HUMAN ACTIVITY RECOGNITION WITH SMARTPHONES USING MACHINE LEARNING

A Project work submitted to

Acharya Nagarjuna University

Department of Computer Science and Engineering

In partial fulfillment of the requirements for

The award of the degree of

Master of Science in
Computational Data Science (CDS)

by

PARASA SIRISHA

Regd. No. Y24DS20025

Under the guidance of

Dr. U. SURYA KAMESWARI., M.Sc., M. Tech., Ph.D.
Assistant Professor

Department of computer science & engineering

University College of Sciences

Acharya Nagarjuna University

DEPARTMENT OF COMPUTER SCIENCE &
ENGINEERING UNIVERSITY COLLEGE OF SCIENCES
ACHARYA NAGARJUNA UNIVERSITY
Nagarjuna Nagar, Guntur,
Andhra Pradesh, India

June 2025

ACHARYA NAGARJUNA UNIVERSITY

NAGARJUNA NAGAR, GUNTUR

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that this project entitled "HUMAN ACTIVITY RECOGNITION WITH SMARTPHONES USING MACHINE LEARNING" is a Bonafide record of the project work done and submitted by P. SIRISHA (Y24DS20025) during the year 2023 - 2025 in partial fulfillment of the requirements for the award of degree of Master of Computational Data Science (MSc-CDS) in the department of Computer Science & Engineering. I certify that he carries this project as an independent project under my guidance.

Head of the Department
(Prof. K. Gangadhara Rao)

Project Guide
(Dr. U.Surya Kameswari)

External Examiner

DECLARATION

I hereby declare that the entire thesis work entitled " **HUMAN ACTIVITY RECOGNITION WITH SMARTPHONES USING MACHINE LEARNING"** is being submitted to the Department of **Computer Science and Engineering, University College of Sciences, Acharya Nagarjuna University,** in partial fulfillment of the requirement for the award of the degree of **Master of Computational Data Science (MSc-CDS)** is a Bonafide work of my own, carried out under the supervision of **Dr. U. Surya Kameswari,** Faculty Member, Department of Computer Science & Engineering, Acharya Nagarjuna University.

I further declare that the Project, either in part or full, has not been submitted earlier by me or others for the award of any degree in any University.

PARASA SIRISHA

Reg. No. Y24DS20025

ACKNOWLEDGEMENT

Undertaking this Project has been a truly life-changing experience for me and it would not have been possible to do without the support and guidance that I received from many people.

I would like to first say a very big thank you to my supervisor **Dr. U. SURYA KAMESWARI** for all the support and encouragement he gave me. Her friendly guidance and expert advice have been invaluable throughout all stages of the work. Without her guidance and constant feedback this Project work not have been achievable.

I would also wish to express my gratitude to **Prof. K. Gangadhara Rao** for extended discussions and valuable suggestions which have contributed greatly to the improvement of the thesis.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of Department which helped us in success fully completing our project work. Also, I would like to extend our sincere regards to all the non-teaching staff of the department for their timely support.

I must also thank my parents and friends for the immense support and help during this project. Without their help, completing this project would have been very difficult.

HUMAN ACTIVITY RECOGNITION WITH SMARTPHONES USING MACHINE LEARNING

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 classify data into activity like Walking, walking up stairs, walking down stairs, sitting, standing, laying are recognized. Sensor data generated using its accelerometer and gyroscope, the sensor signals (accelerometer and gyroscope) were pre-processed by applying noise filters. The sensor acceleration signal, which has gravitational and body motion components, was separated using a Butterworth low-pass filter into body acceleration and gravity. The gravitational force is assumed to have only low frequency components. a vector of features was obtained by calculating variables from the time and frequency domain. The aim is to predict machine learning based techniques for Human Activity Recognition results in best accuracy. The analysis of dataset by supervised machine learning techniquen(SMLT) to capture several information‘s like, variable identification, uni-variate analysis, bi-variate and multi-variate analysis, missing value treatments and analyze the data validation, data cleaning/preparing and data visualization will be done on the entire given dataset. To propose a machine learning-based method to accurately predict the stock price Index value by prediction results in the form of stock price increase or stable state best accuracy from comparing supervise classification machine learning algorithms. Additionally, to compare and discuss the performance of various machine learning algorithms from the given transport traffic department dataset with evaluation. dataset with evaluation classification report, identify the confusion matrix and to categorizing data from priority and the result shows that the effectiveness of the proposed machine learning algorithm technique can be compared with best accuracy with precision, Recall and F1 Score.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ML - Machine Learning

