HUMAN ACTIVITY RECOGNITION USING SMARTPHONE DATA

Thesis Submitted in Partial Fulfillment of Requirements for the Award of the Degree of

Master of Technology in

INFORMATION SYSTEMS

SUBMITTED BY

SAMEER DUBEY

(2K15/ISY/17)

UNDER THE GUIDANCE OF

MANOJ KUMAR

ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI-110092

(2015-2017)

CERTIFICATE

This is to certify that **Mr. SAMEER DUBEY (2K15/ISY/17)** has carried out the major project titled "**Human Activity Recognition using smartphone data**" as a partial requirement for the award of **Master of Technology** degree in **Information System** by **Delhi Technological University, Delhi.**

The Major project is a bonafide piece of work carried out and completed under my supervision and guidance during the academic session 2015-2017. The Matter contained in this thesis has not been submitted elsewhere for the award of any other degree.

Date:

(Project guide)
Mr. Manoj Kumar
Associate Professor
Department of Computer Science and Engineering
Delhi Technological University

ACKNOWLEDGEMENT

I express my gratitude to my major project guide **Mr. Manoj Kumar, Associate Professor in Department of Computer Science and Engineering at Delhi Technological University, Delhi** for the valuable support and guidance he provided in making this major project. It is my pleasure to record my sincere thanks to my respected guide for his constructive criticism, interminable encouragement and valuable insight without which the project would not have shaped as it has. I humbly extend my word of gratitude to Dr. Kapil Sharma, Head of Department and all the other faculty members and staff of department for providing their valuable help, time and facilities at the need of hour.

SAMEER DUBEY

Roll No.: 2K15/ISY/17

M.Tech(Information System)

Department of Information Technology

Delhi Technological University, Delhi

ABSTRACT

In the past few years, the use of smartphone has been increased incredibly. The smartphones are used in our day to day activities. In this research we have tried to use smartphone for recording a human's day to day activities. The research focuses on collecting everyday data of a person using a triaxial accelerometer and derive results which determine the various activities performed by the person. The research has put to use Machine learning algorithms to derive results. The research compares the functionality of various machine learning algorithms and their efficiency to determine the activities performed by an individual.

The research also outputs a particular activity performed by an individual for a given time frame data. The research also takes into its ambit various problems related to machine learning and data science such as overfitting.

Table of Contents

Certificate ii

Acknowledgement iii

Abstract iv

Table of Contents v

List of Figures vi

List of Tables viii

CHAPTER 1: INTRODUCTION 1

1.1 Objective 2

1.2 Scope 3

1.3 Organisation 3

CHAPTER 2: BACKGROUND 4

2.1 Support Vector Machines 4

2.1.1 Working of SVM 5

2.1.2 Kernel Method 7

2.2 Convolutional Neural Networks 8

2.2.1 Local Connectivity 9

2.2.2 Convolutional Layer 9

2.2.3 Pooling 10

2.3 The problem of Overfitting 10

2.3.1 Regularization 11

2.3.2 Artificially Expanding the training Data 11

CHAPTER 3: PROPOSED FRAMEWORK 13

3.1 Working of Accelerometer 14

3.2 Framework 15

3.2.1 Data Gathering 15

3.2.2 Feature Extraction 17

3.2.3 Feature Engineering 18
3.2.4 Training and Testing 21

CHAPTER 4: IMPLEMENTATION 24
4.1 Dataset Used 24
4.2 Platform Used 25

CHAPTER 5: CONCLUSION 29
5.1 Results 29
5.2 Analysis 32
5.3 Future Work 33

REFERENCES 34

Appendix 36

List of Figures

Figure 2.1 Support Vector Machines 4
Figure 2.2 Working of SVM - I 5
Figure 2.3 Working of SVM - II 6
Figure 2.4 Working SVM - III 7
Figure 2.5 Representation of Kernel Method 8
Figure 2.6 Architecture of CNN 9
Figure 2.7 Representation of Convolutional layer 10
Figure 3.1 Accelerometer recording data 14
Figure 3.2 Framework representation 15
Figure 3.3 Holding position of Smartphone 16
Figure 3.4 Smartphone in hand vs smartphone in pocket 17
Figure 3.5 Graphical representation of Activities 18
Figure 3.6 Dimensionality reduction I 21
Figure 3.7 Dimensionality reduction II 22
Figure 4.1 Representation of readings recorded by smartphone 25
Figure 4.2 Implementation of SVM 28
Figure 4.3 Training SVM 29
Figure 4.4 Linear Kernel 29

List of Tables

Table 5.1 Performance of Algorithms 31
Table 5.2 Output of various Activities 32
Table 5.3 Classification of Activities 32

Chapter 1

INTRODUCTION

Most smartphones today come in fitted with vivid sensors. A normal smartphone contains different types of sensors such as gyroscope, accelerometer, barometer, light sensor, proximity sensor. These sensors generate data which can be analysed by the humans and can be used to describe human activities.

We in our work have used the data from these sensors to obtain results used to determine the activities conducted by humans. We have used accelerometer as a sensor to record human activities in given span of time and the variations of readings of accelerometer over these activities.

An accelerometer is basically a electromechanical device. It is used to measure acceleration forces. These forces may be static or dynamic. A static force can be a continous gravitational force acting upon a body whereas a dynamic force can be used to sense vibrations and motions. Accelerometers are used to describe the surroundings of an item. With this device we can determine is a body is moving or is stationary and even complicated movements if the body is moving uphill or downhill. We in our work have used these interesting properties of accelerometer to determine the motion of a human being.

Accelerometer as a device:

Accelerometer consist of microstructures and is capable of sensing capacitive changes in between these structures due to an external force. If any accelerative force comes into existence and disturbs one of these structures, the capacitance will notice a considerable change and

thereby the accelerometer will translate that capacitance to voltage for reading and human interpretation.

Typically accelerometers are made up of multiple axes. A two dimensional accelerometer can be used to determine the movement on two vector dimensions whereas a three dimensional accelerometer can be takes the height factor into consideration as well. Most smartphones generally come in fitted with three-axis models. We in our experiments have used a triaxial accelerometer.

An accelerometer can be used to record activities such as:

1. Sitting: Accelerometer data can be used to determine if a person is at rest by analysing the patterns of change in its readings.
2. Lying: It can determine is the phone is at rest which would state that the phone is lying and the user is at rest.
3. Standing: An accelerometer will determine is the person is standing, it will use static analysis to analyse gravitation force and differentiate sitting from standing.
4. Running : Accelerometer will detect sudden changes to determine is the object is in a rapid motion state.

1.1 Objective

The master thesis aims at recording accelerometer data of various activities performed by a human and describe the nature of activity. We use standard machine learning algorithm to not only categorize the data into various activities but also output the activity performed for a particular time frame. The algorithm uses standard machine learning algorithm to analyse various features from the data and output the activity performed in a particular time snap.

The underlying project uses Support Vector Machines with a linear kernel and Convolutional Neural Networks to classify the given activity.

1.2 Scope

The project can be used to encompass various activities performed by user throughout the day. It can help a user keep a track of his activities and manage them accordingly. It also has its uses in detecting the physical activity of a user and can also be combined with various health based processes to determine the calories burnt by a user. Further research on the same can be used to determine the location of a user in a particular area provided we are aware of the area of the location without gps.

The project can be used to determine the movements of any third party one would like to keep an eye upon. It can easily detect and provide the classification of various activities performed by the user throughout the day. Once trained on a standard dataset it can be deployed anywhere for a range of users.

1.3 Organisation

The master thesis is organised in the following chapters. The first and the current chapter introduces the report and also briefs the project along with the organisation of thesis into various modules. The second chapter contains the background of the project which explains in depth working of Support Vector Machines and Convolutional Neural Networks. The third chapter layouts the the experimental set up which includes the hardware as well as software set up of devices. The fourth chapter deals with the software based technical details and entails how the experiment has been implemented. The last but not the least, fifth chapter derives the result and presents them in understandable form.

