

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

SNEHA J

REG. NO. 38119011



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119, TAMILNADU

MARCH 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI

CHENNAI– 600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Sneha J (Reg. No: 38119011)** who carried out the project entitled “**Human activity recognition with smartphones using machine learning process**” under my supervision from June 2021 to November 2021.

Internal Guide

Dr. R.M.Gomathi, M.Tech, Ph.D.,

Head of the Department

**Dr. L. LAKSHMANAN M.E., Ph.D.,
Dr.S.VIGNESHWARI, M.E., Ph.D.,**

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Sneha J** hereby declare that the Project Report entitled **Human activity recognition with smartphones Using Machine Learning Process** done by me under the guidance of **Dr. R.M.Gomathi, M.Tech., Ph.D.,** and **Dr.L.LAKSHMANAN M.E., Ph.D.,** and **Dr.S.VIGNESHWARI M.E.,Ph.D.,** at **Sathyabama Institute of Science and Technology** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE:

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean, School of Computing , and Dr.L.Lakshmanan M.E., Ph.D., Dr.S.Vigneshwari M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.R.M.Gomathi M.Tech., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

Human activity recognition with smartphones using machine learning process

TABLE OF CONTENT

SL.NO	TITLE	PAGE.NO
01	ABSTRACT	
02	EXISTING SYSTEM 2.1 DRAWBACKS	
03	INTRODUCTION 3.1 DATA SCIENCE 3.2 ARTIFICIAL INTELLIGENCE	
04	MACHINE LEARNING	
05	PREPARING DATASET	
06	PROPOSED SYSTEM 6.1 ADVANTAGES	
07	LITERATURE SURVEY	
08	SYSTEM STUDY 8.1 OBJECTIVES 8.2 PROJECT GOAL 8.3 SCOPE OF THE PROJECT	
09	FEASIBILITY STUDY	
10	LIST OF MODULES	
11	PROJECT REQUIREMENTS 11.1 FUNCTIONAL REQUIREMENTS 11.2 NON-FUNCTIONAL REQUIREMENTS	
12	ENVIRONMENT REQUIREMENT	
13	SYSTEM ARCHITECTURE	
14	WORKFLOW DIAGRAM	
15	MODULE DESCRIPTION 15.1 MODULE DIAGRAM 15.2 MODULE GIVEN INPUT EXPECTED OUTPUT	
16	DEPLOYMENT	
17	CODING	
18	CONCLUSION	
19	FUTURE WORK	

LIST OF FIGURES

SL.NO	TITLE	PAGE.NO
01	SYSTEM ARCHITECTURE	
02	WORKFLOW DIAGRAM	
03	MODULE DIAGRAM	

1. Abstract:

Human activity recognition requires to predict the action of a person based on sensor-generated data. It has attracted major interest in the past few years, thanks to the large number of applications enabled by modern ubiquitous computing devices. It classifies data into activities like Walking, walking upstairs, walking down stairs, sitting,

standing, laying are recognized. Sensor data generated using its accelerometer and gyroscope, the sensor signals (accelerometer and ore.

2.EXISTING SYSTEM

This paper proposes and develops a cascaded deep neural network (CDNN) to analyze data, collected using the sensors of smart-phones, to accurately localize an object in an indoor environment. There are many existing studies that have attempted to identify the location of an inhabitant in a room through the analysis of the radio signal strength (RSS), with varying success. places. The proposed CDNN suffers from space and computational complexities, especially for training each of the DNNs in the CDNN. We also plan to improve the CDNN structure such that the number of DNNs can be reduced without affecting the localization accuracy.

2.1 Drawbacks

- The CDNN achieves only 80.41% and 74.14% localization accuracies for the training and testing data.
- Evidently proves the difficulty of localizing the exact position of the object within a very short distance/radius (say a 1 to 1.5 m radius).

3.INTRODUCTION

Domain overview

3.1 Data Science

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data, and apply knowledge and actionable insights from data across a broad range of application domains.

The term "data science" has been traced back to 1974, when **Peter Naur** proposed it as an alternative name for computer science. In 1996, the International Federation of Classification Societies became the first conference to specifically feature data science as a topic. However, the definition was still in flux.

Required Skills for a Data Scientist:

- **Programming:** Python, SQL, Scala, Java, R, MATLAB.
- **Machine Learning:** Natural Language Processing, Classification, Clustering, ...
- **Data Visualization:** Tableau, SAS, D3.js, Python, Java, R libraries.
- **Big data platforms:** MongoDB, Oracle, Microsoft Azure, Cloudera.
-

3.2 ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. The term may also be applied to any machine that exhibits traits associated with a human mind such as learning and problem-solving.

4.MACHINE LEARNING

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python. Process of training and prediction involves use of specialized algorithms. It feed the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated in to three categories. There are supervised learning, unsupervised learning and reinforcement learning.

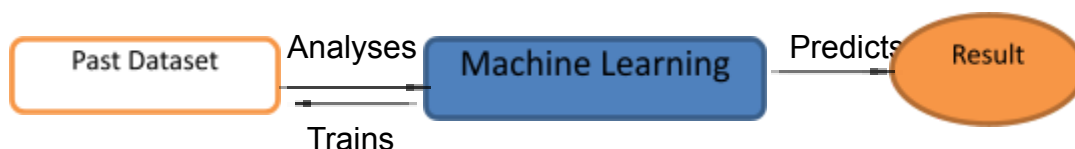


Fig: Process of Machine learning

recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development o

Human Activity Recognition or HAR for short, we can able to predict the human activity based on a trace of their movement using mobile sensors.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments

generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate

[15].However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15].However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15].However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human activities.

In recent years, Human Activity Recognition (HAR) through smart phones became a well known field of research. As we have entered the era of intelligent environment, the automated detection of Activity has become a point of high interest. Intelligent environments generally exploit information gathered from users and their

environments in order to produce an appropriate action [16]. In this regard, different studies have been conducted in this field. Based on these studies, we observed that basic locomotion activities like Walking, Running, Sitting, Lying on bed can be detected with good accuracy rate [15]. However, similar activities, such as Going upstairs or downstairs, Slow running (Jogging), Fast Walking can not be detected perfectly [15]. It is important to detect these activities for development of systems that promote the improvement of people's Quality of Life (QoL) through the recognition of human acti

5.Preparing the Dataset

The experiments have been carried out with a group of 30 volunteers within an age bracket of 19-48 years. Each person performed four activities (WALKING,SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, we captured 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz. The experiments have been video-recorded to label the data manually. The obtained dataset has been randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data.

Variable	Description
Smart Phone	Samsung (Galaxy S II)
Sensors	Accelerometer and Gyroscope
Axis	3-axis (x, y, z)
No. of volunteers	30
Volunteers Age	19-48
Features	19
Activates	WALKING, SITTING, STANDING, LAYING

6. Proposed System

The process of human activities recognition is very similar to a general-purpose pattern recognition system and corresponds to a set of steps ranging from data collection to activities classification. This process involves a set of transformations of the raw data extracted from sensors to generate efficient classification models of human activities. While applying photo **graphic-based** method is critical to evaluate parameters and it's taken data size is high

To overcome this method to implement machine learning approach by user interface of GUI application

Multiple datasets from different sources would be combined to form a generalized dataset, and then different machine learning algorithms would be applied to extract patterns and to obtain results with maximum accuracy.

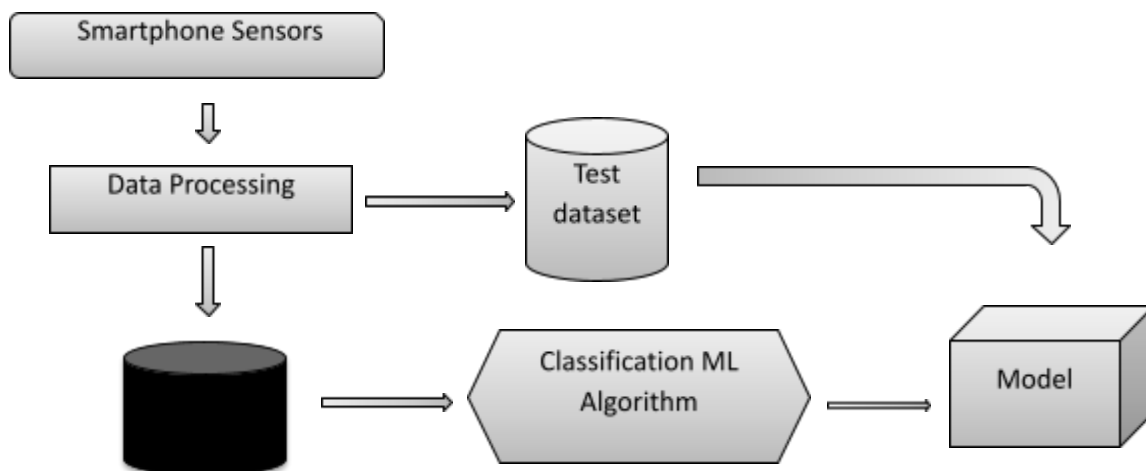


Fig: Architecture of Proposed model

6.1 Advantages:

These reports are to the investigation of applicability of machine learning techniques for air quality forecasting in operational conditions.

Finally, it highlights some observations on future research issues, challenges, and needs.