AI - Artificial Intelligence

IoT - Internet of Things

SVM - Support Vector Machine

RF - Random Forest ANN - Artificial Neural Network

LSTM - Long Short-Term Memory

CNN - Convolutional Neural Network

PCA - Principal Component Analysis

KNN - K-Nearest Neighbours

MLP - Multilayer Perceptron

LR - Logistic Regression

DT - Decision Tree

NB - Naive Bayes

CNN - Convolutional Neural Network

RNN - Recurrent Neural Network

LSTM - Long Short-Term Memory

RMSE - Root Mean Square Error

MAE - Mean Absolute Error

R^2 - Coefficient of Determination

GIS - Geographic Information System

CHAPTER 1

INTRODUCTION

INTRODUCTION

These days smartphones became an increasing number of famous in human each day lifestyles. Most people used it looking for information, looking films, playing games and having access to the social community, however, there have been many useful studies on smartphones. Activity Recognition (AR) is one of the maximum vital technology behind many packages on a smartphone consisting of heal1.th tracking, fall detection, context-aware cellular packages, human survey gadget, and home automation and many others., Smartphone based activity recognition system is an energetic vicinity of studies because they could lead to new kinds of smart applications.

The rapid proliferation of smartphones has made them an integral part of everyday life, not just for communication, entertainment, and information access, but increasingly for research and intelligent applications. One particularly promising field is Human Activity Recognition (HAR), which leverages the sensors embedded in smartphones—such as accelerometers and gyroscopes—to monitor, identify, and interpret human movements. These capabilities are at the heart of many emerging use cases including health tracking, elderly care, fall detection, fitness monitoring, and context-aware mobile apps. As smartphones become more ubiquitous, their use in HAR systems offers a cost-effective and accessible solution for recognizing physical activities like walking, standing, sitting, or climbing stairs.

This integration of smart devices with our daily routines reflects the broader vision of ambient intelligence—where computing systems learn from and adapt to human behavior. In this context, smartphones can automatically and continuously track user behavior, identify abnormal patterns, and assist users in making better lifestyle or health-related decisions. For example, in assisted living and rehabilitation support, HAR systems can detect irregular activities or emergency situations (like falls) in real-time and alert caregivers, thus enhancing both safety and quality of care without constant human supervision.

From a technical standpoint, HAR using smartphones depends heavily on processing the signals generated by inertial sensors. These signals are captured in three-dimensional space (x, y, z axes) and are highly granular—often generating tens or hundreds of data points per second. To transform this raw data into meaningful activity labels, machine learning (ML) algorithms

are employed. Traditional ML approaches, such as Decision Trees, Support Vector Machines, and Random Forests, require careful feature engineering—extracting meaningful attributes like signal magnitude area, energy, and standard deviation from raw signals.

However, there are challenges. For instance, resource limitations on mobile hardware (processing power, battery life, memory) make it impractical to use computation-heavy models like Random Forests or large-scale neural networks without optimization. Thus, lightweight models or those that can be efficiently trained and deployed—like Decision Trees or pruned versions of ensemble methods—are often preferred in on-device systems. Moreover, training HAR models with minimal user input (like a few minutes of recorded activity) and still achieving good accuracy is a desirable goal, reducing user burden and enabling rapid personalization.

Another growing trend in HAR research is the adoption of deep learning. Unlike classical models, deep neural networks can automatically learn discriminative features from raw sensor data without hand-crafted input. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory networks (LSTMs) have shown strong performance in HAR by capturing both spatial and temporal patterns in activity data. These models, when carefully optimized, can outperform traditional approaches in terms of accuracy and robustness, especially in more complex multi-class activity recognition tasks.
In summary, Human Activity Recognition using smartphones is an evolving and impactful domain that sits at the intersection of mobile computing, sensor technology, and artificial intelligence. It promises to make daily living smarter and safer, especially in healthcare, home automation, and personal wellness. Despite hardware limitations, ongoing innovations in algorithm optimization, low-power computing, and deep learning are pushing the boundaries of what can be achieved with just a smartphone in your pocket.

More recently, deep learning methods have been achieving success on HAR problems given their ability to automatically learn higher-order features.

.