Chapter 2

Background

2.1 Support Vector Machines

Support Vector Machines is yet another supervised machine learning algorithm technique which is used for classification as well as regression. In most cases it is used in classification. Each data item is treated as a point in n dimensional space (Where n represents the number of features). The value of each feature is the value of a particular coordinate in this n dimensional plane. Svm in simple words plots each point in the n dimensional plane and then finds a hyper plane that differentiates the classes very well.

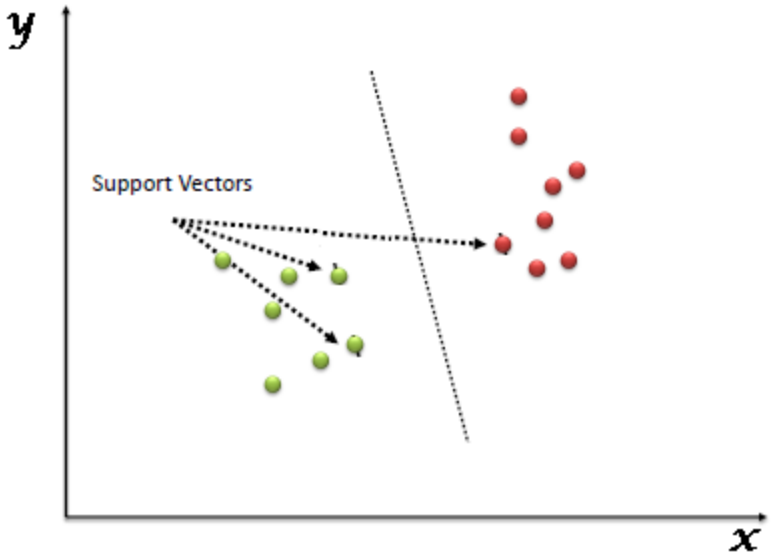

Figure 2.1

Support Vector Machines can be called as a frontier which segregates the classes. Support Vector Machines are very powerful classification algorithms. They are used in cases where high predictive power is required.

2.1.1 Working of SVM

As we have seen how SVM can be put to use to segregate the classes with a hyper plane. Now the question arises "How to identify the right hyper plane?"

- **Scenario-1:**

  It is important to remember the thumb rule. In the figure 2.2 it can be seen how different planes can be used to partition the given classes. But the best hyper plane is B.

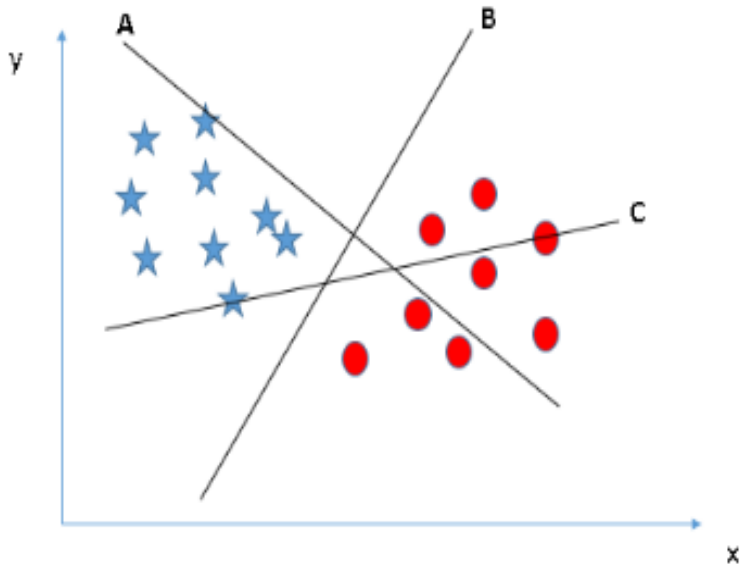

Figure 2.2

- **Scenario-2:**

When the hyper planes separate out the classes well. Here we need to decide which hyper plane to choose from all the hyper planes available. Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin.

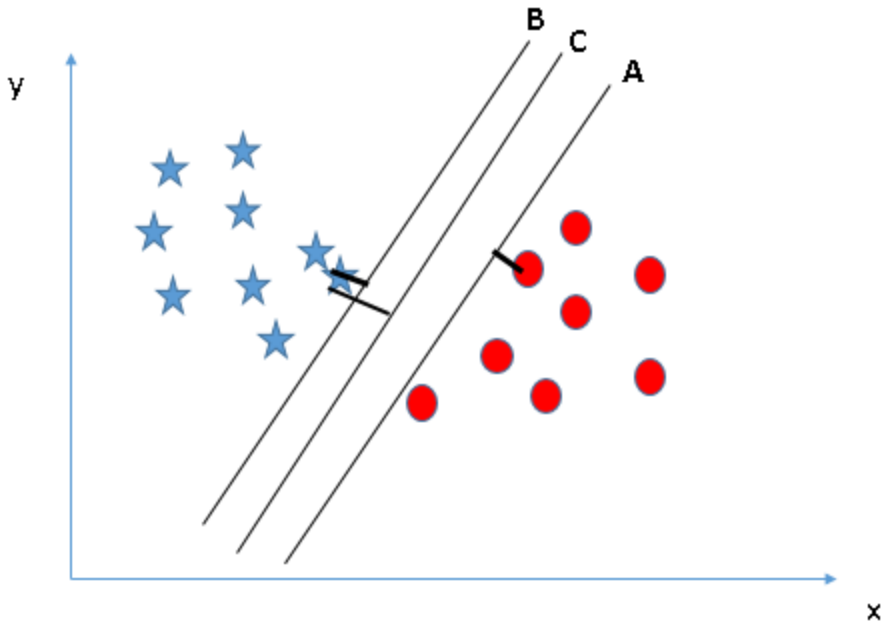

Figure 2.3

It is clear from figure 2.3 that hyper plane C has higher margin than hyper plane A and B. It is robust to select the hyper plane with a higher margin. A hyper plane with lower margin may lead to mis-classification.

- **Scenario-3:**

SVM selects the hyper plane which classifies accurately as compared to hyper plane with a higher margin. So in the race of a hyper plane with a higher margin and a hyper plane with accuracy, the hyper plane with a higher accuracy of classification wins the race. As

is clear from figure 2.4, the hyper plane the hyper plane B has a higher margin but hyper plane A classifies the classes more accurately thus hyper plane A is chosen.

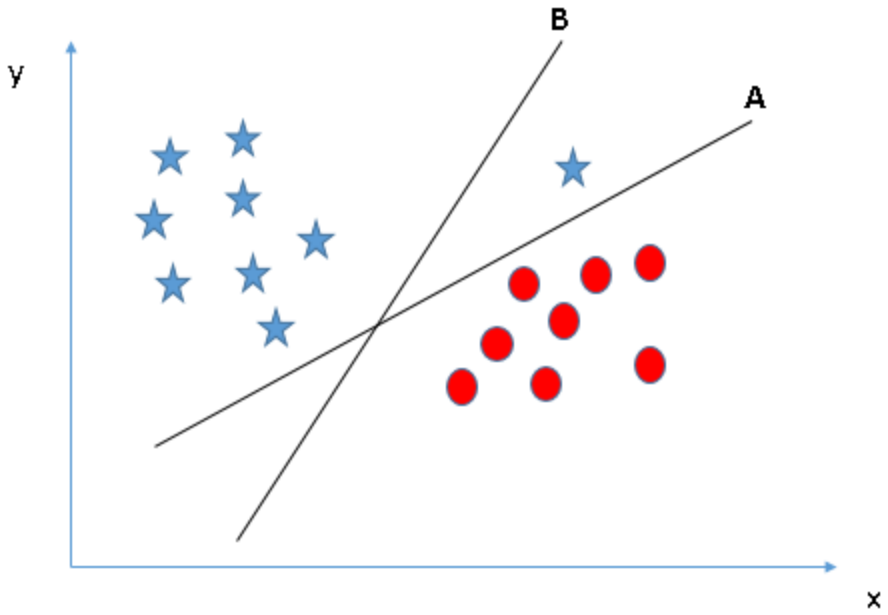

Figure 2.4

2.1.2 Kernel Method

In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the kernel trick. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs we've defined.