7. LITERATURE SURVEY

General

A literature review is a body of text that aims to review the critical points of current knowledge on and or methodological approaches to a particular topic. It is secondary sources and discuss published information in a particular subject area and sometimes information in a particular subject area within a certain time period. Its ultimate goal is to bring the reader up to date with current literature on a topic and forms the basis for another goal, such as future research that may be needed in the area and precedes a research proposal and may be just a simple summary of sources. Usually, it has an organizational pattern and combines both summary and synthesis.

Review of Literature Survey

Title: A review of human activity Recognition methods

Author Michalis Vrigkas¹, Christophoros Nikou¹ and Ioannis A. Kakadiaris

Year: 16- November- 2016

Recognizing human activities from video sequences or still images is a challenging task due to problems, such as background clutter, partial occlusion, changes in scale, view point, lighting, and appearance. Many applications, including video surveillance systems, human-computer interaction, and robotics for human behavior characterization, require a multiple activity recognition system. In this work, we provide a detailed review of recent and state-of-the-art research advances in the field of human activity classification. We propose a categorization of human activity methodologies and discuss their advantages and limitations. In particular, we divide human activity classification methods in to two large categories according to whether they use data from different modalities or not. Then each of these categories is further analyzed into sub-categories, which reflect how they model human activities and what type of activities they are interested in. Moreover, we provide a comprehensive analysis of the existing, publicly available human activity classification datasets and examine the requirements for an ideal human activity recognition dataset. A comprehensive review of existing human activity classification benchmarks was also presented and we examined the challenges of data acquisition to the problem of understanding human activity. Finally, we provided the characteristics of building an ideal human activity recognition system.

Title: A Survey on Activity Recognition and Behavior Understanding in Video Surveillance

Author: Sarvesh Vishwakarma , Anupam Agrawal

Year : October 2012

This paper provides a comprehensive survey for activity recognition in video surveillance. It starts with a description of simple and complex human activity, and various applications. The applications of activity recognition are manifold, ranging from visual surveillance through content-based retrieval to human computer interaction. The organization of this paper covers all aspects of the general framework of human activity recognition. Then it summarizes and categorizes recent-published research progresses under a general framework.

In this survey, a brief overview of all preprocessing (i.e., detection, classification, and tracking) steps has been included. There are many limitations and open challenges, which we have highlighted by providing the comparison. Motion detection in dynamic scenes is a difficult task in the presence of illumination and weather changes, detection of a shadow, self-occlusion, and complete occlusion. Fast and accurate methods are still needed for segmentation techniques to affect the performance of latter stages. Description based approaches are doing well to recognize high level activities whose sub events are organized concurrently and occurring in a sequential manner in comparison to the statistical or syntactic approaches. The statistical and syntactic approaches can effectively handle the activity video polluted with noise.

Title : Trends over 5 Decades in U.S. Occupation-Related Physical Activity and Their Associations with Obesity

Author: P Timothy S. Church, Diana M. Thomas , Catrine Tudor-Locke , Peter T. Katzmarzyk, Conrad P. Earnest , Ruben Q. Rodarte , Corby K. Martin , Steven N. Blair , Claude Bouchard

Year : May 2011

The true causes of the obesity epidemic are not well understood and there are few longitudinal population-based data published examining this issue. The objective of this analysis was to examine trends in occupational physical activity during the past 5 decades and explore how these trends relate to concurrent changes in body weight in the U.S.: Analysis of energy expenditure for occupations in U.S. private industry since 1960 using data from the U.S. Bureau of Labor Statistics. Mean body weight was derived from the U.S. National Health and Nutrition Examination Surveys (NHANES). In the early 1960's almost half the jobs in private industry in the U.S. required at least moderate intensity physical activity whereas now less than 20% demand this level of energy expenditure. Since 1960 the estimated mean daily energy expenditure due to work related physical activity has dropped by more than 100 calories in both women and men. Energy balance model predicted weights based on change in occupation-related daily energy expenditure since 1960. Given a baseline weight of 76.9 kg in 1960–02, we estimated that a 142 calories reduction would result in an increase in mean weight to 89.7 kg, which closely matched the mean NHANES weight of 91.8 kg in 2003–06. The results were similar for women. Over the last 50 years in the U.S. we estimate that daily occupation-related energy expenditure has decreased by more than 100 calories, and this reduction in energy expenditure accounts for a significant portion of the increase in mean U.S. body weights for women and men.

Title : Human detection in surveillance videos and its applications - a review

Author : Sarvesh Vishwakarma·Anupam Agrawal

Year : May 2011

Detecting human beings accurately in a visual surveillance system is crucial for diverse application areas including abnormal event detection, human gait characterization, congestion analysis, person identification, gender classification and fall detection for elderly people. The first step of the detection process is to detect an object which is in motion. Object detection could be performed using background subtraction, optical flow and spatio-temporal filtering techniques. Once detected, a moving object could be classified as a human being using shape-based, texture-based or motion-based features. A comprehensive review with comparisons on available

techniques for detecting human beings in surveillance videos is presented in this paper. At the end of this paper, a discussion is made to point the future work needed to improve the human detection process in surveillance videos. These include exploiting a multi-view approach and adopting an improved model based on localized parts of the image.

8. SYSTEM STUDY

8.1 Overview of the system

we used a triaxial accelerometer in the Android phone to measure acceleration. Data from this accelerometer includes the acceleration along the x-axis, y-axis and z-axis. These axes capture the horizontal /sideways movement of the user (x-axis), upward/downward movement (y-axis), and forward/backward movement (z-axis). Demonstrates these axes relative to a user. In our work, the accelerometer generates three time series along x-axis, y-axis and z-axis, denoted by AX, AY and AZ, respectively. Each time series combines the linear acceleration due to body motion and due to gravity. We propose a digital low pass filter in order to separate the AC component from the DC component in each time series. The following Involvement steps are,

- Define a problem
- Preparing data
- Evaluating algorithms
- Improving results
- Predicting results

8.1 Objectives

The goal is to develop a machine learning model for real-time Human activity recognition, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithm.

8.2 Project Goals

Exploration data analysis of variable identification

- Loading the given dataset
- Import required libraries packages
- Analyze the general properties
- Find duplicate and missing values
- Checking unique and count values

Uni-variate data analysis

- Rename, add data and drop the data
- To specify data type

Exploration data analysis of bi-variate and multi-variate

- Plot diagram of pairplot, heatmap, bar chart and Histogram

Method of Outlier detection with feature engineering

- Pre-processing the given dataset
- Splitting the test and training dataset
- Comparing the Decision tree and Logistic regression model and random forest etc

Comparing algorithm to predict the result

- Based on the best accuracy

8.3 Scope

The scope of this project is to investigate a dataset of smartphone sensor values and meteorological sector using machine learning technique. To identifying the Human behavior of their regular activity tracing is not an easy one and try to reduces the risk factor behind the Human activity prediction, to using different algorithms and methodology based on our smartphone sensors dataset we predict the human activities.

9. Feasibility study

Data Wrangling

In this section of the report will load in the data, check for cleanliness, and then trim and clean given dataset for analysis. Make sure that the document steps carefully and justify for cleaning decisions.

Data collection

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Random Forest, logistic, Decision tree algorithms and Support vector classifier (SVC) are applied on the Training set and based on the test result accuracy, Test set prediction is done.

Preprocessing

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion need to be done.

Building the classification model

The predicting the Human activity recognition, decision tree algorithm prediction model is effective because of the following reasons: It provides better results in classification problem.

It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables

Construction of a Predictive Model

Machine learning needs data gathering have lot of past data. Data gathering have sufficient historical data and raw data. Before data pre-processing, raw data can't be used directly. It's used to preprocess then, what kind of algorithm with model. Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.

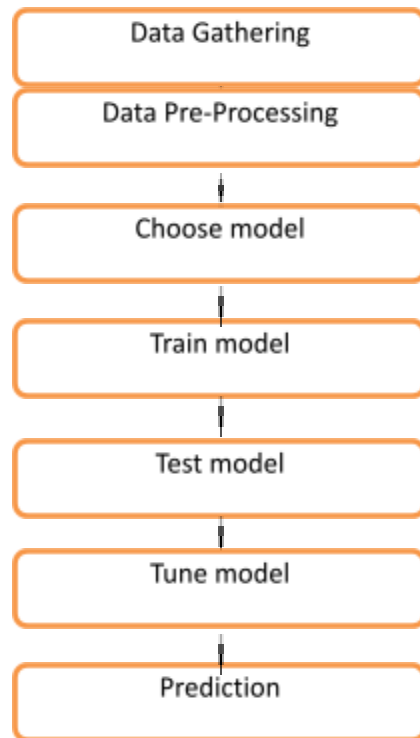


Fig: Process of dataflow diagram

10. List of Modules:

Data Preprocessing Technique

Data analysis of visualization

Comparing Algorithm with prediction in the form of best accuracy result

Deployment using Flask

List of Modules:

Data Pre-processing

Data Analysis of Visualization

Comparing Algorithm with prediction in the form of best accuracy result

Deployment Using Flask

11. Project Requirements

General:

Requirements are the basic constraints that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environment requirements
 - A. Hardware requirements
 - B. software requirements

11.1 Functional requirements:

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like sk-learn, pandas, NumPy, matplotlib and seaborn.