1.1 Background and Motivation

In today's digitally driven world, understanding and interpreting human behavior through technology has become a significant area of research, particularly in the context of Human Activity Recognition (HAR). HAR refers to the process of identifying physical activities such as walking, sitting, standing, running, or climbing stairs using data collected from sensors. Initially, this was predominantly achieved using external hardware such as video surveillance systems or specialized wearable devices. Although effective in controlled environments, these methods often face challenges related to scalability, cost, limited portability, privacy intrusions, and the inconvenience of wearing dedicated sensors all day.

The emergence and widespread adoption of smartphones have revolutionized this domain. Modern smartphones are equipped with a rich set of sensors—accelerometers, gyroscopes, magnetometers, GPS, and more—that can continuously and unobtrusively capture a user's movement patterns throughout the day. These devices are not only powerful enough to perform real-time data processing but are also portable, widely available, and familiar to users. This makes smartphones an ideal platform for implementing practical and scalable HAR systems that can operate in everyday environments without requiring users to change their behavior or carry additional devices.

The motivation behind leveraging smartphones for HAR lies in their pervasive presence, computational capabilities, and energy-efficient sensors. Since most individuals carry a smartphone at all times, especially in their pockets or hands, it becomes feasible to monitor physical activities passively and consistently. This opens up new opportunities across various domains. In healthcare, smartphone-based HAR can support continuous health monitoring, enabling timely detection of abnormal patterns such as falls or sedentary behavior in the elderly. In the fitness industry, it can provide real-time feedback and personalized workout plans. In smart homes and environments, HAR can improve automation, enhance user experience, and increase security by making devices context-aware.

Moreover, the exponential growth of machine learning (ML) has significantly enhanced the ability to model, analyze, and predict human activities from sensor data. ML algorithms can identify subtle and complex patterns in large volumes of time-series data collected from smartphones. These models learn from examples and can adapt to different user profiles, thereby enabling highly personalized and robust HAR systems. Techniques like Decision

Trees, Random Forests, Support Vector Machines (SVM), and deep learning models such as LSTM and CNN have been successfully applied to HAR, demonstrating impressive accuracy levels.

This project is driven by the growing demand for intelligent systems that can accurately recognize human activity using smartphones, with minimal computational cost and high efficiency. The goal is to design and evaluate machine learning models that can process sensor data to classify human activities in real-time. By bridging the gap between raw sensory input and actionable insights, this work aims to contribute to the development of smart, user-centric applications in healthcare, fitness, safety, and smart living environments. Ultimately, the project envisions HAR systems that are not only accurate and efficient but also scalable and accessible to the general population.

1.2 Problem Statement

Now a days maximum of peoples are using smart phones, with the help of smartphone sensors like accelerometer and gyro, we can find the activities of the human, which is help to find out how The Activity people is in active like walking, running, sitting .Some people are not active in real life, those peoples are have obesity and some other health issues .We can use this also some security purpose and Transportation.

In today's technology-driven world, smartphones have become ubiquitous and are equipped with various sensors such as accelerometers and gyroscopes that can continuously monitor user movements. These sensors generate vast amounts of data that can be leveraged to recognize and classify human activities such as walking, sitting, standing, or lying down.

However, accurately identifying and classifying these activities from raw sensor data poses significant challenges due to noise in data, variation in movement patterns among individuals, and similarity between certain activities. Manual analysis of such data is not feasible, and traditional rule-based approaches are often ineffective or non-scalable.

There is a growing need for intelligent systems that can automatically learn from data and recognize human activities in real-time with high accuracy. Machine learning offers powerful

tools for building such systems, but the choice of algorithm, data preprocessing techniques, and feature engineering methods significantly impact the overall performance.

Therefore, this project aims to design and evaluate machine learning models for robust human activity recognition using smartphone sensor data. The goal is to develop a system that can classify activities accurately, compare different ML approaches, and provide insights into the most effective techniques for real-world deployment.

In recent years, Human Activity Recognition (HAR) has emerged as a vital area of research due to its wide-ranging applications in healthcare, fitness, assisted living, human-computer interaction, and smart environments. The ability to automatically recognize human activities like walking, sitting, standing, running, or climbing stairs using data from various sensors can significantly enhance the development of context-aware and intelligent systems.

Traditionally, HAR systems have relied on specialized hardware such as video surveillance cameras or wearable sensors. While these approaches can provide high levels of accuracy, they are often accompanied by significant drawbacks. These include the high cost of equipment, the intrusiveness of wearable devices, dependence on fixed infrastructure, and privacy concerns related to video monitoring. These limitations reduce the feasibility of large-scale or long-term deployments in real-world environments.