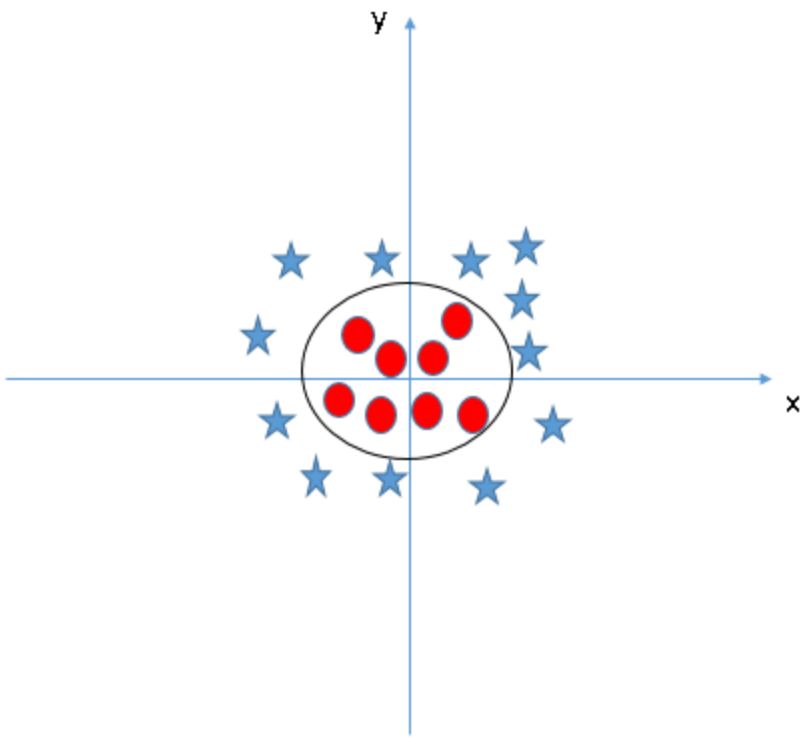

Figure 2.5: Representation of Kernel Method

When we look at the hyper plane it can be easily seen that it looks like a circle in the case above.

2.2 Convolutional Neural Networks

It is a simple type of feed forward Neural Network which uses the design and connectivity pattern of neurons of the animal visual cortex. In an animal cortex each neuron is arranged in such a way that each responds to the overlapping regions tilling the visual field.

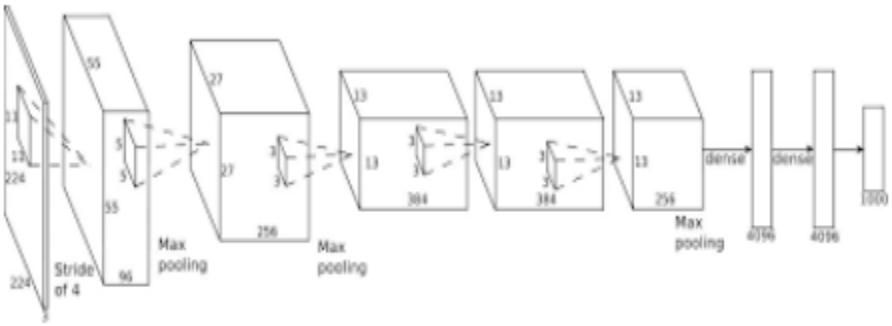

Figure 2.6: Architecture of CNN

CNNs consist of three layers :

1. Fully Connected
2. Convolutional
3. Pooling.

2.2.1 Local Connectivity

CNNs use the idea that all the neurons from the previous layer should not be connected to the current layer when dealing with high dimensional images such as such as images and audios. In CNNs each neuron is connected to a small region of input neurons. Thus Convolutional Neural Networks are able to use the functionality of one specific region rather than focussing on tempering the weights of all the neurons combined.

2.2.2 Convolutional Layer

Convolutional Layers are the major part of Convolutional Neural Networks. A convolutional layer consists of a series of kernels which convolve during the forward pass. The output of a convolutional layer is a 2 dimensional activation map of the kernel.

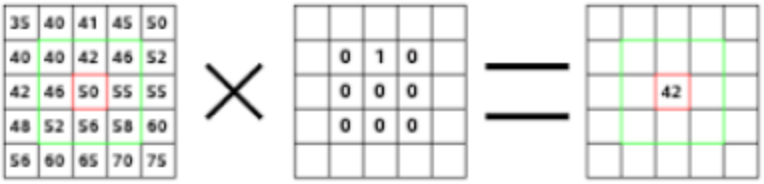

Figure 2.7: Representation of Convolutional Layer

Figure 2.3 represents how the process of convolution is conducted. As an example we have taken an image of 5X5 pixels in the figure. In the given image 0 represents the pixel to be black whereas 255 represents the pixel to be completely white. In the central block it has been defined a kernel of 3x3 pixels with all eight 0 except one white set at 1. The output shows the result of the computing kernel.

Irrespective of the fact that kernel is convolved, the position is determined by the stride. Considering stride 1 represents the typical convolutional whereas stride 2 determines how half of the convolutions are avoided and that there should be a distance to 2 pixels from the center.

2.2.3 Pooling

Pooling can be considered to non linear down sampling. A few common functions used to implement pooling are minimum, maximum, mean, median, percentile. But the most commonly used function is maximum. Max pooling basically partitions the input into several parts such that each part is a set of non overlapping rectangles. The function then gives the maximum value for each subregion entailed.

2.3 The problem of Overfitting

Overfitting occurs when the refers to a situation where the model describes the activity rather learning it. The separation of classification curve becomes so accurate that it goes on refining

each feature. This problem leads to the inability of the algorithm to determine the future sample characteristics.

We in this section will take into consideration various techniques to overcome the problem of overfitting:

2.3.1 Regularization

Regularization adds an extra term which in turn helps learn smaller weights and penalize the larger weights. Thus this term can be thought of bringing down the effect of a higher order polynomial thus making the curve less complicated, considering a 2 dimensional setup of the cost function curve. Moreover, having large biases doesn't make a neuron sensitive to its inputs in the same way as having large weights.

- L1 regularization

$$L(W,b) = L(W,b)_0 + \frac{\lambda}{2n}\sum_w w^2$$

- L2 regularization

$$L(W,b) = L(W,b)_0 + \frac{\lambda}{n}\sum_w |w|$$

where L(W, b)0 is the original unregularized loss function.

2.3.2 Artificially expanding the training data

One of the best ways to tackle the impact of overfitting problem the best way is to expand the training data. It may be difficult to acquire the new training data. In recent times some feature

based algorithms have been used to obtain a new training data from the available. Sometimes it may also be implemented by using extra features from the same data.

Chapter 3

Proposed Framework

Mobile phones today come in fitted with sensors such as accelerometer, gyroscope, etc. We in our experiments have used the data from a real smartphone. The data has been obtained realistically from a high end smartphone which had a triaxial accelerometer fitted. The accelerometer readings were then collected and recorded over a fixed span of time for various activities such as standing, walking, sitting, lying, etc.These activities were further subjected to a series of tests where the data was cleaned, normalized and filtered. Furthermore the data had to be engineered to be made suitable to be fed to a machine learning algorithm.
The activities were supposedly recorded by a single personnel over a fixed span of time. One activity was considered at one time. For example if a person is running only running data is recorded. The data post that activity is omitted.

The experiment is conducted over a span of 15 days and around 50,000 readings have been noted and put to the procedure of feature extraction. After the features were successfully extracted they were engineered and normalized to make them fit for use. The system then accepts the features as values to the machine. The machine is then trained over a set of values and then tested against the test values.