11.2 Non-Functional Requirements:

Process of functional steps,

1. Problem defines
2. Preparing data
3. Evaluating algorithms
4. Improving results
5. Prediction the result

12. Environmental Requirements:

1. Software Requirements:

Operating System : Windows

Tool : Anaconda with Jupyter Notebook

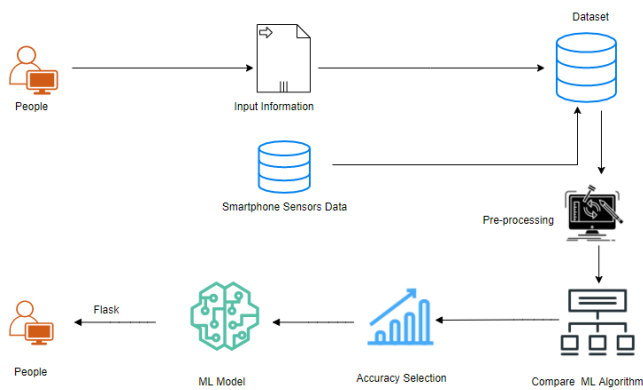
2. Hardware requirements:

Processor : Pentium IV/III

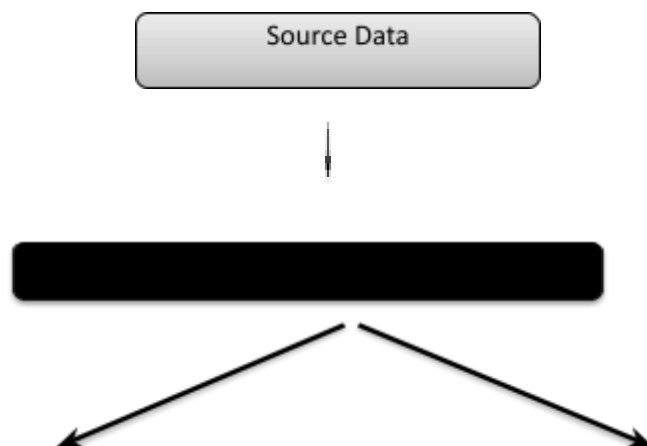
Hard disk : minimum 80 GB

RAM : minimum 2 G

13. System



14. Work flow diagram



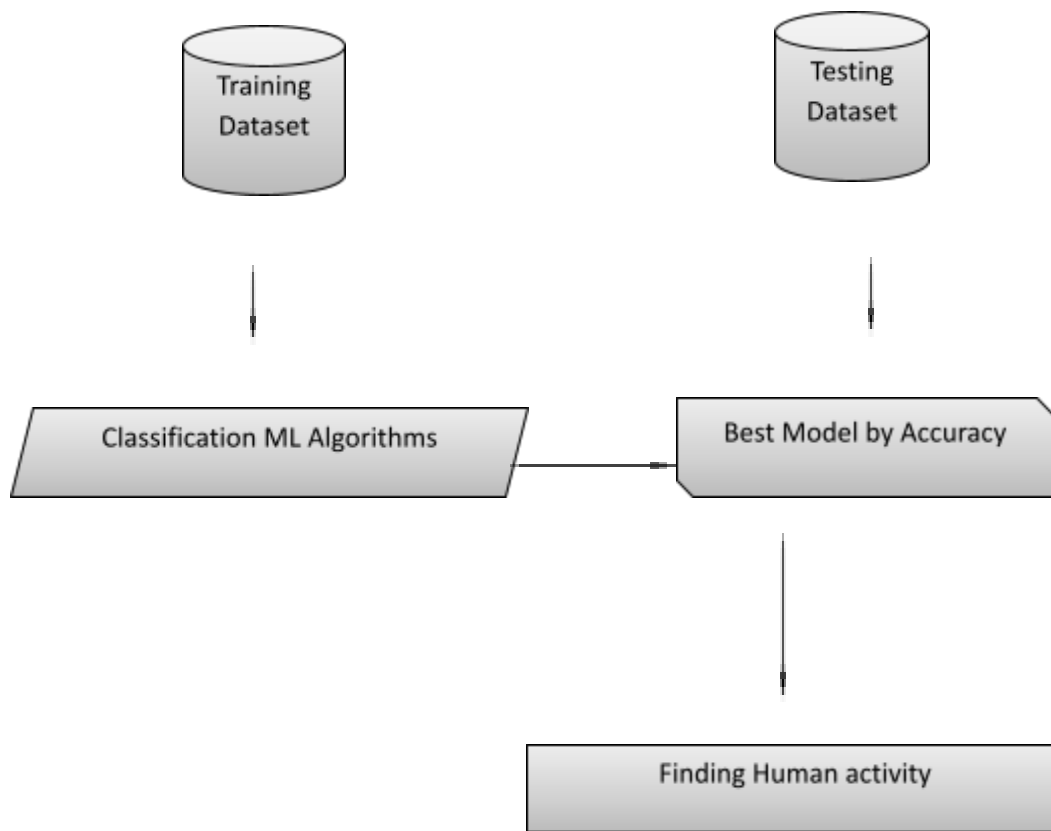


Fig: Workflow Diagram

15. Model description:

Data Pre-processing

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset.

In [22]: data.describe()

Out[22]:

	tBodyAcc.mean.X	tBodyAcc.mean.Y	tBodyAcc.mean.Z	tGravityAcc.mean.X	tGravityAcc.mean.Y	tGravityAcc.mean.Z	tBodyAcc.Jerk.mean.X	tBodyAcc.Jerk.Y
count	3609.000000	3609.000000	3609.000000	3609.000000	3609.000000	3609.000000	3609.000000	3609.000000
mean	0.274544	-0.017415	-0.109195	0.670336	0.00702	0.087888	0.075124	0.
std	0.063689	0.042589	0.056218	0.513473	0.38357	0.329519	0.179744	0.
min	-0.521000	-1.000000	-0.926000	-0.947000	-1.000000	-1.000000	-1.000000	-0.
25%	0.262000	-0.025200	-0.122000	0.812000	-0.24400	-0.119000	0.060500	-0.
50%	0.277000	-0.017200	-0.109000	0.921000	-0.14500	0.039000	0.076000	0.
75%	0.287000	-0.011000	-0.098000	0.955000	0.14500	0.204000	0.089900	0.
max	0.693000	1.000000	1.000000	0.994000	1.00000	0.999000	1.000000	0.

In [23]: pd.Categorical(data['label']).describe()

Out[23]:

	counts	freqs
LAYING	681	0.188695
SITTING	623	0.172624

Some of these sources are just simple random mistakes. Other times, there can be a deeper reason why data is missing. It's important to understand these different types of missing data from a statistics point of view. The type of missing data will influence how to deal with filling in the missing values and to detect missing values, and do some basic imputation and detailed statistical approach for dealing with missing data. Before, joint into code, it's important to understand the sources of missing data. Here are some typical reasons why data is missing:

- User forgot to fill in a field.
- Data was lost while transferring manually from a legacy database.
- There was a programming error.
- Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.

import libraries for access and functional purpose and read the given dataset

General Properties of Analyzing the given dataset

Display the given dataset in the form of data frame

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

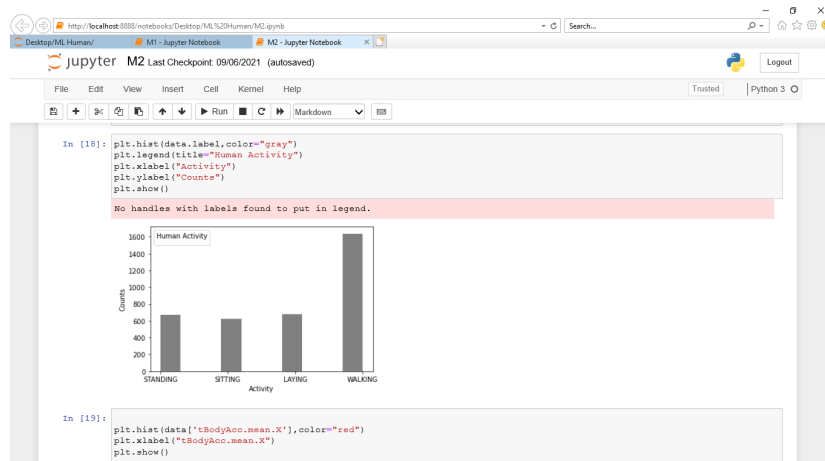
input : data

output : removing noisy data

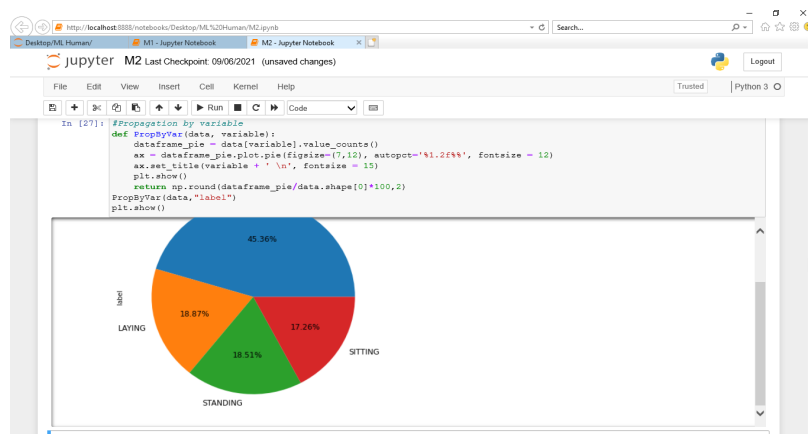
Data Validation/ Cleaning/Preparing Process

Importing the library packages with loading given dataset. To analyzing the variable identification by data shape, data type and evaluating the missing values, duplicate values. A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill while tuning model's and procedures that you can use to make the best use of validation and test datasets when evaluating your models. Data cleaning / preparing by rename the given dataset and drop the column etc. to analyze the uni-variate, bi-variate and multi-variate process. The steps and techniques for data cleaning will vary from dataset to dataset. The primary goal of data cleaning is to detect and remove errors and anomalies to increase the value of data in analytics and decision making.