On the other hand, smartphones have become ubiquitous in daily life. Most modern smartphones are equipped with multiple embedded sensors, including accelerometers, gyroscopes, and magnetometers, that can continuously record motion data. This presents a promising, cost-effective, and non-intrusive alternative for performing HAR without additional hardware. However, extracting meaningful information from raw smartphone sensor data is not straightforward. The data is often noisy, high-dimensional, and varies significantly across users, activities, phone positions, and environments.

Moreover, designing a HAR system that works in real-time on smartphones introduces additional challenges. Mobile devices have limited processing power, battery life, and storage capacity compared to dedicated computers. Many existing machine learning models used for activity recognition are computationally intensive and not optimized for mobile deployment.

Given these limitations, there is a clear need for an efficient, lightweight, and accurate HAR system that can be implemented on smartphones using advanced machine learning algorithms. Such a system should:

- Accurately identify multiple human activities based on raw sensor data,
- Work in real-time with minimal delay,
- Be robust to variations in user behavior, sensor noise, and device placement,
- And consume minimal computational resources to be feasible on a smartphone.

This project aims to solve this problem by collecting and processing sensor data from smartphones and applying machine learning techniques to train and evaluate models that can classify various physical activities with high accuracy. The ultimate goal is to develop a real-time, practical HAR solution that works seamlessly in daily life and enhances user interaction, health monitoring, and context-aware services.

1.3 Objectives of the Project

The main objective of this project is to develop an efficient and accurate Human Activity Recognition (HAR) system using smartphone sensor data and machine learning techniques. The project aims to recognize and classify different physical activities (such as walking, sitting, standing, lying, etc.) in real-time.

The primary objective of this project is to design and develop an efficient and accurate Human Activity Recognition (HAR) system that utilizes sensor data collected from smartphones in conjunction with machine learning algorithms. By harnessing the capabilities of sensors such as accelerometers, gyroscopes, and magnetometers, the system aims to detect and classify a range of human physical activities—such as walking, sitting, standing, and lying—in real-time. This technology has widespread practical value, especially in areas such as healthcare monitoring, fitness tracking, elder care, and smart environments, where recognizing user activity in a timely and reliable manner is crucial.

To begin with, the system must be capable of collecting and preprocessing raw sensor data from the smartphone. This involves capturing continuous time-series data from motion sensors and performing essential data preparation tasks. The raw data, often noisy and unstructured, must be cleaned, normalized, and segmented into meaningful windows. These steps ensure that the data is suitable for analysis and can be transformed into feature vectors for machine learning models to process effectively.

Another key objective is to explore, implement, and compare a variety of machine learning models to determine the most suitable algorithm for activity classification. The project focuses on well-known supervised learning techniques including Logistic Regression, Decision Tree, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Linear Discriminant Analysis (LDA). Each model will be evaluated based on performance metrics such as accuracy, confusion matrix, and ROC-AUC scores. This comparison will help identify the model that provides the best balance between prediction accuracy, processing speed, and computational efficiency.

The project also aims to build a robust and adaptable recognition system capable of handling real-world challenges. These include inconsistencies in sensor readings, user-specific movement variations, and the diverse ways in which a smartphone may be carried or positioned. The system should demonstrate resilience to these variables and maintain a high level of classification performance across different users and scenarios, making it a dependable solution for real-life deployment.

Visual interpretation of results is another important objective. To facilitate clear understanding of model performance, the system will generate graphical outputs such as confusion matrices and ROC curves. These visualizations will help in identifying misclassifications, understanding model confidence levels, and comparing multiple models side by side. Such visual tools are valuable not only for system evaluation but also for communicating results to stakeholders.

Furthermore, the project seeks to develop a system that is lightweight and scalable for real-time implementation on smartphones. The final solution should be optimized for low latency and minimal resource consumption, making it suitable for mobile platforms with limited computing power and battery life. Real-time classification is essential in applications such as fall detection or continuous health monitoring, where any delay in activity recognition could impact user safety or the effectiveness of the service.

Lastly, the system will highlight its practical significance through the demonstration of real-world use cases. These include applications in healthcare, such as monitoring elderly patients; in fitness, such as tracking workouts or posture; and in smart homes, where the system can trigger automatic responses based on user activity. By showcasing these applications, the project aims to underline the value and versatility of smartphone-based HAR solutions in improving quality of life and enabling context-aware computing.

