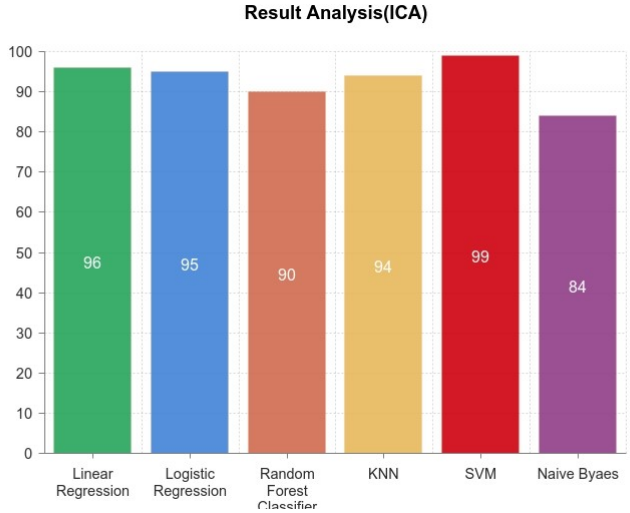


Figure 6.5: Result Analysis Graph for ICA

Table 6.1 given below, shows the summarized observations of all accuracy of every classifiers that we have used so far. This observation table also includes the accuracy of classifiers with applying dimension reduction techniques.

Algorithms	Normal	PCA(40)	PCA(150)	SVD	LDA	ICA(20)	ICA(150)
Linear Regression	98.470%	91.86877%	95.2327%	94.816%	96.887%	90.5236%	95.28046%
Logistic Regression	96.199%	93.5418%	98.2324%	97.8246%	98.096%	83.140%	94.7654%
Random-forest Classifier	97.42%	90.75%	89.6%	89.6%	98.03%	75.66%	89.53%
KNN	85.34%	93.94%	93.74%	95.51%	97.55%	89.81%	93.67%
SVM	99.68%	99.28%	100%	100%	98.64%	98.43%	99.79%
Naive Bayes	77%	89.12%	86.33%	88.64%	98.164%	84.02%	81.91%
MLP	98.57%	–	98.70%	98.64%	–	–	–
Decision Tree	78.18%	–	–	–	–	–	–
LSTM	90.70%	–	–	–	–	–	–

Table 6.1: List of accuracies of all classifiers

On the other hand, for new dataset, we have used three machine learning algorithms which are Random-forest classifier, SVM and KNN. In SVM , the accuracy result is 35.27% for training set and 36.17% for testing set. The accuracy is lower because the new dataset has less number of features but SVM works well for large number of dimensions where dimensions are greater than samples. But in new dataset we have less number of dimensions than number of samples as we did not calculate any other features from the x,y,z values of accelerometer and gyroscope. In KNN, we got 36.99% accuracy and in random forest classifier accuracy is 43.9%. By analyzing these three algorithms we can find that random forest classifier is the best fit classifier for new dataset. Random forest classifier gives a higher performance where one needs less interpretation; as we have less number of dimensions in the new dataset there is less need of interpretation here. Therefore, random forest classifier gives maximum accuracy for the new dataset. Figue 6.6 shows the comparative analysis of results for new dataset.