Exploration data analysis of visualization



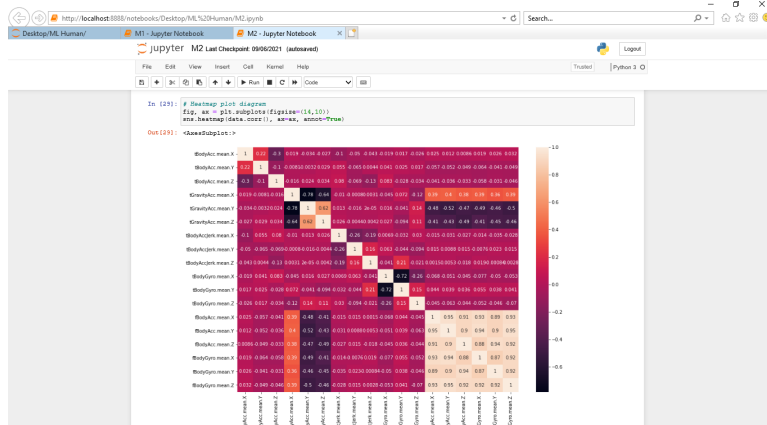
Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding.



How to chart time series data with line plots and categorical quantities with bar charts.

How to summarize data distributions with histograms and box plots.

Pre-processing refers to the transformations applied to our data before feeding way that more than one Machine Learning and Deep Learning algorithms are executed in given dataset.



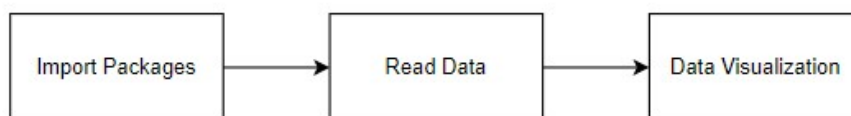
False Positives (FP): A person who will pay predicted as defaulter. When actual class is no and predicted class is yes. E.g., if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN): A person who default predicted as payer. When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

True Positives (TP): A person who will not pay predicted as defaulter. These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN): A person who default predicted as payer. These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g. if actual class says this passenger did not survive and predicted class tells you the same thing.

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input : data

output : visualized data

Comparing Algorithm with prediction in the form of best accuracy result

It is important to compare the performance of multiple different machine learning algorithms consistently and it will discover to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created.

In the example below 4 different algorithms are compared:

Logistic Regression

Decision Tree

Random Forest

Support Vector Machine

Prediction result by accuracy:

Logistic regression algorithm also uses a linear equation with independent predictors to predict a value.

True Positive Rate(TPR) = $TP / (TP + FN)$

False Positive rate(FPR) = $FP / (FP + TN)$

Accuracy: The Proportion of the total number of predictions that is correct otherwise overall how often the model predicts correctly defaulters and non-defaulters.

Accuracy calculation:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

Precision: The proportion of positive predictions that are actually correct. (When the model predicts default: how often is correct?)

$$\text{Precision} = TP / (TP + FP)$$

Recall: The proportion of positive observed values correctly predicted. (The proportion of actual defaulters that the model will correctly predict)

$$\text{Recall} = TP / (TP + FN)$$

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

General Formula:

$$F\text{-Measure} = 2TP / (2TP + FP + FN)$$

F1-Score Formula:

$$F1 \text{ Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

ALGORITHM AND TECHNIQUES

Algorithm Explanation Used Python Packages:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, we are using some of its modules like train_test_split, DecisionTreeClassifier or Logistic Regression and accuracy_score.

NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.
- Data manipulation can be done easily with data frames.

Matplotlib:

- Data visualization is a useful way to help with identify the patterns from given dataset.
- Data manipulation can be done easily with data frames.

Logistic Regression

It is a statistical method for analyzing a data set in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes). The goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent variables. Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).

```
In [74]: #LabelEncoding
from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i]).astype(int)

In [77]: data.label.unique()
Out[77]: array([2, 1, 0, 3])

In [78]: data.head()
Out[78]:
```

	lBodyAcc.mean.X	lBodyAcc.mean.Y	lBodyAcc.mean.Z	lGravityAcc.mean.X	lGravityAcc.mean.Y	lGravityAcc.mean.Z	lBodyAcc.jerk.mean.X	lBodyAcc.jerk.mean.Y	lBodyAcc.jerk.mean.Z
0	0.279	-0.0196	-0.1180	0.968	-0.144	0.0931	0.0822	-0.01	-0.01
1	0.277	-0.0127	-0.1030	0.969	-0.152	0.0813	0.0714	0.01	0.01
2	0.277	-0.0147	-0.1070	0.969	-0.154	0.0818	0.0776	0.01	0.01
3	0.290	0.0371	-0.0017	0.945	-0.263	-0.0214	0.0607	-0.06	-0.06
4	0.276	-0.0170	-0.1110	0.954	-0.263	-0.0114	0.0761	0.01	0.01

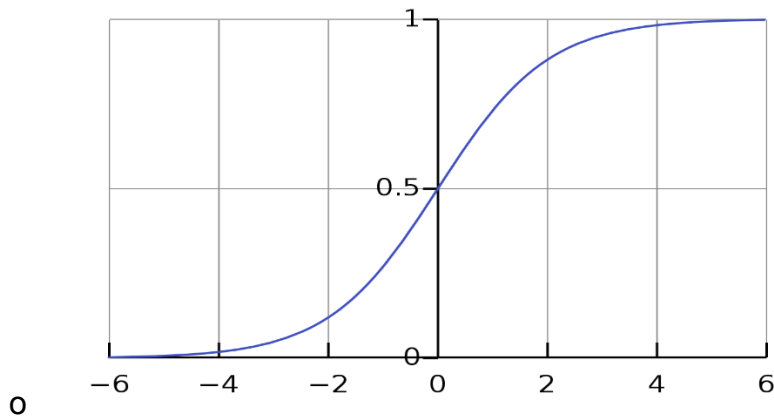
```
In [79]: # Now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef, cohen_kappa_score, accuracy_score
```

In other words, the logistic regression model predicts $P(Y=1)$ as a function of X .
Logistic regression Assumptions:

- o Logistic regression predicts the output of a categorical dependent variable. therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1.**
- o Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems.**

Logistic Function (Sigmoid Function):

- o The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- o It maps any real value into another value within a range of 0 and 1.
- o The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.



$$1 / (1 + e^{-\text{value}})$$

- o In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

- o The dependent variable must be categorical in nature.
- o The independent variable should not have multi-collinearity.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- o We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- o In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; 0 \text{ for } y=0, \text{ and infinity for } y=1$$

- o But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

Type of Logistic Regression:

On the basis of the categories, Logistic Regression can be classified into three types:

- o **Binary or Binomial**

In such a kind of classification, a dependent variable will have only two possible types either 1 and 0. For example, these variables may represent success or failure, yes or no, win or loss etc.

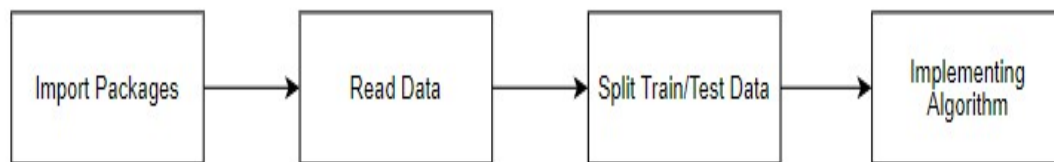
- o **Multinomial**

In such a kind of classification, dependent variable can have 3 or more possible unordered types or the types having no quantitative significance. For example, these variables may represent “Type A” or “Type B” or “Type C”.

- o **Ordinal**

In such a kind of classification, dependent variable can have 3 or more possible **ordered** types or the types having a quantitative significance. For example, these variables may represent “poor” or “good”, “very good”, “Excellent” and each category can have the scores like 0,1,2,3.

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

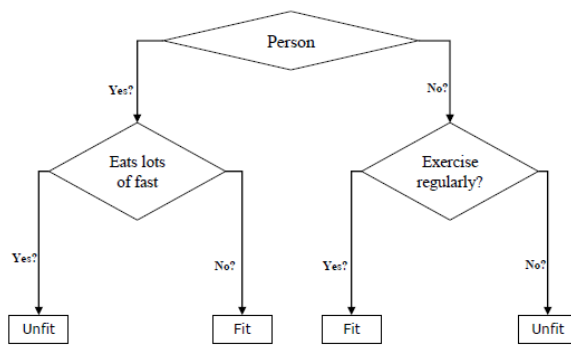
input : data

output: getting accuracy

Decision Tree

Introduction to Decision Tree

In general, Decision tree analysis is a predictive modelling tool that can be applied across many areas. Decision trees can be constructed by an algorithmic approach that can split the dataset in different ways based on different conditions. Decisions trees are the most powerful algorithms that falls under the category of supervised algorithms.



In the above decision tree, the question are decision nodes and final outcomes are leaves. We have the following two types of decision trees.

- **Classification decision trees** – In this kind of decision trees, the decision variable is categorical. The above decision tree is an example of classification decision tree.
- **Regression decision trees** – In this kind of decision trees, the decision variable is continuous.