3.1 Working of accelerometer

An accelerometer is basically a electromechanical device. It is used to measure acceleration forces. These forces may be static or dynamic. A static force can be a continous gravitational force acting upon a body whereas a dynamic force can be used to sense vibrations and motions. Accelerometers are used to describe the surroundings of an item. With this device we can

determine is a body is moving or is stationary and even complicated movements if the body is moving uphill or downhill. We in our work have used these interesting properties of accelerometer to determine the motion of a human being.

Accelerometer consists of microstructures and is capable of sensing capacitive changes in between these structures due to an external force. If any accelerative force comes into existence and disturbs one of these structures, the capacitance will notice a considerable change and thereby the accelerometer will translate that capacitance to voltage for reading and human interpretation.

Typically accelerometers are made up of multiple axes. A two dimensional accelerometer can be used to determine the movement on two vector dimensions whereas a three dimensional accelerometer can be takes the height factor into consideration as well. Most smartphones generally come in fitted with three-axis models. We in our experiments have used a triaxial accelerometer.

Figure 3.1 : Accelerometer recording data

3.2 Framework

The main stages of our project consist of an ordered framework which collects data and the then analyses it based on the required. The main stages include:

1. Data Gathering
2. Feature Extraction
3. Feature Engineering
4. Training
5. Testing

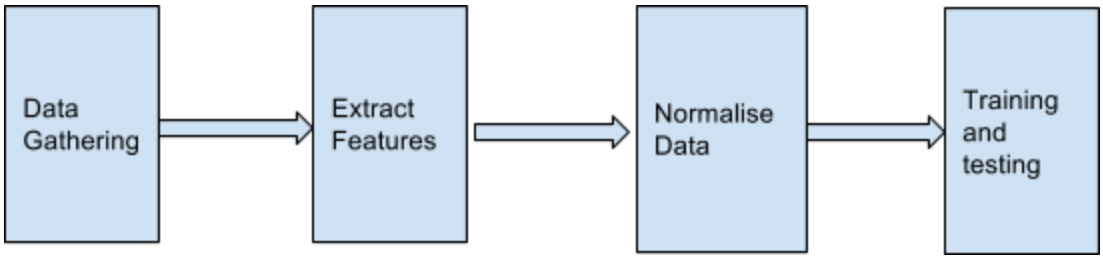

Figure 3.2 Framework Representation

3.2.1 Data Gathering

Data has been collected using a high end smartphone consisting of a triaxial accelerometer. The accelerometer readings were recorded at various instances and were saved on a local storage from time to time. The experiment has been done by a single individual over a time duration of 15 days for several hours everyday. The activities have not been overlapped which means only one activity was recorded at one time. The accelerometer readings are influenced by gravity value of 9.8 which has been omitted by our app used to record the readings. The app has been downloaded from play store and is licensed to record the correct readings of movements. The smartphone is kept in hand in a horizontal position across its length during the recording of the

experiment. The activities are recorded in such a way that no two activities overlap. The data is gathered and the is labeled with the activity performed against them. The raw data is then stored on the local machine for preprocessing.

As we record the data we can make out some very easy analysis. As we run the z axis and x axis readings can be seen to be deviating very frequently whereas when the phone is at rest the readings change very momentarily. These observations lay the foundations for extracting the features. The features have to be extracted in such a way that the features together give us sufficient information about the data present at our end. The features can be used to find out the patterns in the data and thus we can classify the data using our standard supervised machine learning algorithms.

Figure 3.3: Holding position of smartphone

Figure 3.3 shows Android smartphone with a built-in accelerometer and its axis directions

Figure 3.4: Smartphone in hand vs Smatphone in pocket while walking

Figure Sample acceleration signals for fast walking (performed by an identical subject) with the smartphone in two different positions

A simple observation can help us know how standard deviation can be used to determine the movement data. Such a feature can be helpful in organising the data as that of climbing upstairs.

3.2.2 Feature Extraction

The premature step before we go on to generalise the nature of data and find out all the relevant features is the evaluate the features which is to understand the type of features required to analyse the behaviour of various activities.In our work, the accelerometer generates three time series along x-axis, y-axis and z-axis. Each time series combines the linear acceleration due to body motion and due to gravity.

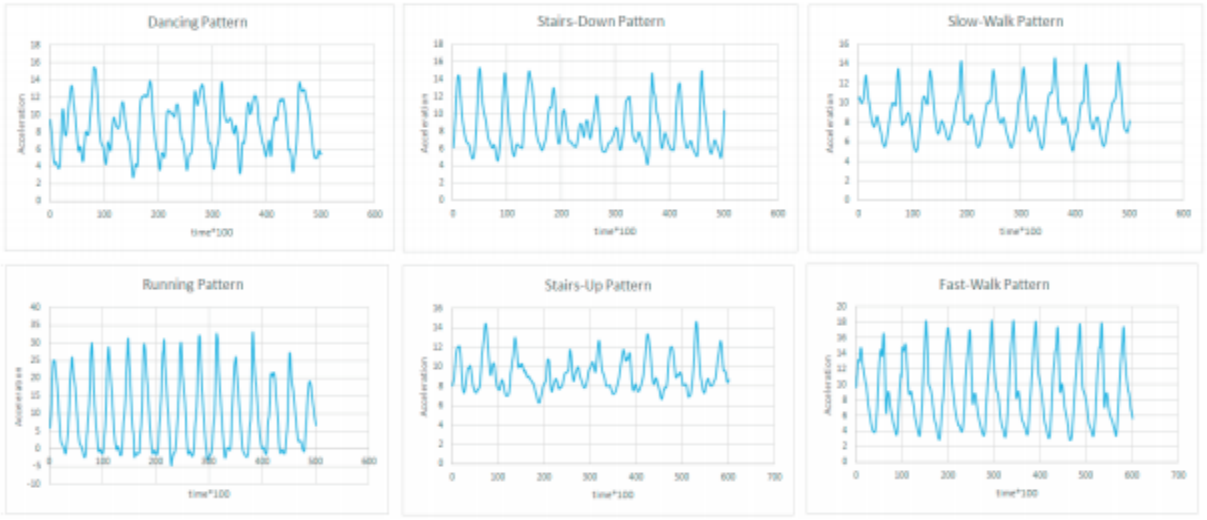

Figure 3.5 Graphical representation of activities

Different activities exhibit different types of features. The above is a sample data which shows the variation of the vector component along the time series. As is clear there are frequency peaks in the running data whereas there are steep rises in the climbing data. Thus we need to model this data and choose the features to feed to our machine learning algorithm. Various types of data may contain hidden information about various types of activities. For example as per the patterns it may be relevant to say that percentile can be a key factor in determining the running activity of a person. In the similar fashion various set of features can be used to distinguish our activities from one another. When fed to a machine these features can play a key role in distinguishing the planes and help us realise the classification in an efficient manner. For this to happen it is very important to model our data on the basis of key features which play a very significant role.

As we have seen all activities exhibit periodic behaviour but with distinctive patterns. We apply a technique called window overlapping. We will use this periodicity to calculate features from

consecutive reading using windowing. We choose a window of 128 samples which represents a time window of 1.28 seconds. These windows are selected one after the other and various features are extracted using their standard procedures.

Various features being used are:

1. Mean
2. Median
3. Percentile
4. Min
5. Max
6. Quartiles
7. Standard Deviation
8. Minimum and Maximum Peaks Frequency
9. Zero Crossings
10. Co-relations between axes

3.2.3 Feature Engineering

The data so obtained may not lie in the same range as we would expect it to be to feed it to the underlying machine learning algorithm. In order to fit the data to the machine and obtain relevant results it is very important to choose the data which fits in our range. The unwanted and irrelevant data should be discarded. This method of feature engineering is required to ensure that algorithm works properly and not waste data is fed to the machine. The unordered data may tend to improper training of the machine and thus improper classification.

The two techniques used in this process are:

1. Normalization
2. Dimensionality Reduction.

Normalization

Data is normalized to ensure that it is in the standard form and does not exceed the threshold value to give improper peaks. The unnormalized data can be lead to improper conjecture of machine learning algorithm. Data needs to be normalized to ensure all the data is in a proper range and does not acquire very high or very low values which may become a problem in the training of the machine.