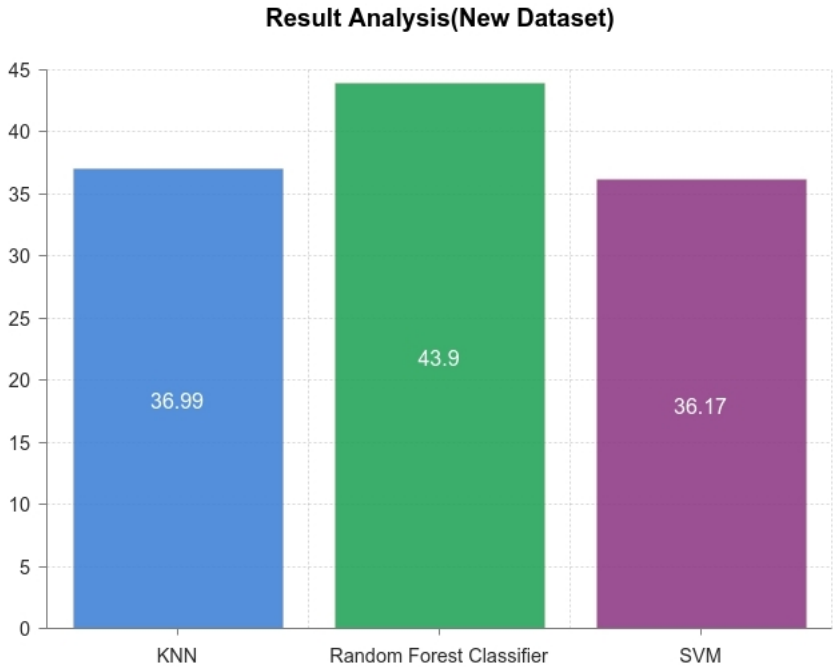


Figure 6.6: Result Analysis Graph for new dataset

Chapter 7

Conclusion & Future works

Our research focused on classifying six human activities laying,sitting,standing,walking, walking downstairs and walking upstairs using machine learning and neural network. We worked on a publicly available existing dataset which has 561 features. Our main objective was deducting dimension of these huge dataset using several techniques and observing the accuracies and thus declaring which algorithm gives the best result in predicting human activity for which reduction technique. We used four dimension reduction techniques which are PCA, LDA, ICA and SVD. Here we applied machine learning algorithms linear regression, logistic regression, SVM, KNN, naive bayes classifier, decision tree and random forest classifier. Along with these we used LSTM and MLP neural network algorithms.After applying the algorithms we found that SVM is the best fit algorithm for predicting human activity in all section. Apart from these works, we started to develop a new dataset using a waist mounted device. We made the device using accelerometer and gyroscope sensors and also used clock module and memory card module for storing the data according to their designated time. In our dataset we just applied SVM, KNN and random forest classifier and could not obtain noticeable result as we could not make it as greater as other datasets. But in future we want to extend our dataset in such a way so that we can predict elderly persons activity and build a health monitoring system for them.

one of the main limitations of our research is the size of our new Dataset. As we have worked with two dataset; one is existing dataset which we have collected from uci machine learning repository and another one is our new dataset in which we collected data through a waist-mounted device. In our device we have used smartphone sensors accelerometer and gyroscope. We have collected data of 30 people aged Between 18-60. We just started to build up the dataset at our homely environment. Waist mounted device is easy to record the daily human activities like laying, sitting, standing, walking, walking downstairs and upstairs. In our dataset there are six features; the triaxial records (x,y,z) of accelerometer and gyroscope. This amount of observations are really inadequate for an in depth research. We have seen that in the existing dataset there are 561 features among them 6 were the x,y,z values of smartphone sensors. The rest of the Values are obtained by applying mathematical formulas such as fast fourier analysis, euclidean method.they have extracted these huge number of features for observing the variation of the data for each activity and also the correlations among the features. We must say, this is publicly available a rich dataset to predict human activities. On the contrary in our

dataset we have not these much features which are not sufficient to fit on a machine learning model and obtain desired accuracy along with the prediction of activities. With these least amount of data we have got accuracies within 45% for svm, knn, random-forest classifiers which are really poor results.
In future, we are hoping to focus our research on predicting elderly persons Daily activities. Firstly, we will need to extend our dataset by collecting more data from old age homes of the city. After collecting raw data we will be also extracting more features from the data. Our target is to monitor old persons activities and thus assisting them during their illness especially to those who do not live with their family. In that case we need to enhance our device features too by adding more sensors for more efficient data needed to monitor elder citizens. We may choose sonar sensor and pir sensor for monitoring the elder person while they will be in washroom. In these days wrist band can measure heart rate blood pressure footsteps and also can be connected to mobile phones for notification purpose. But the ones who are old and homeless cannot afford wrist bands. That's why we want to develop a health monitoring system for the deprived senior citizens. Our motto is to monitor their activities continuously and if anything seems abnormal notify the managing section of the old age homes. The device will be along with their waist always. We will be trying to develop it as comfortable as possible. Our device will be also available to the senior members of the family whose family members work outside of home all day long. In this situation if anything unusual happened their family members will be notified through a Mobile application synchronised with our device. This real time data will help to enrich our dataset and extraction of other features by applying compatible mathematical formulas will make it eligible for modeling with necessary classifiers. Thus we want to enhance our research on health monitoring of senior citizens of our country and also build up a better version of our current dataset to predict their daily activities so that one day our dataset will be also a model for prediction of human activities.

Bibliography

[1] T. Iso and K. Yamazaki, "Gait analyzer based on a cell phone with a single three-axis accelerometer", in *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, ACM, 2006, pp. 141–144.

[2] I. Anderson, J. Maitland, S. Sherwood, L. Barkhuus, M. Chalmers, M. Hall, B. Brown, and H. Muller, "Shakra: Tracking and sharing daily activity levels with unaugmented mobile phones", *Mobile networks and applications*, vol. 12, no. 2-3, pp. 185–199, 2007.

[3] M. Berchtold, M. Budde, D. Gordon, H. R. Schmidtke, and M. Beigl, "Actiserv: Activity recognition service for mobile phones", in *International Symposium on Wearable Computers (ISWC) 2010*, IEEE, 2010, pp. 1–8.

[4] E. Miluzzo, C. T. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu, and A. T. Campbell, "Darwin phones: The evolution of sensing and inference on mobile phones", in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, ACM, 2010, pp. 5–20.