Implementing Decision Tree Algorithm

Gini Index

Important Terminology related to Decision Tree

```

Classification report DecisionTreeClassifier Results:
      precision    recall  f1-score   support

     0       1.00      1.00      1.00       204
     1       0.98      0.95      0.96       187
     2       0.91      0.89      0.90       203
     3       1.00      1.00      1.00       493

 accuracy avg      0.95      0.95      0.95      1087
 macro avg      0.95      0.95      0.95      1087
 weighted avg      0.94      0.95      0.94      1087

Accuracy result of DecisionTree is: 94.2142173992414

Confusion Matrix result of DecisionTree Classifier is:
[[204  0  0  0]
 [ 10 187  0  0]
 [ 10 179  0  0]
 [  0  0 493  0]
 Sensitivity : 1.0
 Specificity : 1.0

Cross validation test results of accuracy:
[0.9404321 0.9174218 0.9013185 0.9390581 0.9376688]
Accuracy result of DecisionTree Classifier is: 93.5937912332391

Accuracy comparison of Human activity
  
```

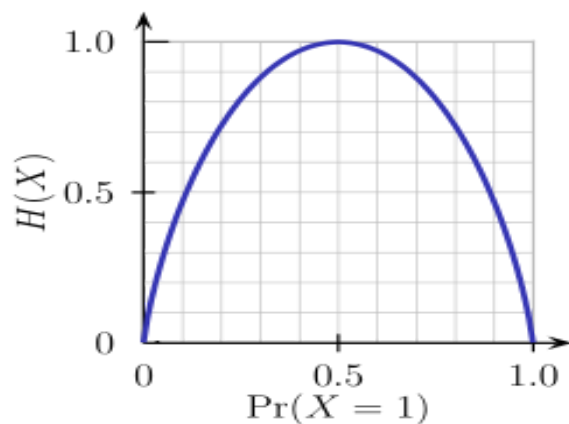
Assumptions while creating Decision Tree

Below are some of the assumptions we make while using Decision tree:

- In the beginning, the whole training set is considered as the **root**.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

Entropy

Entropy is a measure of the randomness in the information being processed. The higher the entropy, the harder it is to draw any conclusions from that information. Flipping a coin is an example of an action that provides information that is random.



From the above graph, it is quite evident that the entropy $H(X)$ is zero when the probability is either 0 or 1. The Entropy is maximum when the probability is 0.5 because it projects perfect randomness in the data and there is no chance of perfectly determining the outcome.

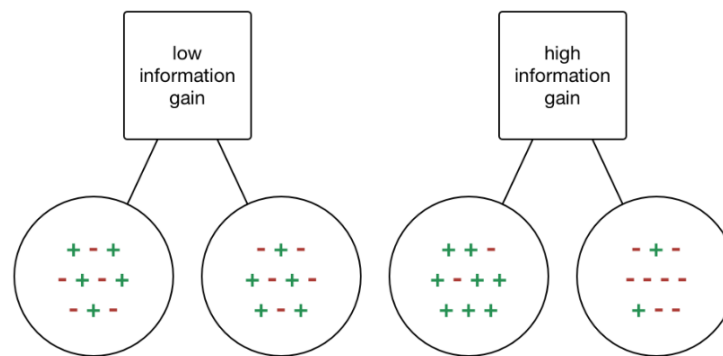
Mathematically Entropy for 1 attribute is represented as:

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

Where **S** → **Current state**, and **P_i** → **Probability of an event i of state S or Percentage of class i in a node of state S.**

Information Gain

Information gain or **IG** is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.



$$Information\ Gain = Entropy(before) - \sum_{j=1}^K Entropy(j, after)$$

Information gain is a decrease in entropy. It computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values. ID3 (Iterative Dichotomies) decision tree algorithm uses information gain.

Mathematically, IG is represented as:

$$Information\ Gain(T,X) = Entropy(T) - Entropy(T, X)$$

In a much simpler way, we can conclude that:

$$Information\ Gain = Entropy(before) - \sum_{j=1}^K Entropy(j, after)$$

Information Gain

Where “before” is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.

Prediction

After building a decision tree, we need to make a prediction about it. Basically, prediction involves navigating the decision tree with the specifically provided row of data.

We can make a prediction with the help of recursive function, as did above. The same prediction routine is called again with the left or the child right nodes.

Assumptions

The following are some of the assumptions we make while creating decision tree –

- While preparing decision trees, the training set is as root node.
- Decision tree classifier prefers the features values to be categorical. In case if you want to use continuous values then they must be done discretized prior to model building.
- Based on the attribute's values, the records are recursively distributed.
- Statistical approach will be used to place attributes at any node position i.e.as root node or internal node.

MODULE DIAGRAM

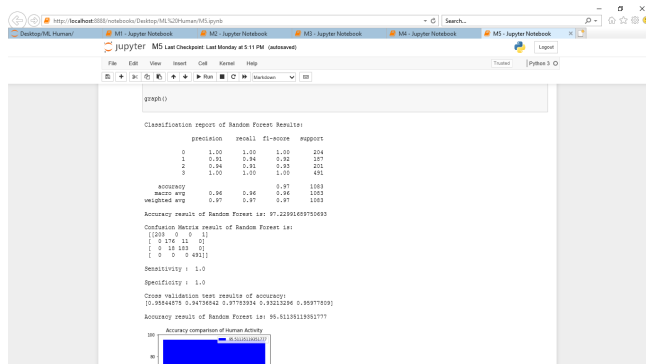


GIVEN INPUT EXPECTED OUTPUT

- o input : data
- o output : getting accuracy

Random forest Algorithm

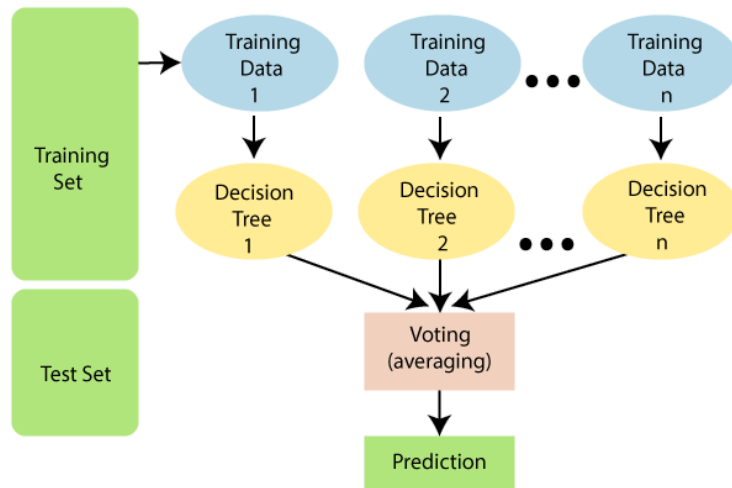
Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning**, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.



As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

- o There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- o The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm:

- o It takes less training time as compared to other algorithms.
- o It predicts output with high accuracy, even for the large dataset it runs efficiently.
- o It can also maintain accuracy when a large proportion of data is missing.

Implementation in Scikit-learn

For each decision tree, Scikit-learn calculates a nodes importance using Gini Importance, assuming only two child nodes (binary tree):

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)}$$

- $ni_{sub(j)}$ = the importance of node j
- $w_{sub(j)}$ = weighted number of samples reaching node j
- $C_{sub(j)}$ = the impurity value of node j
- $left(j)$ = child node from left split on node j
- $right(j)$ = child node from right split on node j

$sub()$ is being used as subscript isn't available in Medium

See method `compute_feature_importances` in `_tree.pyx`

The importance for each feature on a decision tree is then calculated as:

$$fi_i = \frac{\sum_{j: \text{node } j \text{ splits on feature } i} ni_j}{\sum_{k \in \text{all nodes}} ni_k}$$

- $fi_{sub(i)}$ = the importance of feature i
- $ni_{sub(j)}$ = the importance of node j

These can then be normalized to a value between 0 and 1 by dividing by the sum of all feature importance values:

The final feature importance, at the Random Forest level, is its average over all the trees. The sum of the feature's importance value on each tree is calculated and divided by the total number of trees:

- RF_{fi} = the importance of feature i calculated from all trees in the Random Forest model
- $norm_{fi}$ = the normalized feature importance for i in tree j
- T = total number of trees

See method `feature_importances_` in [forest.py](#)

Notation was inspired by this [StackExchange thread](#) which I found incredibly useful for this post.

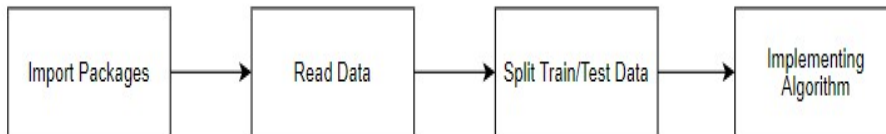
Implementation in Spark

For each decision tree, Spark calculates a feature's importance by summing the gain, scaled by the number of samples passing through the node:

- $f_{i,j}$ = the importance of feature i
- s_j = number of samples reaching node j

- $C_{sub(j)}$ = the impurity value of node
- $normfi_{sub(ij)}$ = the normalized feature importance for i in tree j

MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input : data

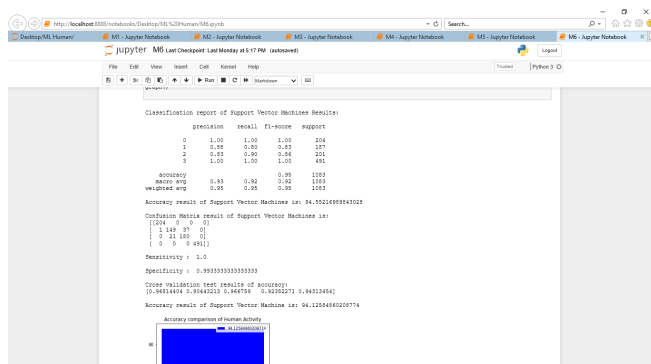
output : getting accuracy

Support Vector Machines.