A feature ***x*** may be normalized using the standard statistics formula:

$$x = \frac{x - \mu}{\sigma}$$

Where ***x*** is the feature to be normalised, ***μ*** is the mean of the feature data over the sample window and ***σ*** is the standard deviation of the feature data over the window.

Normalising in simple terms refers to transforming so as to transforming normal data. When the data is seen as vectors, normalising means transforming the data so that it acquires unit form. If data is seen in the form of variables normalising the data refers to transforming to normal distribution. When the data are hypothesized to be normal, normalizing means transforming to unit variance.

Dimensionality Reduction

When the data contains a lot of features it is feasible to reduce the number of features which convey the same information concisely. A simple example can be thought of a set of features of property attributes where length and breadth can be reduced to a new feature called area. Dimensionality reduction is essential before training the data as it is irrelevant to have useless features.These techniques are typically used while solving machine learning problems to obtain better features for a classification or regression task. It is thus a process of converting a set of

data having vast dimensions to a data set having lesser dimensions which conceal the same information as that of the original dataset.

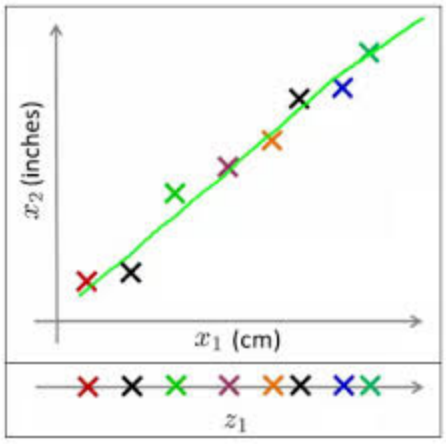

Figure 3.6 : Dimensionality reduction I

In Figure it shows 2 dimensions x1 and x2, which are let us say measurements of several object in cm (x1) and inches (x2). Now, if you were to use both these dimensions in machine learning, they will convey similar information and introduce a lot of noise in system, so you are better of just using one dimension. Here we have converted the dimension of data from 2D (from x1 and x2) to 1D (z1), which has made the data relatively easier to explain. In the similar n dimensions of data can be reduced to k dimensions. These k dimensions have to be chosen in such a way that these k dimensions convey the same information as that of the n dimensions. . These k dimensions can be directly identified (filtered) or can be a combination of dimensions (weighted averages of dimensions) or new dimension(s) that represent existing multiple dimensions well.

One of the straightforward applications of dimensionality reduction can be image processing.

Benefits of dimensionality reduction :

- It helps in compressing data and reducing the storage space required.
- It improves time required for performing the same computations. It helps algorithms act faster.

- Reducing the dimension of the data may help us to visualize and plot and understand the data and analyze it easily.

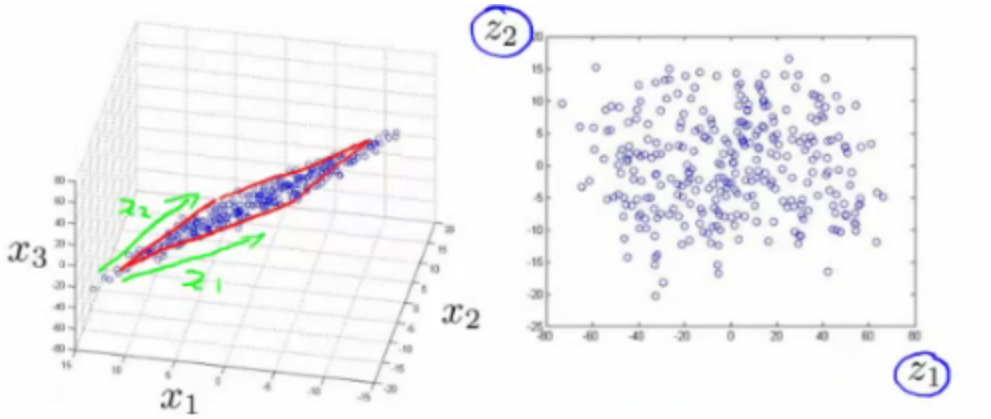

Figure 3.7: Dimensionality Reduction II

3.2.4 Training and Testing

We split our dataset into training set and testing set. In supervised machine learning practice we have the correct answer tags labelled to our data. Once the features have been obtained, it is important to use those features to distinguish the data. For instance in our example we have used Support vector Machine and Convolutional Neural Networks to train our data and instantiate it. On the basis of features and their properties, the training algorithm finds the most suitable plane which splits the data into various classes. Since we are dealing with a classification problem, a large amount of training data is partitioned out to be used for training the machine over a number of examples. It is important to note here that initial accelerometer readings are not used as the training set but the features collected as a whole are used for training the machine.

The rest of the readings are used as test sets. Test cases validate our data over its performance. It is not ideal that a machine may always give correct answer when made to perform against the test data. We are more interested in finding out the accuracy of the machine against the test data.

Testing is an important part in understanding the efficacy of the machine over a given set of data. The next step is that of validation which can be performed on an entirely new set of data obtained from any accelerometer exhibiting the same properties. We in our experiments have performed validation over a uniform data whose features have been obtained. This step helps in estimating the response of the machine over new data and also help in understanding the performance of the machine in response to new data. It may help us resolve the problem of over fitting or underfitting as we want our data to be classified in such a way that it is responsive to new incoming data sets from any accelerometer exhibiting the same properties.

Chapter 4

Implementation

The following section consists of various implementation to bring about our thoeretical and mathematical discussions a success. It explains in detail the data sets and platforms used. Not only it provides a deeper insight into how data was extracted and engineered but also explains how various machine learning algorithms were implemented using our platform.

4.1 Dataset Used

As it has been mentioned the dataset has been obtained from accelerometer readings with experiments over a range of time. The dataset has been partitioned into the training dataset and test dataset. Both datasets are stored in .csv format. Training set is used to train the data and classify into various classes whereas test data is used to validate the efficiency of the trained machine. Figure 4.1 shows a snapshot of the raw data obtained from the mobile phone sensor.