1.4 Scope of the Work

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 there regular activity tracing is not a 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.

At the core of the project is the utilization of smartphone sensor data, particularly from in-built components like the accelerometer and gyroscope. These sensors are capable of capturing detailed time-series motion data across three axes (X, Y, Z), which is essential for detecting and differentiating between various physical activities such as walking, sitting, standing, and lying down. The project intentionally excludes any dependency on external hardware or devices, making the approach both cost-effective and suitable for large-scale or personal use scenarios, especially in mobile and wearable contexts.

A critical part of the project involves data preprocessing and feature extraction. Raw sensor data is typically noisy, high-dimensional, and unstructured. To make this data useful for machine learning, the system must perform a series of preprocessing tasks, including cleaning, normalization, and segmentation. Feature engineering is then used to extract meaningful attributes such as mean, standard deviation, root mean square, and frequency components. These features provide a compact and informative representation of the data that enhances the classification process.

The scope also includes the implementation and evaluation of multiple supervised machine learning models, including algorithms such as Logistic Regression, Decision Tree, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Linear Discriminant Analysis (LDA). Each model will be trained and tested on the preprocessed dataset to determine its ability to

classify different human activities accurately. The aim is to identify the model that offers the best combination of speed, accuracy, and resource efficiency for deployment on mobile platforms.

Additionally, the project focuses on model evaluation and visualization, where each model's performance is assessed using quantitative metrics like accuracy, confusion matrix, and ROC-AUC score. Visualization tools such as confusion matrices and ROC curves will be generated to facilitate intuitive understanding of each model's classification strengths and weaknesses. These visualizations will also guide improvements in model tuning and selection.

An optional extension of the scope includes designing the system with real-time compatibility in mind. While the primary dataset used may be pre-recorded, the system will be structured in a way that allows future adaptation to real-time use cases, where data is streamed live from a smartphone and classified on the fly. This involves ensuring that the models are computationally efficient and capable of operating with minimal latency on resource-constrained mobile devices.

The final component within the scope of the project is the exploration of practical use-case applications. The HAR system is envisioned to be applicable in several real-world scenarios, including fitness monitoring, health and wellness tracking, fall detection for elderly individuals, and context-aware automation in smart environments. These applications demonstrate the wide-reaching impact of HAR systems when integrated into everyday life.

CHAPTER 2

LITERATURE SURVEY

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 10 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.

Finally, the survey outlines the key characteristics required to build a robust and reliable human activity recognition system. These include high recognition accuracy, real-time performance, adaptability across users and environments, and the ability to generalize from limited training data. Such systems should be efficient enough to run on embedded devices or smartphones, paving the way for widespread use in daily-life monitoring, health tracking, smart environments, and interactive systems.

Title: A Survey on Activity Recognition and Behavior Understanding in Video Surveillance
Author: Sarvesh Vishwakarma·, Anupam Agrawal
Year: October 2012

This paper offers a comprehensive and structured survey of human activity recognition techniques specifically tailored for video surveillance applications. The study begins by distinguishing between simple and complex human activities and their diverse real-world applications. These include areas such as visual surveillance, content-based retrieval, and human-computer interaction, all of which demand accurate interpretation of human motion and behavior. The authors organize their review around a general framework for activity recognition that encapsulates the complete pipeline, including detection, classification, and tracking of human movements.

The paper presents a taxonomy of recent advancements in the field, categorizing them based on methodology and functionality within this framework. One of the key highlights is the discussion of challenges associated with motion detection in dynamic scenes—particularly when facing variations in illumination, weather conditions, presence of shadows, self-occlusion, and complete occlusion. Such conditions severely impact the accuracy and reliability of recognition systems. The authors emphasize the continuing need for fast and robust segmentation techniques, as their performance directly affects subsequent stages in the recognition process.

Furthermore, the study compares different modeling approaches and notes that description-based methods—which focus on capturing the temporal structure and hierarchical relationships between sub-events—are particularly effective in recognizing high-level, complex human activities. These methods outperform traditional statistical and syntactic approaches, especially when events occur concurrently or in sequence. However, statistical and syntactic methods still have significant value, particularly in handling noisy video data, where their probabilistic and structured nature allows for greater resilience. The survey concludes by identifying open challenges and suggesting areas where further research is required to develop more accurate, scalable, and real-time-capable activity recognition systems for surveillance environments.