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

The objective of applying SVMs is to find the best line in two dimensions or the best hyperplane in more than two dimensions in order to help us separate our space into classes. The hyperplane (line) is found through the **maximum margin**, i.e., the maximum distance between data points of both classes.

Don't you think the definition and idea of SVM look a bit abstract? No worries, let me explain in details.



MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

input : data

output : getting accuracy

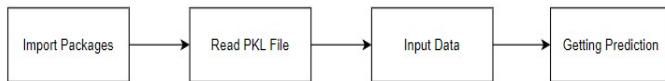
16. Deploy

Flask (web framework)

Flask is based on Werkzeug, Jinja2 and inspired by Sinatra Ruby framework, available under BSD licence. It was developed at pocoo by Armin Ronacher. Although Flask is rather young compared to most Python frameworks, it holds a great promise and has already gained popularity among Python web developers. Let's take a closer look into Flask, so-called "micro" framework for Python.



MODULE DIAGRAM



GIVEN INPUT EXPECTED OUTPUT

FEATURES:

Flask was designed to be **easy to use and extend**. The idea behind Flask is to build a solid foundation for web applications of different complexity. From then on you are free to **plug in any extensions** you think you need. Also you are free to build your own modules. Flask is great for all kinds of projects. It's especially good for prototyping. Flask depends on two external libraries: the Jinja2 template engine and the Werkzeug WSGI toolkit.

Advantages of Flask:

- Higher compatibility with latest technologies.
- Technical experimentation.
- Easier to use for simple cases.
- Codebase size is relatively smaller.
- High scalability for simple applications.
- Easy to build a quick prototype.
- Routing URL is easy.
- Easy to develop and maintain applications.

17. Coding

Module 1: Data validation and pre-processing technique

```
#import library packages
import pandas as pd
import numpy as np
```

```
#load given dataset
data=pd.read_csv("human.csv")
```

```
data
```

```
data.head()
```

```
data.tail()
```

```
#shape
data.shape
```

```
# size
data.size
```

```
#Checking datatype and information about dataset
data.info()
```

```
#To describe the dataframe
data.describe()
```

```
#show columns
data.columns
```

```
data.head()
```

```
data.isnull()
```

```
#null values
```

```
data.isnull().sum()
```

```
data.duplicated()
```

```
data.duplicated().sum()
```

```
# drop duplivcate values  
data.dropna()
```

```
data.duplicated().sum()
```

```
#correlation  
data.corr()
```

```
data.activity.unique()
```

```
data["label"]=data.activity.map({"STANDING":"STANDING","SITTING":"SITTING","LAYI  
NG":"LAYING","WALKING":"WALKING","WALKING_DOWNSTAIRS":"WALKING","WALK  
ING_UPSTAIRS":"WALKING"})
```

```
data.label.unique()
```

```
del data['activity']
```

```
data.describe()
```

```
pd.Categorical(data['label']).describe()
```

```
len(data['fBodyGyro.mean.X'].unique())
```

```
len(data['fBodyGyro.mean.Y'].unique())
```

```
len(data['fBodyGyro.mean.Z'].unique())
```

```
data["label"].value_counts()
```

```
data['tBodyAcc.mean.X'].unique()
```

```
data['tBodyAcc.mean.Y'].unique()
```

```
data['tBodyAcc.mean.Z'].unique()
```

```
from sklearn.preprocessing import LabelEncoder
var_mod = ['tBodyAcc.mean.X', 'tBodyAcc.mean.Y', 'tBodyAcc.mean.Z',
           'tGravityAcc.mean.X', 'tGravityAcc.mean.Y', 'tGravityAcc.mean.Z',
           'tBodyAccJerk.mean.X', 'tBodyAccJerk.mean.Y', 'tBodyAccJerk.mean.Z',
           'tBodyGyro.mean.X', 'tBodyGyro.mean.Y', 'tBodyGyro.mean.Z',
           'fBodyAcc.mean.X', 'fBodyAcc.mean.Y', 'fBodyAcc.mean.Z',
           'fBodyGyro.mean.X', 'fBodyGyro.mean.Y', 'fBodyGyro.mean.Z', 'label']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i]).astype(int)
```

In []:

```
data.head()
```

In []:

```
data["label"].unique()
```

Module 2: Exploration data analysis of visualization and training a model by given attributes

```
#import library packages
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
#loading dataset
data=pd.read_csv("human.csv")

data.head(10)

data.shape

data.size

data.isnull().sum()

data.duplicated()

data.dropna()

data.shape

data.columns

data.activity.unique()

data["label"]=data.activity.map({"STANDING":"STANDING","SITTING":"SITTING","LAYI
NG":"LAYING","WALKING":"WALKING","WALKING_DOWNSTAIRS":"WALKING","WALK
ING_UPSTAIRS":"WALKING"})
data.columns

data.label.unique()

del data['activity']

plt.hist(data.label,color="gray")
plt.legend(title="Human Activity")
plt.xlabel("Activity")
plt.ylabel("Counts")
plt.show()
```

```
plt.hist(data['tBodyAcc.mean.X'],color="red")
plt.xlabel("tBodyAcc.mean.X")
plt.show()
```

```
plt.hist(data['tBodyAcc.mean.Y'],color="blue")
plt.xlabel("tBodyAcc.mean.Y")
plt.show()
```

```
plt.hist(data['tBodyAcc.mean.Z'],color="green")
plt.xlabel("tBodyAcc.mean.Z")
plt.show()
```

```
plt.show()
```

```
#Propagation by variable
def PropByVar(data, variable):
    dataframe_pie = data[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(7,12), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable + ' \n', fontsize = 15)
    plt.show()
    return np.round(dataframe_pie/data.shape[0]*100,2)
PropByVar(data,"label")
plt.show()
```

```
data.columns
```

```
plt.figure(figsize=(10,6))
plt.plot(data['tBodyGyro.mean.X'])
plt.show()
```

```
# Heatmap plot diagram
fig, ax = plt.subplots(figsize=(14,10))
sns.heatmap(data.corr(), ax=ax, annot=True)
```

```
boxplot = data.boxplot(column=['tBodyAcc.mean.Y'], by="label",figsize=(8,6),
fontsize=15)
plt.show()
```

```
plt.figure(figsize=(12,8))
```

```
plt.scatter(data["tBodyGyro.mean.Y"],data["label"],color="red")
```

```
data1=data.iloc[:,9]  
data2=data.iloc[:,9:-1]
```

```
a=pd.plotting.scatter_matrix(data1, alpha=0.05, figsize=(18,10), diagonal='hist')  
axis = 'off'
```

```
plt.show()
```

```
a=pd.plotting.scatter_matrix(data2, alpha=0.05, figsize=(18,10), diagonal='hist')  
axis = 'off'
```

```
plt.show()
```

```
plt.figure(figsize=(12,7))  
sns.countplot(x='label',data=data)
```

```
data.columns
```

```
from sklearn.preprocessing import LabelEncoder  
var_mod = ['tBodyAcc.mean.X', 'tBodyAcc.mean.Y', 'tBodyAcc.mean.Z',  
           'tGravityAcc.mean.X', 'tGravityAcc.mean.Y', 'tGravityAcc.mean.Z',  
           'tBodyAccJerk.mean.X', 'tBodyAccJerk.mean.Y', 'tBodyAccJerk.mean.Z',  
           'tBodyGyro.mean.X', 'tBodyGyro.mean.Y', 'tBodyGyro.mean.Z',  
           'fBodyAcc.mean.X', 'fBodyAcc.mean.Y', 'fBodyAcc.mean.Z',  
           'fBodyGyro.mean.X', 'fBodyGyro.mean.Y', 'fBodyGyro.mean.Z', 'label']  
le = LabelEncoder()  
for i in var_mod:  
    data[i] = le.fit_transform(data[i]).astype(int)
```

```
data.head()
```

Module 3 : Performance measurements of Logistic regression algorithms

```
#import library packages  
import pandas as p
```

```
import matplotlib.pyplot as plt
import seaborn as s
import numpy as n

#Load given dataset
data = p.read_csv("human.csv")

import warnings
warnings.filterwarnings('ignore')

data.head()

data.shape

data.size
len(data.columns)
data.isnull().sum()
data.dropna()
data.corr()
data.columns
data.activity.unique()
#mapping
data["label"]=data.activity.map({"STANDING":"STANDING","SITTING":"SITTING","LAYI
NG":"LAYING","WALKING":"WALKING",'WALKING_DOWNSTAIRS':"WALKING",'WALK
ING_UPSTAIRS':"WALKING"})
data.label.unique()
data.head()
del data["activity"]
data.columns
```

```

#labelEncoding
from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i]).astype(int)
data.label.unique()
data.head()
# now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score
X = data.drop(labels='label', axis=1)
#Response variable
y = data.loc[:, 'label']
#We'll use a test size of 30%. We also stratify the split on the response variable, which
is very important to do because there are so few fraudulent transactions.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)

```

Logistic Regression :

```

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score

logR= LogisticRegression()

logR.fit(X_train,y_train)

predictLR = logR.predict(X_test)