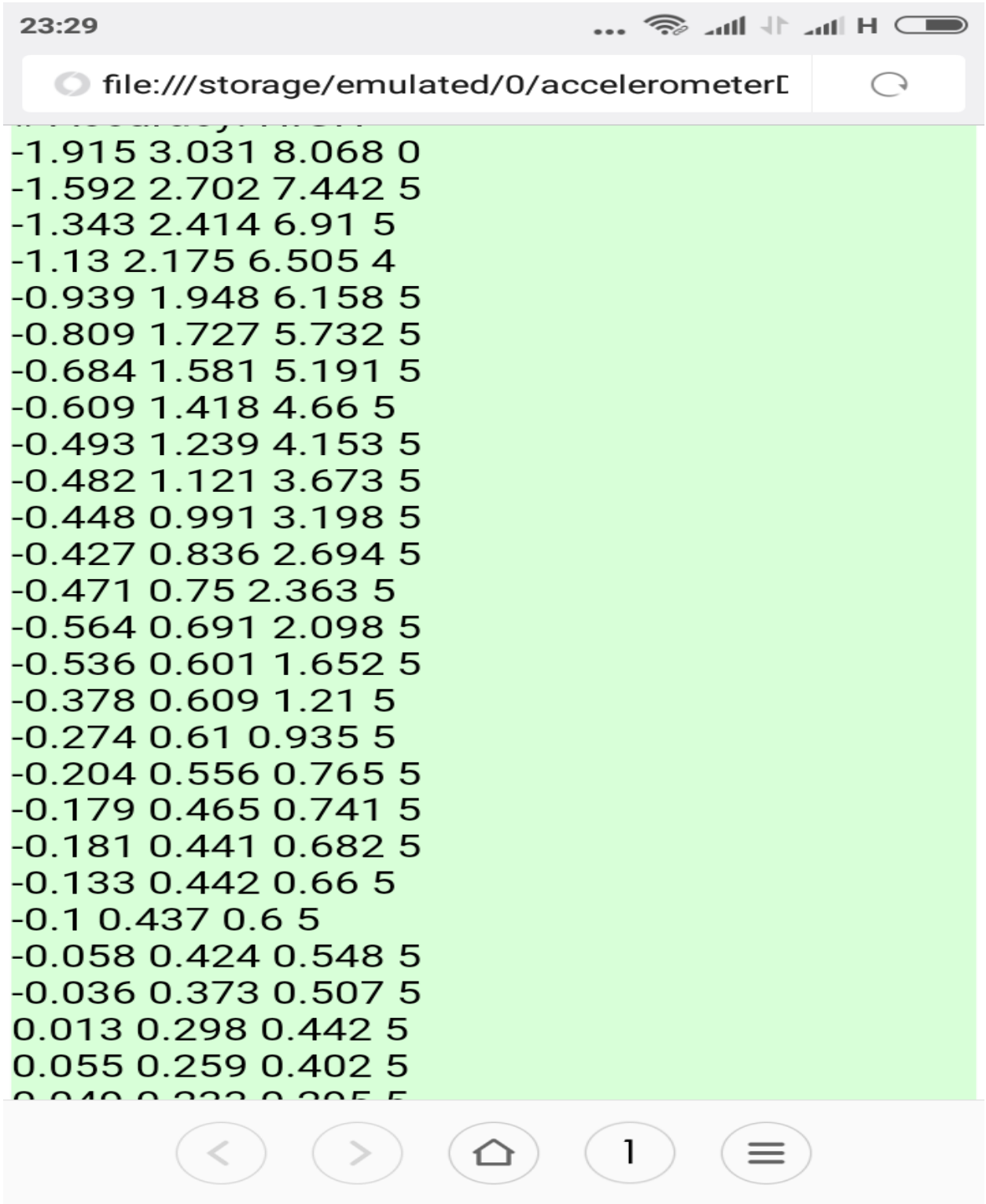

Figure 4.1 Representation of readings as recorded by smartphone

In our experiment we have used pandas to import the data. Pandas is a standard python library to import the data sets. Pandas use data frame to hold the for operations. A data frame can be thought of a multidimensional array with standard functions defined within python library to perform standard operations on the data set. ***readcsv()*** function is used to read any standard format file from the local hard disk. Pandas then import this file to data frame which stores the data and makes it ready for various operations to be performed upon it. For example ***iterrows()*** function helps iterate over each record of the file. Fig shows a snapshot of the data used in our examples.

Following are the advantages of using pandas over our dataset:

1. Having an R-style dataframe (with column names) can help a lot in keeping track of your data.

2. A numpy array requires homogeneous data. With a pandas dataframe, you can have different data types (float, int, string, datetime, etc) all in one place

3. Pandas has built in functionality for a lot of common data-processing applications: for example, easy group by syntax, easy joins (which are also really efficient in pandas), rolling windows

4. Good IO capabilities; I frequently pull data from a MySQL database directly into a data frame

4.2 Platform used

To conduct experiments and perform various functions such as gathering, processing, engineering data, We have used python as our programming language. To gather all the requirements in one place we have used Canopy as our framework. As discussed in the previous section, various python modules have been used to gather and process data. Along with Pandas, modules such as numpy have been used to modify and change data.

In order to implement various machine learning algorithm, Scikit learn module is used. Scikit learn is a machine learning library in python which consists of simple and efficient tools for data analysis and data mining. It contains most machine learning algorithms implemented in its libraries. It is easier to use scikit learn and import standard machine learning algorithms rather than writing your own algorithms. Scikit learn machine learning algorithms are fast and efficient and great for dealing vast majority of data.

Actually, it's been the introduction to information Science. This science has become quite common recently. Competitions in machine learning square measure progressively control (for example, Kaggle, TudedIT), and their budget is commonly quite goodish.

The most common tools for an information somebody nowadays square measure R and Python. every tool has its professionals and cons, however Python wins recently altogether respects . This happened once there had appeared a really well documented Scikit-Learn library that contains an excellent variety of machine learning algorithms.

Please note that we'll concentrate on Machine Learning algorithms within the article. it's typically higher to perform the first information analysis by suggests that of the Pandas package that's quite easy to subsume on your own. So, let’s concentrate on implementation. For predictability, we have a tendency to assume that there's a feature-object matrix at the input, and it should on in an exceedingly *.csv file.

The scikit learn module helps in the following:

1. Data Loading
2. Feature Extraction using Numpy
3. Feature selection
4. Algorithm Design

Data Loading is the basic and the first part of the experiment where we need to load the data collected through a series of real time experiments. Although this work is now done with pandas

and as a matter of fact we have also used pandas as the basic tool to extract data but scikit learn along with numpy can also be used to load data into array and work effectively on them.

Feature extraction is the process of analysing the data and its properties and to choose the features wisely which best determine the nature of our data. Scikit learn module can be used with numpy arrays to inherit data and deal with the most widely used features of the data in the present dataset.

Feature selection directly refers to reducing the features and their dimensionality. It is important to choose only relevant features to make our algorithms work efficiently. Scikit learn can be used to choose those features using its inbuilt support. Figure 4.2 shows how scikit learn can be used to train and model data using SVM.

Scikit learn consists of modules which contain codes of algorithms implemented in them. Scikit learn can import these modules and import algorithms without us to have write codes in full. Also they can be used to design the algorithm. For instance to choose what kind of kernel to use in SVM.

```
#Import Library
from sklearn import svm
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_d
ataset
# Create SVM classification object
model = svm.svc(kernel='linear', c=1, gamma=1)
# there is various option associated with it, like changing kernel, gamma and C value. Will discuss m
ore # about it in next section.Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

Figure 4.2 Implementation of SVM