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

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

Year: May 2011

Despite increasing attention to the rising rates of obesity in the United States, the true underlying causes remain only partially understood, particularly due to the limited availability of long-term, population-level data. This study aims to shed light on one potential contributor: the decline in occupational physical activity over the past five decades. Using historical data from the U.S. Bureau of Labor Statistics and health-related data from the National Health and Nutrition Examination Surveys (NHANES), the analysis explores how shifts in job-related energy expenditure correlate with rising body weights in the U.S. population. In the early 1960s, nearly 50% of private-sector jobs in the U.S. required at least moderate levels of physical exertion. However, by the early 2000s, fewer than 20% of jobs met this criterion, indicating a substantial decline in the physical demands of work.

The findings reveal that the estimated average daily energy expenditure from occupational activity has decreased by more than 100 kilocalories for both men and women since 1960. Using an energy balance model, researchers predicted that this reduction—specifically a decrease of approximately 142 kilocalories per day—would increase the average body weight from a baseline of 76.9 kg to roughly 89.7 kg. This prediction aligns closely with actual NHANES measurements from 2003–2006, which reported a mean body weight of 91.8 kg. The results for women showed a similar trend, confirming that reduced physical activity in the workplace is a significant factor contributing to the overall increase in body weight across the population.

The study concludes that changes in occupational energy expenditure have played a major role in the obesity epidemic observed over the last 50 years. While factors like diet, lifestyle, and genetics also contribute, the decline in physically active jobs appears to be a substantial and measurable influence. These insights highlight the need for public health strategies to address not just caloric intake but also physical activity, especially given the increasingly sedentary nature of modern employment.

Title: Human detection in surveillance videos and its applications - a review
Author: Sarvesh Vishwakarma. Anupam Agrawal
Year: May 2011

Accurate detection of human beings in visual surveillance systems is foundational to a wide range of critical applications, including abnormal event detection, human gait analysis, congestion monitoring in public spaces, individual identification, gender classification, and fall detection—particularly for elderly individuals. Effective human detection serves as the cornerstone for enabling higher-level tasks such as behavior recognition and security threat assessment. The process typically begins with motion-based object detection, identifying moving entities within the video feed. This is commonly achieved through techniques such as background subtraction, which isolates moving objects from static backgrounds; optical flow, which analyzes movement patterns between consecutive frames; and spatio-temporal filtering, which enhances relevant motion cues over space and time.

Once a moving object is detected, the next critical step is to determine whether the object is a human being. For this, several approaches are used, typically based on shape features (e.g., body contour, silhouette), texture features (e.g., clothing patterns or surface variations), or motion features (e.g., walking rhythm or arm-leg coordination). Each method has its strengths and limitations depending on the scene complexity, lighting conditions, and camera angle.

The paper provides a comprehensive review of the various techniques used in human detection for surveillance applications, comparing their performance, robustness, and computational efficiency. It highlights that while significant progress has been made, current systems still face several challenges—particularly in dynamic environments with variable lighting, occlusions, or crowded scenes. Toward the end of the study, the authors propose future research directions to enhance human detection accuracy. These include employing multi-view detection strategies to mitigate occlusion and perspective issues and developing more advanced models that utilize localized body parts, allowing the system to focus on key discriminative regions even under partial visibility.

Such improvements are expected to make human detection systems more reliable and effective in real-world deployments, especially in security, healthcare, and smart city monitoring applications.

Title: A Study of Vision based Human Motion Recognition and Analysis.
Author: Geetanjali Vinayak Kale, Varsha Hemant Patil
Year: 2 - July-December 2016

Vision-based human motion recognition has emerged as a highly active area of research, capturing the interest of scientists and engineers due to both its inherent challenges and its wide range of practical applications. These applications extend from basic gesture recognition, which is often used in gaming and user interface control, to more complex behavior understanding in areas such as video surveillance, healthcare monitoring, and human-robot interaction. The increasing demand for systems that can interpret human motion accurately in real-time has driven substantial progress in the development of robust and intelligent motion recognition techniques.

The referenced study offers a comprehensive overview of the applications, frameworks, and methodologies used in human motion recognition. It details the general pipeline of a motion recognition system, typically consisting of components such as data acquisition, motion representation, feature extraction, classification, and interpretation. A major focus is placed on the representation of human motion, which plays a critical role in determining the effectiveness of the recognition system. Various approaches—ranging from model-based to appearance-based representations—are compared in terms of their computational efficiency, scalability, and accuracy under varying environmental conditions.