```

```

print("")
print('Classification report of Logistic Regression Results:')
print("")
print(classification_report(y_test,predictLR))
x = (accuracy_score(y_test,predictLR)*100)

print('Accuracy result of Logisticregression is:', x)

print("")

cm2=confusion_matrix(y_test,predictLR)
print('Confusion Matrix result of Logistic Regression is:\n',cm2)
print("")
sensitivity2 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
print('Sensitivity : ', sensitivity2 )
print("")
specificity2 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
print('Specificity : ', specificity2)
print("")

accuracy = cross_val_score(logR, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Logistic Regression is:",accuracy.mean() * 100)
LR=accuracy.mean() * 100

```

```

def graph():
    import matplotlib.pyplot as plt
    data=[LR]
    alg="Logistic Regression"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color="b")
    plt.title("Accuracy comparison Human Activity",fontsize=15)
    plt.legend(b,data,fontsize=9)

```

graph()

```

TN = cm2[1][0]
FN = cm2[0][0]
TP = cm2[1][1]
FP = cm2[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)

```

```

print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm2, title='Confusion matrix-LR', cmap=plt.cm.Blues):
    target_names=[""]
    plt.imshow(cm2, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = n.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm2=confusion_matrix(y_test, predictLR)
print('Confusion matrix-LR:')
print(cm2)
plot_confusion_matrix(cm2)

```

Module 4 : Performance measurements of DecisionTree:

```

import pandas as pd
import numpy as np

```

```
import matplotlib.pyplot as plt
import seaborn as sns
#Warning
import warnings
warnings.filterwarnings('ignore')
data=pd.read_csv("human.csv")
data.head()
data.shape
data.shape
data.duplicated().sum()
data.dropna()
data.shape
data.isnull()
data.isnull().sum()
data.columns

data.activity.unique()
data["label"]=data.activity.map({'STANDING':'STANDING', 'SITTING':'SITTING',
'LAYING':'LAYING', 'WALKING':'WALKING',
'WALKING_DOWNSTAIRS':'WALKING','WALKING_UPSTAIRS':'WALKING'})
data.label.unique()
data.head()
del data["activity"]
data.describe()
data.corr()
#labelencoding
from sklearn.preprocessing import LabelEncoder

var_mod=["label"]
le = LabelEncoder()
for i in var_mod:
```

```

data[i] = le.fit_transform(data[i]).astype(int)
data
data.label.unique()

# let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score
X = data.drop(labels='label', axis=1)
#Response variable
y = data.loc[:, 'label']
#We'll use a test size of 30%. We also stratify the split on the response variable, which
is very important to do because there are so few fraudulent transactions.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)
#We'll use a test size of 30%. We also stratify the split on the response variable, which
is very important to do because there are so few fraudulent transactions.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)

```

DecisionTree:

```

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

DT=DecisionTreeClassifier()

DT.fit(X_train,y_train)

predictDT = DT.predict(X_test)

```

```

print("")
print('Classification report DecisionTree classifier Results:')
print("")
print(classification_report(y_test,predictDT))

print("")
x = (accuracy_score(y_test,predictDT)*100)

print('Accuracy result of DecisionTree is:', x)
print("")

cm2=confusion_matrix(y_test,predictDT)
print('Confusion Matrix result of DecissionTree Classifier is:\n',cm2)
print("")
sensitivity2 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
print('Sensitivity : ', sensitivity2 )
print("")
specificity2 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
print('Specificity : ', specificity2)
print("")

accuracy = cross_val_score(DT, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of DecisionTree Classifier is:",accuracy.mean() * 100)
dt=accuracy.mean() * 100

```

```

def graph():
    import matplotlib.pyplot as plt
    data=[dt]
    alg="Decision Tree"
    plt.figure(figsize=(5,5))
    b=plt.bar(alg,data,color="b")
    plt.title("Accuracy comparison of Human activity",fontsize=15)
    plt.legend(b,data,fontsize=9)

```

```

graph()
TN = cm2[1][0]
FN = cm2[0][0]
TP = cm2[1][1]
FP = cm2[0][1]
print("True Positive :",TP)
print("True Negative :",TN)
print("False Positive :",FP)
print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)

```

```

print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

def plot_confusion_matrix(cm2, title='Confusion matrix-DT', cmap=plt.cm.Blues):
    target_names=[]
    plt.imshow(cm2, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm2=confusion_matrix(y_test, predictDT)
print('Confusion matrix-DT:')
print(cm2)
plot_confusion_matrix(cm2)

```

Module 5 : Performance measurements of Random Forest algorithms

```
import pandas as pd
```

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
data = pd.read_csv("human.csv")
data.head()
data.shape
data.size
data.columns
data.isnull()
data.isnull().sum()
data.duplicated()
data.dropna()
data.shape
data.corr()
data.columns
data.activity.unique()
#mapping
data["label"]=data.activity.map({"STANDING":"STANDING","SITTING":"SITTING","LAYI
NG":"LAYING","WALKING":"WALKING","WALKING_DOWNSTAIRS":"WALKING","WALK
ING_UPSTAIRS":"WALKING"})
data.head()
data.label.unique()
data.head()
del data["activity"]
data.columns
#labelEncoding
from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
```

```

for i in var_mod:
    data[i] = le.fit_transform(data[i]).astype(int)
data.head()
# model, so now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score
X = data.drop(labels='label', axis=1)
#Response variable
y = data.loc[:, 'label']
#We'll use a test size of 30%. We also stratify the split on the response variable, which
is very important to do because there are so few fraudulent transactions.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)

```

Random Forest:

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import cross_val_score

rfc = RandomForestClassifier()

rfc.fit(X_train,y_train)

predictR = rfc.predict(X_test)

print("")
print('Classification report of Random Forest Results:')
print("")

print(classification_report(y_test,predictR))

```

```

x = (accuracy_score(y_test,predictR)*100)

print('Accuracy result of Random Forest is:', x)
print("")
cm1=confusion_matrix(y_test,predictR)
print('Confusion Matrix result of Random Forest is:\n',cm1)
print("")
sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity : ', sensitivity1 )
print("")
specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity : ', specificity1)
print("")

accuracy = cross_val_score(rfc, X, y, scoring='accuracy')
print('Cross validation test results of accuracy:')
print(accuracy)
#get the mean of each fold
print("")
print("Accuracy result of Random Forest is:",accuracy.mean() * 100)
RFC=accuracy.mean() * 100

```

```

def graph():
    import matplotlib.pyplot as plt
    data=[RFC]
    alg="Random orest"
    plt.figure(figsize=(5,5))

```

```
b=plt.bar(alg,data,color=("b"))  
plt.title("Accuracy comparison of Human Activity")  
plt.legend(b,data,fontsize=9)
```

graph()

```
TN = cm1[1][0]  
FN = cm1[0][0]  
TP = cm1[1][1]  
FP = cm1[0][1]  
print("True Positive :",TP)  
print("True Negative :",TN)  
print("False Positive :",FP)  
print("False Negative :",FN)  
print("")  
TPR = TP/(TP+FN)  
TNR = TN/(TN+FP)  
FPR = FP/(FP+TN)  
FNR = FN/(TP+FN)  
print("True Positive Rate :",TPR)  
print("True Negative Rate :",TNR)  
print("False Positive Rate :",FPR)  
print("False Negative Rate :",FNR)  
print("")  
PPV = TP/(TP+FP)  
NPV = TN/(TN+FN)  
print("Positive Predictive Value :",PPV)  
print("Negative predictive value :",NPV)
```

```

def plot_confusion_matrix(cm2, title='Confusion matrix-RF', cmap=plt.cm.Blues):
    target_names=[""]
    plt.imshow(cm2, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm2=confusion_matrix(y_test, predictR)
print('Confusion matrix-RF:')
print(cm2)
plot_confusion_matrix(cm2)

```

model building

```

import joblib
joblib.dump(rfc, 'RFC.pkl')

```

Module 6 : Performance measurements of Support Vector Machines algorithms

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data = pd.read_csv("human.csv")
data.head()
data.shape

```

```

data.size
data.columns
data.isnull()
data.isnull().sum()
data.duplicated()
data.dropna()
data.shape
data.corr()
data.columns
data.activity.unique()
#mapping
data["label"]=data.activity.map({"STANDING":"STANDING","SITTING":"SITTING","LAYI
NG":"LAYING","WALKING":"WALKING","WALKING_DOWNSTAIRS":"WALKING","WALK
ING_UPSTAIRS":"WALKING"})
data.head()
data.label.unique()
data.head()
del data["activity"]
data.columns
#labelEncoding
from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i]).astype(int)
data.head()
# now let's evaluate its performance on the test set.
from sklearn.metrics import confusion_matrix, classification_report, matthews_corrcoef,
cohen_kappa_score, accuracy_score, average_precision_score, roc_auc_score
X = data.drop(labels='label', axis=1)
#Response variable

```

```
y = data.loc[:, 'label']  
#We'll use a test size of 30%. We also stratify the split on the response variable, which  
is very important to do because there are so few fraudulent transactions.  
from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,  
stratify=y)
```

Support Vector Machines:

```
from sklearn.svm import SVC  
from sklearn.metrics import accuracy_score, confusion_matrix  
from sklearn.model_selection import cross_val_score
```

```
s = SVC()
```

```
s.fit(X_train, y_train)
```

```
predicts = s.predict(X_test)
```

```
print("")  
print('Classification report of Support Vector Machines Results:')  
print("")
```

```
print(classification_report(y_test, predicts))  
x = (accuracy_score(y_test, predicts)*100)
```