```
sklearn.svm.SVC(C=1.0, kernel='rbf', degree=3, gamma=0.0, coef0=0.0, shrinking=True, probability=Fals
e,tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, random_state=None)
```

Figure 4.3: Training SVM

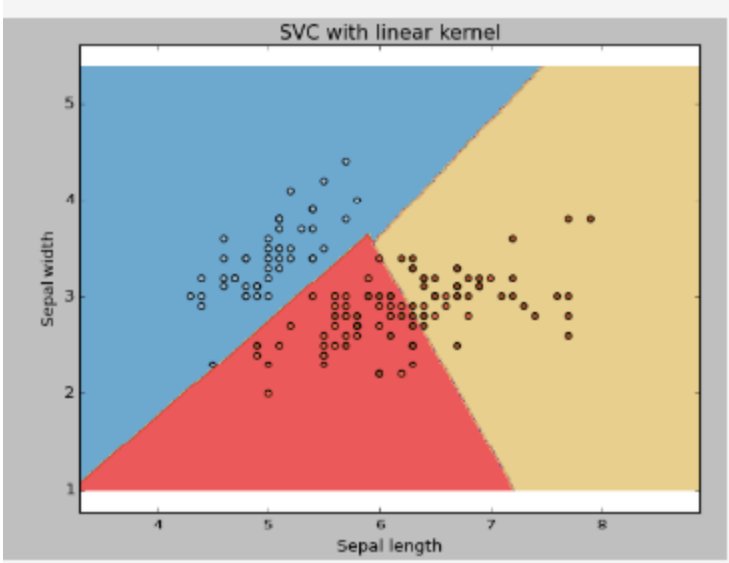

Figure 4.4: Linear Kernel

Chapter 5

Conclusion

The following section consist of various results that we have obtained from our experiments. The section not only compares the two techniques of Machine learning but also provides a specific output for a given sample of data. It specifies us the efficiency in calculating the results using the two machine learning techniques we have used in our experiment and also performs calculations on the given dataset and finds out the specific activity of the person.

We in this section will also analyse how the two machine learning algorithms have been used and what role have various features played in the calculations. We will also put to test the efficiency of the two machine learning and explore how the machine learning algorithms work against a new dataset. We will also take into account the problem of overfitting and how it can effect the two machine learning algorithms.

The last part of this section talks about how our experiments can be used to perform various experiments in the future. We shall be discussing the future scope of this project and how it can be used with various other sensors to perform and analyse different human activities.

5.1 Results

The section explains how Machine learning algorithms have worked on the underlying dataset to obtain the results and calculate the efficiency of the machine learning algorithms. The following table shows accuracy with each machine learning algorithm. It was found that second order kernel in svm overfitted the data whereas the accuracy of the convolutional neural network was lower than that of a first order SVM kernel.

Algorithm	Accuracy
SVM with linear Kernel	88.04%
SVM with 2nd order Kernel	83.67%
CNN	85.67%

Table 5.1 Performance of Algorithms

It could be shown that a second order SVM kernel overfitted the data and did not behave very well against the new dataset obtained. It would have given a great accuracy against the existing training data but it did not respond to the new data set introduced.

We later partitioned a new test set into small parts of several activities. The following table shows how it responded to a particular activity data.

Dataset	Actual Activity	Activity Recognized by SVM
Dataset 01	Standing	Standing
Dataset 02	Sitting	Sitting
Dataset 03	Walking Downstairs	Walking Downstairs
Dataset 04	Walking Upstairs	Walking Upstairs
Dataset 05	Walking	Walking
Dataset 06	Standing	Standing
Dataset 07	Standing	Standing
Dataset 08	Lying	Lying
Dataset 09	Sitting	Sitting

Dataset 10	Walking Upstairs	Walking Upstairs
Dataset 11	Lying	Lying
Dataset 12	Sitting	Sitting
Dataset 13	Walking Downstairs	Walking Downstairs

Table 5.2 Output of various Activities

As could be seen our experiments give a 100% accuracy in recognising the activities separately. Each activity was recognized on the basis of majority. Say if the machine outputs that 85% of times the person was standing then we output the result as standing. Based on these calculations our outputs were computed over thirteen files which had been gathered by breaking down a sample test data set.

We have not used CNNs to compute activities separately. These activities have been computed using SVM with linear kernel. As we had stated it was found that the SVM with second order kernel resulted in overfitting the training and did not act well towards the new test data. So we have computed the actual activities using the SVM with second order kernel.

The program also finds out various activities performed by the individual over the total time.

Activity	**Percentage of time (approx)**
Standing	16%
Sitting	18%
Walking	20%
Walking Upstairs	13%
Walking Downstairs	15%
Lying	18%

Table 5.3

5.2 Analysis

For our experiments it was shown that the Support Vector Machines with a linear kernel gave a better performance over Convolutional Neural Networks. As we tried to perform the same activity using SVMs with second order kernel we encountered the problem of overfitting. Convolutional Neural Networks are so designed that they do not encounter the problem of overfitting. But the performance of SVMs were slightly better than those of Convolutional Neural Networks.

So we used SVM with linear Kernel to compute the output of various activity data when obtained separately.

The overfitting problem of SVM could have been resolved with the use of Radial Basis Function(RBF) Kernel. Unfortunately, the performance of the SVM can be quite sensitive to the selection of the regularisation and kernel parameters, and it is possible to get over-fitting in tuning these hyper-parameters via e.g. cross-validation. The theory underpinning SVMs does nothing to prevent this form of overfitting in model selection.

Now the verdict is clear, the linear model is not only subjectively better, but now also quantitatively performs better as well (measured using the squared error metric). But this leads to a very interesting point about training and evaluating machine learning models. By building a very complex model, it's quite easy to perfectly fit our dataset. But when we evaluate such a complex model on new data, it performs very poorly. In other words, the model does not *generalize* well.

This is a phenomenon called *overfitting*, and it is one of the biggest challenges that a machine learning engineer must combat. This becomes an even more significant issue in deep learning, where our neural networks have large numbers of layers containing many neurons. The number of connections in these models is astronomical, reaching the millions. As a result, overfitting is commonplace.

5.3 Future Work

The future work on the experiments conducted can be very wide and subjective. Our experiments have dealt with human day to day activities and have put a calculation on the percentage of time a particular person was doing a particular activity.

We can enhance this experiment where a person's phone usage time may also be calculated. These calculations can be used to calculate his power consumption. The generation today is using smartphone excessively. We can design apps on the similar layout to encourage humans to make more physical activity and engage less in smartphones for better health.

In the similar ways it can be extended to calculate the calories burnt by a user in a day. We can enhance the same model to find out if the human is performing a physical activity. It can be designed to find out an individual's performance in the gym or while running. We can combine our calculations with simple health formula to find out the calorie giveaway of a human being. Such apps can be of great use to keep a track of fitness activities by humans.

Sensors combined with other sensors such as Barometer can be used to give readings of an individual's pressure thus determining the height at which individual is located. These can be very handy in determining the location of an individual without an Global positioning system.

The GPS today determines the position of the individual irrespective of his height. Our experiments when used in unison with barometer and Gps can also be significant in determining the height at which a particular individual is located.

The same technology when used with other sensors can also determine the location of an individual at a given location provided we are aware of the structure of the particular location. Such apps can also come in handy in determining the direction of movement of an individual.The research uses the sensor data. This data when used to obtain the right parameters can be very useful in a number of activities.

References

[1] M. Makikawa and D. Murakami, “Development of an ambulatory physical activity and behavior map monitoring system,” in 18th Annual Conf. IEEE Engineering in Medicine Biology Soc. Amsterdam, Holland, 1996.

[2] M. J.Mathie, N. H. Lovell, A. C. F. Coster, and B. G. Celler, “Determining activity using a triaxial accelerometer,” in Proc. 2nd Joint EMBS-BMES Conf., Houston, TX, Oct. 2002.

[3] A. V. Ng and J. A. Kent-Braun, “Quantitation of lower physical activity in persons with multiple sclerosis,” Med. Sci. Sports Exerc., vol. 29, pp. 517– 523, 1997.

[4] K. Kiani, C. J. Snijders, and E. S. Gelsema, “Computerized analysis of daily lifemotor activity for ambulatory monitoring,” Technol.Health Care, vol. 5, pp. 307–318, 1997.

[5] J. Fahrenberg, F. Foerster, M. Smeja, andW. M¨uller, “Assessment of posture and motion by multichannel piezo resistive accelerometer recordings,” Psychophysiol., vol. 34, pp. 607–612, 1997.

[6] F. Foerster and J. Fahrenberg, “Motion pattern and posture: Correctly assessed by calibrated accelerometers,” Behav. Res.Meth. Instrum.Comput., vol. 32, pp. 450–457, 2000.

[7] P. H. Veltink, H. B. Bussmann, W. de Vries, W. L. Martens, and R. C. van Lummel, “Detection of static and dynamic activities using uniaxial accelerometers,” IEEE Trans. Rehabil. Eng., vol. 4, no. 4, pp. 375–385, Dec. 1996.

[8] Schlömer, T., Poppinga, B., Henze, N., Boll, S, “ Gesture Recognition with a Wii Controller, ” In: International Conference on Tangible and Embedded Interaction (TEI 2008), Bonn Germany, February 18-20, pp. 11–14 ,2008.

[9] Mäntyjärvi, J., Kela, J., Korpipää, P., Kallio, S,"Enabling fast and effortless customization in accelerometer based gesture interaction," In: Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia (MUM 2004), October 27-29, pp. 25–31. ACM Press, New York ,2004.

[10] Mäntylä, V.-M," Discrete Hidden Markov Models with Application to Isolated User-Dependent Hand Gesture Recognition," VTT publications (2001).

[11] Hofmann, F.G., Heyer, P., Hommel, G,"Velocity profile based recognition of dynamic gestures with discrete hidden markov models," In: Wachsmuth, I., Fröhlich, M. (eds.) GW 1997. LNCS, vol. 1371, pp. 81–95,1998.

[12] Cho, S.-J., Choi, E., Bang, W.-C., Yang, J., Sohn, J., Kim, D.Y., Lee, Y.-B., Kim, S,"Two stage Recognition of Raw Acceleration Signals for 3D-Gesture-Understanding Cell Phones," In: 10th International Workshop on Frontiers in Handwriting Recognition ,2006.

[13] Mirabella, Brischetto, Mastroeni," MEMS based gesture recognition, Human System Interactions (HSI)," 2010 3rd Conference on, 2010: 599-604.

[15] Nishkam,R., Nikhil,D., Preetham,M.,Michael,L.L.,"Activity Recognition from Accelerometer Data," Proceeding IAAI'05 Proceedings of the 17th conference on Innovative applications of artificial intelligence: AAAI Press,2005: 1541-1546.

[16] Bao,L., Stephen,S.I.," Activity recognition from user annotated acceleration data". In Proceedings of the 2nd International Conference on Pervasive Computing,2004: 1-17.

Appendix A

Activity_Recognition.py