Additionally, the paper discusses multiple recognition methods, highlighting their strengths and weaknesses. Among the most successful techniques are hierarchical approaches, which decompose complex activities into simpler, structured sub-actions. These methods are particularly effective for understanding interactions and multi-step behaviours. Also notable are methods adapted from other domains, such as the bag-of-words model and Hidden Markov Models (HMM), originally successful in text and speech recognition. These techniques have been successfully repurposed for temporal modelling in human motion sequences, demonstrating their flexibility and power.

As research in this area continues to evolve, there is a growing potential for deploying commercial products equipped with intelligent vision-based systems. Such systems can significantly enhance automation, safety, and user experience in various sectors, from smart surveillance and security to health diagnostics and assistive technologies.

Title: Sensor-Based Human Activity Recognition Systems: A Survey
Authors: Bulling, A., Blanke, U., Schiele, B.
Year: 2014

This comprehensive survey offers a deep and systematic examination of sensor-based Human Activity Recognition (HAR) systems, with particular emphasis on the use of body-worn and wearable sensors. The study begins by classifying the different types of sensors commonly employed in HAR systems, such as accelerometers, gyroscopes, and magnetometers, each of which plays a distinct role in capturing physical movement data. These sensors, often embedded in smartphones, smartwatches, or dedicated wearable devices, provide continuous streams of time-series data that reflect various human activities.

The paper delves into the full system architecture pipeline involved in HAR, beginning with data acquisition, where raw sensor signals are collected under real-world or controlled conditions. Following this, it details the preprocessing techniques used to filter noise and handle missing values, ensuring data quality before it enters the feature extraction phase. Here, meaningful statistical, temporal, and frequency-domain features are computed to represent patterns of human motion. These features are then used to train machine learning models that perform activity classification, distinguishing between actions such as walking, running, sitting, standing, or more complex behaviors.

One of the significant contributions of the survey is its focus on real-world deployment challenges. The authors emphasize the practical implications of applying HAR in everyday settings, especially in healthcare monitoring, personal fitness tracking, rehabilitation, and context-aware computing. Unlike lab-controlled datasets, real-world environments introduce challenges such as inconsistent sensor placement, varying user behaviors, device heterogeneity, and data imbalance.

The paper also addresses key system constraints such as energy efficiency, crucial for wearable and mobile devices that rely on battery power; privacy, especially in healthcare applications where sensitive personal data is involved; and generalization, the ability of a system trained on one group of users to perform accurately on new, unseen users or activities.

In conclusion, the survey not only reviews existing technologies and methodologies but also provides valuable insights and future directions. It suggests that future HAR systems should focus on adaptive learning models, personalized calibration techniques, and lightweight computation strategies to enhance their effectiveness, scalability, and ease of deployment in diverse and dynamic real-world scenarios.

Title: A Review of Activity Recognition Using Mobile Phones
Authors: Shoaib, M. et al.
Year: 2013

This review extensively examines smartphone-based Human Activity Recognition (HAR) systems, emphasizing the use of built-in mobile sensors such as accelerometers, gyroscopes, and magnetometers. The study highlights how smartphones, due to their portability, ubiquity, and computing capabilities, have become powerful platforms for continuous activity monitoring in real-world environments. A central focus of the paper is on multi-sensor fusion techniques, which combine data from multiple sensors to improve recognition accuracy and robustness.

The paper evaluates several stages of the HAR pipeline including data acquisition, preprocessing, segmentation, feature extraction, and classification using machine learning algorithms. The authors also explore the challenges of data labeling, especially when real-time annotations are needed for supervised learning models, and discuss methods for obtaining reliable labels in uncontrolled environments. Additionally, the review compares the performance of various machine learning classifiers—such as decision trees, support vector machines, and k-nearest neighbors—in classifying a range of physical activities like walking, sitting, jogging, and climbing stairs.

The study pays special attention to the constraints of mobile platforms, such as battery consumption, memory limitations, and processing speed. It highlights the importance of designing lightweight models that are suitable for on-device processing without offloading data to the cloud, which can raise latency and privacy concerns. Moreover, it addresses context-awareness, where HAR systems adapt based on time, location, or user behavior, thus enhancing prediction quality.

Overall, the paper offers a valuable synthesis of smartphone-centric HAR techniques and proposes best practices for optimizing activity recognition systems for mobile deployment. It concludes by encouraging further research in real-time recognition, personalized modeling, and energy-efficient algorithm design for widespread application in health monitoring, fitness tracking, and lifestyle management.