```
print('Accuracy result of Support Vector Machines is:', x)  
print("")  
cm2=confusion_matrix(y_test, predicts)  
print('Confusion Matrix result of Support Vector Machines is:\n', cm2)  
print("")
```

```
sensitivity1 = cm2[0,0]/(cm2[0,0]+cm2[0,1])
```

```
print('Sensitivity : ', sensitivity1 )
```

```
print("")
```

```
specificity1 = cm2[1,1]/(cm2[1,0]+cm2[1,1])
```

```
print('Specificity : ', specificity1)
```

```
print("")
```

```
accuracy = cross_val_score(s, X, y, scoring='accuracy')
```

```
print('Cross validation test results of accuracy:')
```

```
print(accuracy)
```

```
#get the mean of each fold
```

```
print("")
```

```
print("Accuracy result of Support Vector Machine is:",accuracy.mean() * 100)
```

```
S=accuracy.mean() * 100
```

```
def graph():
```

```
    import matplotlib.pyplot as plt
```

```
    data=[S]
```

```
    alg="Support Vector Machine"
```

```
    plt.figure(figsize=(5,5))
```

```
    b=plt.bar(alg,data,color=("b"))
```

```
    plt.title("Accuracy comparison of Human Activity")
```

```
    plt.legend(b,data,fontsize=9)
```

```
graph()
```

```
TN = cm2[1][0]
```

```
FN = cm2[0][0]
```

```
TP = cm2[1][1]
```

```
FP = cm2[0][1]
```

```
print("True Positive :",TP)
```

```
print("True Negative :",TN)
```

```
print("False Positive :",FP)
```

```

print("False Negative :",FN)
print("")
TPR = TP/(TP+FN)
TNR = TN/(TN+FP)
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
print("True Positive Rate :",TPR)
print("True Negative Rate :",TNR)
print("False Positive Rate :",FPR)
print("False Negative Rate :",FNR)
print("")
PPV = TP/(TP+FP)
NPV = TN/(TN+FN)
print("Positive Predictive Value :",PPV)
print("Negative predictive value :",NPV)

```

```

def plot_confusion_matrix(cm2, title='Confusion matrix-SVM', cmap=plt.cm.Blues):
    target_names=[]
    plt.imshow(cm2, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(target_names))
    plt.xticks(tick_marks, target_names, rotation=45)
    plt.yticks(tick_marks, target_names)
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

```

```

cm2=confusion_matrix(y_test, predicts)
print('Confusion matrix-SVM:')
print(cm2)

```

```
plot_confusion_matrix(cm2)
```

Flask deploy

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
import joblib
```

```
app = Flask(__name__)
model = joblib.load('RFC.pkl')
```

```
@app.route('/')
def home():
    return render_template('index.html')
```

```
@app.route('/predict',methods=['POST'])
def predict():
    """
    For rendering results on HTML GUI
    """
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model.predict(final_features)

    output = prediction[0]
    print(output)
    if output == 0:
        output = 'Laying'
    elif output == 1:
        output = 'Sitting'
    elif output == 2:
        output = 'Standing'
    else:
        output = 'Walking'
```

```
return render_template('index.html', prediction_text='HUMAN : {}'.format(output))
```

```
if __name__ == "__main__":  
    app.run(host="localhost", port=6002)
```

HTML & CSS

```
<!DOCTYPE html>  
<html >  
<!--From https://codepen.io/frytyler/pen/EGdtg-->  
<head>  
    <meta charset="UTF-8">  
    <title>TITLE</title>  
<link rel="stylesheet" href="{{ url_for('static', filename='css/bootstrap.min.css') }}">  
    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'  
type='text/css'>  
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'  
type='text/css'>  
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'  
type='text/css'>  
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'  
rel='stylesheet' type='text/css'>  
<style>  
.back{  
    background-image: url('{{ url_for('static', filename='image/img.jpg') }}');  
}  
.white{  
    color:white;  
}  
.space{  
    margin:10px 30px;  
    padding:15px 10px;
```

```
background: palegreen;
width:auto;
}
.gap{
padding:10px 20px;
}
```

```
</style>
```

```
</head>
```

```
<body class="back">
```

```
<div>
```

```
<div class="jumbotron">
```

```
<h1 style="text-align:center"> HUMAN ACTIVITY </h1>
```

```
</div>
```

```
<!-- Main Input For Receiving Query to our ML -->
```

```
<form class="form-group" action="{{ url_for('predict')}}"method="post">
```

```
<div class="container">
```

```
<div class="row">
```

```
<div class="gap col-md-6">
```

```
<label class="white" for=""> tBodyAcc.mean.X </label>
```

```
<input type="number" class="space form-control" step="0.01" name="
tBodyAcc.mean.X " placeholder=" tBodyAcc.mean.X " required="required" /><br>
```

```
<label class="white" for=""> tBodyAcc.mean.Y </label>
```

```
<input type="number" class="space form-control" step="0.01" name="
tBodyAcc.mean.Y " placeholder=" tBodyAcc.mean.Y " required="required" /><br>
```

```
<label class="white" for=""> tBodyAcc.mean.Z </label>
```

```
        <input type="number" class="space form-control" step="0.01" name="
tBodyAcc.mean.Z " placeholder=" tBodyAcc.mean.Z " required="required" /><br>
```

```
        <label class="white" for=""> tGravityAcc.mean.X </label>
        <input type="number" class="space form-control" step="0.01" name="
tGravityAcc.mean.X " placeholder=" tGravityAcc.mean.X " required="required" /><br>
        <label class="white" for=""> tGravityAcc.mean.Y </label>
        <input type="number" class="space form-control" step="0.01" name="
tGravityAcc.mean.Y " placeholder=" tGravityAcc.mean.Y " required="required" /><br>
        <label class="white" for=""> tGravityAcc.mean.Z </label>
        <input type="number" class="space form-control" step="0.01" name="
tGravityAcc.mean.Z " placeholder=" tGravityAcc.mean.Z " required="required" /><br>
```

```
        <label class="white" for=""> tBodyAccJerk.mean.X </label>
        <input type="number" class="space form-control" step="0.01" name="
tBodyAccJerk.mean.X " placeholder=" tBodyAccJerk.mean.X " required="required"
/><br>
```

```
        <label class="white" for=""> tBodyAccJerk.mean.Y </label>
        <input type="number" class="space form-control" step="0.01" name="
tBodyAccJerk.mean.Y " placeholder=" tBodyAccJerk.mean.Y " required="required"
/><br>
```

```
        <label class="white" for=""> tBodyAccJerk.mean.Z </label>
        <input type="number" class="space form-control" step="0.01" name="
tBodyAccJerk.mean.Z " placeholder=" tBodyAccJerk.mean.Z " required="required"
/><br>
```

```
    </div>
```

```
    <div class="gap col-md-6">
```

```
        <label class="white" for=""> tBodyGyro.mean.X </label>
        <input type="number" class="space form-control" step="0.01" name="
tBodyGyro.mean.X " placeholder=" tBodyGyro.mean.X " required="required" /><br>
```

```
<label class="white" for=""> tBodyGyro.mean.Y </label>
<input type="number" class="space form-control" step="0.01" name="
tBodyGyro.mean.Y " placeholder=" tBodyGyro.mean.Y " required="required" /><br>
<label class="white" for=""> tBodyGyro.mean.Z </label>
<input type="number" class="space form-control" step="0.01" name="
tBodyGyro.mean.Z " placeholder=" tBodyGyro.mean.Z " required="required" /><br>
```

```
<label class="white" for=""> fBodyAcc.mean.X</label>
<input type="number" class="space form-control" step="0.01" name="
fBodyAcc.mean.X" placeholder=" fBodyAcc.mean.X" required="required" /><br>
<label class="white" for=""> fBodyAcc.mean.Y</label>
<input type="number" class="space form-control" step="0.01" name="
fBodyAcc.mean.Y" placeholder=" fBodyAcc.mean.Y" required="required" /><br>
<label class="white" for=""> fBodyAcc.mean.Z</label>
<input type="number" class="space form-control" step="0.01" name="
fBodyAcc.mean.Z" placeholder=" fBodyAcc.mean.Z" required="required" /><br>
```

```
<label class="white" for=""> fBodyGyro.mean.X </label>
<input type="number" class="space form-control" step="0.01" name="
fBodyGyro.mean.X " placeholder=" fBodyGyro.mean.X " required="required" /><br>
<label class="white" for=""> fBodyGyro.mean.Y </label>
<input type="number" class="space form-control" step="0.01" name="
fBodyGyro.mean.Y " placeholder=" fBodyGyro.mean.Y " required="required" /><br>
<label class="white" for=""> fBodyGyro.mean.Z </label>
<input type="number" class="space form-control" step="0.01" name="
fBodyGyro.mean.Z " placeholder=" fBodyGyro.mean.Z " required="required" /><br>
```

```
</div>
```

```
</div>
```

```

</div>

<div style="padding:2% 35%">
  <button type="submit" class="btn btn-success btn-block"
style="width:350px;padding:20px">Predict</button>
</div>

</form>

<br>
<br>
<div style="background:skyblue;padding:2% 40%">
  {{ prediction_text }}
</div>
</div>

</body>
</html>

```

18. Conclusion

The analytical process started from data cleaning and processing, missing value, exploratory analysis and finally model building and evaluation. The best accuracy on public test set is higher accuracy score is will be find out. This application can help to find the Human Activity Based on the Smartphone sensor.

19. Future Work

Human Activity Recognition connect with AI model.

To automate this process by show the prediction result in web application or desktop application.

To optimize the work to implement in Artificial Intelligence environment.