```
#!/usr/bin/python

# -*- coding: utf-8 -*-

import numpy as np

import parametersConfig, dataInspector, featuresManager, dimensionalityManager, biometricsSupervisedLearning

from sklearn import preprocessing


def activityRecognition():


    #  1  ----------------------------  Pre-processing  /  Feature evaluation  ----------------------------

    print  "\n  1  ----------------------------  Pre-processing  /  Feature evaluation  ----------------------------"

    DI = dataInspector.dataAnalysis()

    # Loading data

    print "**** Loading data ****"
```

```
    data                                                              =
DI.dataLoader('data/HAR_UCI_RAW.txt',',',['userID','activity',
'experimentID','xAcceleration', \


'yAcceleration','zAcceleration','xAngVelocity','yAngVelocity',
'zAngVelocity'], \

          (int,int,int,float,float,float,float,float,float))

    # Selecting certain activities (walking, walking upstairs,
walking downstais, standing and laying)

    print "**** Choosing data linked walking, walking upstairs,
walking downstais, standing and laying activities **** "

    data = DI.activitiesSelector(data, [1,2,3,4,5,6])

    # Checkif dataset is balanced or imbalanced

    print " **** Activities distribution **** "

    DI.dataSummarizer(data,['Walking', 'Walking upstairs', 'Walking
downstairs', 'Sitting', 'Standing', 'Laying'])



    # 2 ----------------------------------- Feature computation
-----------------------------------

    print "\n 2 ----------------------------------- Feature
computation -----------------------------------"

    FM = featuresManager.featuresAgent()

    # New features computation

    print "**** New features computation ****"
```

```
	data = FM.featuresComputation(data, parametersConfig.windowSize)

	# Feautures headers

	headers = parametersConfig.headers

	# Features formats

	formats = ['int'] + (['float'] * (data.shape[1] - 1))

	# Setting headers and formats (for more useful data management)

	dt = {'names':headers, 'formats': formats}

	dataHeaders = np.zeros(data.shape[0], dtype=dt)

	for columnNumber in range(0,data.shape[1]):

		dataHeaders[headers[columnNumber]] = data[:,columnNumber]

	# Creating boxplots for feautures computed

	print "**** Creating boxplots for feautures computed ****"

	DI.boxplotPrinter(dataHeaders,headers[7:len(headers)-1])


	del dataHeaders # Saving memory


	# 3 ---------------------------------- Dimensionality reduction
----------------------------------

	print  "\n  3  ----------------------------------  Dimensionality
reduction ----------------------------------"

	MD = dimensionalityManager.manageDimensionality()
```

```
	# Feature selection (Removing all low-variance features)

	print "**** Features selection ****"

	featuresSelected = MD.featureSelection(data[:,1:data.shape[1]],headers[1:len(data)])

	# Feature extraction (PCA)

	print "**** Features extraction ****"

	featuresExtracted = MD.featureExtraction(featuresSelected)

	# Adding labels to transformed data

	labels = data[:,0].reshape(data.shape[0],1)

	data = np.column_stack((labels,featuresExtracted))


	del featuresSelected, featuresExtracted # Saving memory


	# 4 ---------------------------- Classification (Supervised Learning) ----------------------------

	print "\n 4 ---------------------------- Classification (Supervised Learning) ----------------------------"

	BSL = biometricsSupervisedLearning.supervisedLearning()

	# Splitting between training and test datasets

	# Stratified sampling (Training: 0.8; Testing: 0.2)

	print "**** Splitting between training and test datasets. Stratified sampling (Training: 0.8; Testing: 0.2) ****"
```

```
    trainSet = np.empty((0,data.shape[1]))

    testSet = np.empty((0,data.shape[1]))

    # For each activity, 80% are for training and 20% for testing

    for i in range(1,7):

        dataChoosed =
BSL.trainingAndTestDataChooser(BSL.dataClassSelector(data, i), 0.8)

        trainSet = np.append(trainSet, dataChoosed[0], axis=0)

        testSet = np.append(testSet, dataChoosed[1], axis= 0)

        # For SVM purposes (data needs to be scaled)

        scaledTrainSet = preprocessing.scale(trainSet)

        scaledTestSet = preprocessing.scale(testSet)



    # Train set

    trainSetFeatures = trainSet[:,1:trainSet.shape[1]] # Depedent
variables

    trainSetTarget = trainSet[:,0] # Independent variable: activity
(walking, sitting, laying, etc.)

    # For SVM purposes (data needs to be scaled)

    scaledTrainSetFeatures =
scaledTrainSet[:,1:scaledTrainSet.shape[1]]

    # Test set
```

```
    testSetFeatures = testSet[:,1:testSet.shape[1]] # Depedent
variables

    testSetTarget = testSet[:,0] # Independent variable: activity
(walking, sitting, laying, etc.)

    # For SVM purposes (data needs to be scaled)

    scaledTestSetFeatures =
scaledTestSet[:,1:scaledTestSet.shape[1]]



    # Using differents algorithms

    # Naive Bayes

    print "**** Supervised Learning Algorithms****"


BSL.gaussianNaiveBayes(trainSetFeatures,trainSetTarget,testSetFeature
s,testSetTarget)

    # SVC


BSL.supportVectorClassification(scaledTrainSetFeatures,trainSetTarget
,scaledTestSetFeatures,testSetTarget,parametersConfig.SVMLibrary)

    # Decision Trees


BSL.decisionTree(trainSetFeatures,trainSetTarget,testSetFeatures,test
SetTarget)


if __name__ == '__main__':
```

```
activityRecognition()
```

Human Activity Recognition using Smartphone Data

ORIGINALITY REPORT

12%	10%	4%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.analyticsvidhya.com Internet Source	5%
2	www.kdnuggets.com Internet Source	2%
3	www.cs.umb.edu Internet Source	1%
4	stats.stackexchange.com Internet Source	1%
5	www.efxkits.co.uk Internet Source	<1%
6	Ronao, Charissa Ann, and Sung-Bae Cho. "Human activity recognition using smartphone sensors with two-stage continuous hidden Markov models", 2014 10th International Conference on Natural Computation (ICNC), 2014. Publication	<1%
7	Lecture Notes in Computer Science, 2015. Publication	<1%

8	Martínez, Héctor P., and Georgios N. Yannakakis. "Deep Multimodal Fusion : Combining Discrete Events and Continuous Signals", Proceedings of the 16th International Conference on Multimodal Interaction - ICMI 14, 2014. Publication	<1%
9	neuralnetworksanddeeplearning.com Internet Source	<1%
10	Submitted to University of Wales Swansea Student Paper	<1%
11	Submitted to K. J. Somaiya College of Engineering Vidyavihar, Mumbai Student Paper	<1%
12	Submitted to UC, Boulder Student Paper	<1%
13	www.merak.com Internet Source	<1%
14	Submitted to Coventry University Student Paper	<1%
15	www.mtome.com Internet Source	<1%
16	www.bentley.edu Internet Source	<1%

"Applications of Evolutionary Computation",