[5] T. Saponas, J. Lester, J. Froehlich, J. Fogarty, and J. Landay, "Ilearn on the iphone: Real-time human activity classification on commodity mobile phones", *University of Washington CSE Tech Report UW-CSE-08-04-02*, vol. 2008, 2008.

[6] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "Soundsense: Scalable sound sensing for people-centric applications on mobile phones", in *Proceedings of the 7th international conference on Mobile systems, applications, and services*, ACM, 2009, pp. 165–178.

[7] J. Fontecha, F. J. Navarro, R. Hervás, and J. Bravo, "Elderly frailty detection by using accelerometer-enabled smartphones and clinical information records", *Personal and ubiquitous computing*, vol. 17, no. 6, pp. 1073–1083, 2013.

[8] J. Ryder, B. Longstaff, S. Reddy, and D. Estrin, "Ambulation: A tool for monitoring mobility patterns over time using mobile phones", in *2009 International Conference on Computational Science and Engineering*, IEEE, vol. 4, 2009, pp. 927–931.

[9] S. Purpura, V. Schwanda, K. Williams, W. Stubler, and P. Sengers, "Fit4life: The design of a persuasive technology promoting healthy behavior and ideal weight", in *Proceedings of the SIGCHI conference on human factors in computing systems*, ACM, 2011, pp. 423–432.

[10] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The jigsaw continuous sensing engine for mobile phone applications", in *Proceedings of the 8th ACM conference on embedded networked sensor systems*, ACM, 2010, pp. 71–84.

[11] S. Zhang, P. McCullagh, C. Nugent, and H. Zheng, "Activity monitoring using a smart phone's accelerometer with hierarchical classification", in *2010 Sixth International Conference on Intelligent Environments*, IEEE, 2010, pp. 158–163.

[12] G. Bieber, P. Koldrack, C. Sablowski, C. Peter, and B. Urban, "Mobile physical activity recognition of stand-up and sit-down transitions for user behavior analysis", in *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, ACM, 2010, p. 50.

[13] J. Yang, "Toward physical activity diary: Motion recognition using simple acceleration features with mobile phones", in *Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics*, ACM, 2009, pp. 1–10.

[14] A. Henpraserttae, S. Thiemjarus, and S. Marukatat, "Accurate activity recognition using a mobile phone regardless of device orientation and location", in *2011 International Conference on Body Sensor Networks*, IEEE, 2011, pp. 41–46.

[15] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers", *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[16] S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, and D. J. Cook, "Simple and complex activity recognition through smart phones", in *2012 Eighth International Conference on Intelligent Environments*, IEEE, 2012, pp. 214–221.

[17] A. M. Khan, A. Tufail, A. M. Khattak, and T. H. Laine, "Activity recognition on smartphones via sensor-fusion and kda-based svms", *International Journal of Distributed Sensor Networks*, vol. 10, no. 5, p. 503 291, 2014.

[18] B. Das, A. M. Seelye, B. L. Thomas, D. J. Cook, L. B. Holder, and M. Schmitter-Edgecombe, "Using smart phones for context-aware prompting in smart environments", in *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, IEEE, 2012, pp. 399–403.

[19] J. Guiry, P. Van de Ven, and J. Nelson, "Multi-sensor fusion for enhanced contextual awareness of everyday activities with ubiquitous devices", *Sensors*, vol. 14, no. 3, pp. 5687–5701, 2014.

[20] A. Khan, M. Siddiqi, and S.-W. Lee, "Exploratory data analysis of acceleration signals to select light-weight and accurate features for real-time activity recognition on smartphones", *Sensors*, vol. 13, no. 10, pp. 13 099–13 122, 2013.

[21] O. Banos, J.-M. Galvez, M. Damas, H. Pomares, and I. Rojas, "Window size impact in human activity recognition", *Sensors*, vol. 14, no. 4, pp. 6474–6499, 2014.

[22] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep activity recognition models with triaxial accelerometers", *Proceedings of the Computer Vision and Pattern Recognition, Honolulu, HI, USA*, pp. 21–26, 2017.

[23] Z. S. Abdallah, M. M. Gaber, B. Srinivasan, and S. Krishnaswamy, "Anynovel: Detection of novel concepts in evolving data streams", *Evolving Systems*, vol. 7, no. 2, pp. 73–93, 2016.

[24] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones", *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[25] M. Shoaib, S. Bosch, O. Incel, H. Scholten, and P. Havinga, "Fusion of smartphone motion sensors for physical activity recognition", *Sensors*, vol. 14, no. 6, pp. 10 146–10 176, 2014.

[26] W. Sousa, E. Souto, J. Rodrigres, P. Sadarc, R. Jalali, and K. El-Khatib, "A comparative analysis of the impact of features on human activity recognition with smartphone sensors", in *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, ACM, 2017, pp. 397–404.

[27] A. Wang, G. Chen, J. Yang, S. Zhao, and C.-Y. Chang, "A comparative study on human activity recognition using inertial sensors in a smartphone", *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4566–4578, 2016.

[28] W. Sousa Lima, E. Souto, K. El-Khatib, R. Jalali, and J. Gama, "Human activity recognition using inertial sensors in a smartphone: An overview", *Sensors*, vol. 19, no. 14, p. 3213, 2019.

[29] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones.", in *Esann*, 2013.

[30] L. Cheng, Y. Guan, K. Zhu, and Y. Li, "Recognition of human activities using machine learning methods with wearable sensors", in *2017 IEEE 7th annual computing and communication workshop and conference (CCWC)*, IEEE, 2017, pp. 1–7.

[31] H. Gammulle, S. Denman, S. Sridharan, and C. Fookes, "Two stream lstm: A deep fusion framework for human action recognition", in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2017, pp. 177–186.

[32] D. N. Tran and D. D. Phan, "Human activities recognition in android smartphone using support vector machine", in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, IEEE, 2016, pp. 64–68.

[33] N. Çelenli, K. N. Seviş, M. F. Esgin, K. Altundağ, and U. Uludağ, "An unconstrained activity recognition method using smart phones", in *2014 International Conference of the Biometrics Special Interest Group (BIOSIG)*, IEEE, 2014, pp. 1–7.

[34] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data", in *Aaai*, vol. 5, 2005, pp. 1541–1546.

[35] W. Wu, S. Dasgupta, E. E. Ramirez, C. Peterson, and G. J. Norman, "Classification accuracies of physical activities using smartphone motion sensors", *Journal of medical Internet research*, vol. 14, no. 5, e130, 2012.

[36] X. Su, H. Tong, and P. Ji, "Activity recognition with smartphone sensors", *Tsinghua science and technology*, vol. 19, no. 3, pp. 235–249, 2014.

[37] A. M. Khan, Y.-K. Lee, S.-Y. Lee, and T.-S. Kim, "Human activity recognition via an accelerometer-enabled-smartphone using kernel discriminant analysis", in *2010 5th international conference on future information technology*, IEEE, 2010, pp. 1–6.

[38] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine", in *International workshop on ambient assisted living*, Springer, 2012, pp. 216–223.

[39] Y.-J. Hong, I.-J. Kim, S. C. Ahn, and H.-G. Kim, "Activity recognition using wearable sensors for elder care", in *2008 Second International Conference on Future Generation Communication and Networking*, IEEE, vol. 2, 2008, pp. 302–305.

[40] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher, "Activity recognition and monitoring using multiple sensors on different body positions", CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, Tech. Rep., 2006.

[41] U. Maurer, A. Rowe, A. Smailagic, and D. P. Siewiorek, "Ewatch: A wearable sensor and notification platform.", in *BSN*, vol. 6, 2006, pp. 142–145.

[42] S. Kozina, H. Gjoreski, M. Gams, and M. Luštrek, "Efficient activity recognition and fall detection using accelerometers", in *International Competition on Evaluating AAL Systems through Competitive Benchmarking*, Springer, 2013, pp. 13–23.

[43] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, and B. G. Celler, "Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring", *IEEE transactions on information technology in biomedicine*, vol. 10, no. 1, pp. 156–167, 2006.

[44] *Introduction to machine learning algorithms: Linear regression*, https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a, (Accessed on 12/07/2019).

[45] *Introduction to machine learning algorithms: Logistic regression - by*, https://hackernoon.com/introduction-to-machine-learning-algorithms-logistic-regression-cbdd82d81a36, (Accessed on 12/07/2019).

[46] *Naive bayes in machine learning - towards data science*, https://towardsdatascience.com/naive-bayes-in-machine-learning-f49cc8f831b4, (Accessed on 12/07/2019).

[47] *Support vector machine — introduction to machine learning algorithms*, https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47, (Accessed on 12/07/2019).

[48] *Introduction to knn, k-nearest neighbors : Simplified*, https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/, (Accessed on 12/07/2019).

[49] *Essentials of deep learning : Introduction to long short term memory*, https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/, (Accessed on 12/07/2019).

[50] *Introduction to dimensionality reduction - geeksforgeeks*, https://www.geeksforgeeks.org/dimensionality-reduction/, (Accessed on 12/07/2019).

[51] *Practical guide to principal component analysis (pca) in r & python*, https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/, (Accessed on 12/07/2019).

[52] *Using svd for dimensionality reduction — oracle r technologies blog*, https://blogs.oracle.com/r/using-svd-for-dimensionality-reduction, (Accessed on 12/07/2019).