Title: A Survey on Human Activity Recognition Using Wearable Sensors
Authors: Nweke, H.F., Teh, Y.W., Al-Garadi, M.A.
Year: 2018

This comprehensive survey paper provides a detailed examination of Human Activity Recognition (HAR) systems using wearable sensors, with a particular focus on real-world deployment challenges and opportunities. The authors begin by categorizing the different types of sensors used in HAR, such as accelerometers, gyroscopes, magnetometers, pressure sensors, and biometric sensors. These sensors are commonly embedded in wearable devices like smartwatches, fitness trackers, chest bands, and even smart garments. Their combination allows HAR systems to monitor a variety of physical, physiological, and contextual parameters relevant to user activity.

The paper systematically discusses each stage of the HAR pipeline—starting from data acquisition and preprocessing, where raw sensor signals are cleaned and segmented, to feature extraction, feature selection, and finally activity classification using traditional machine learning models (e.g., decision trees, SVM, k-NN) and more recent deep learning architectures. A notable emphasis is placed on sensor fusion, where data from multiple sensors are integrated to improve recognition accuracy, reduce ambiguity between similar actions, and enhance system robustness in dynamic conditions.

In addition to reviewing computational techniques, the paper addresses practical challenges in wearable HAR systems. These include variability in sensor placement, user behavior differences, battery limitations, data privacy concerns, and inter-device communication issues. The authors highlight that wearable HAR systems must account for heterogeneous sensor configurations, which complicate model generalization and cross-user performance.

Another significant contribution of this survey is its exploration of context-awareness and personalization, both of which are critical in healthcare and fitness applications. By adapting to individual movement patterns, sensor noise characteristics, and environmental contexts, HAR systems can achieve better performance and user satisfaction.

The authors conclude by suggesting directions for future research, including the development of lightweight deep learning models, energy-efficient algorithms, secure data protocols, and unsupervised learning techniques to overcome real-world deployment challenges. The survey serves as a valuable reference for researchers and developers seeking to build practical, user-friendly, and scalable HAR solutions using wearable technologies.

Title: A Survey on Vision-Based Human Action Recognition
Authors: Aggarwal, J.K., Ryoo, M.S.
Year: 2011

This foundational survey provides a thorough overview of vision-based Human Action Recognition (HAR) systems, focusing on the use of video data to automatically identify and understand human actions in a variety of contexts. The paper categorizes existing research into two main approaches: single-layer approaches, which recognize actions as a whole, and hierarchical approaches, which decompose complex activities into simpler, sequential or parallel sub-actions.

The authors explore different representation techniques used in video-based HAR, such as space-time volumes, optical flow, trajectories, and pose-based models. They discuss how actions can be modeled from both global perspectives (e.g., silhouette tracking and full-body motion) and local feature-based methods (e.g., spatiotemporal interest points). These techniques are compared in terms of their robustness, scalability, and suitability for different application environments.

A key contribution of the paper is the discussion of action recognition in challenging visual settings, including dynamic backgrounds, occlusion, viewpoint variation, camera motion, and illumination changes. The paper emphasizes the need for view-invariant models, which can recognize an action regardless of the observer's perspective—a critical capability for surveillance and robotic vision.

The survey also addresses temporal modelling of human actions, highlighting methods such as Hidden Markov Models (HMM), Conditional Random Fields (CRF), and Dynamic Time Warping (DTW) for understanding sequential actions over time. These models help distinguish between actions with similar motion patterns but different temporal structures (

Aggarwal and Ryoo further examine real-time recognition requirements, which are especially important in video surveillance, human-computer interaction (HCI), and assistive technologies. They highlight the trade-offs between model complexity and computational efficiency, as well as the need for online learning in dynamic environments.

In conclusion, the survey identifies open challenges and future directions, including the integration of depth sensors, multi-camera systems, and semantic reasoning to enhance recognition performance. It remains a widely cited reference for researchers working on HAR in video-based contexts, offering both a solid theoretical foundation and practical insights into the design of robust action recognition systems.

Title: A Review on Human Activity Recognition in Smart Homes
Authors: Alemdar, H., Ersoy, C.
Year: 2010

This review paper provides an insightful overview of Human Activity Recognition (HAR) in the context of smart home environments, where embedded and ambient sensors are used to monitor and interpret human behaviors in residential settings. The paper highlights the growing importance of HAR systems in assisted living, elderly care, security, and energy-efficient automation, driven by demographic trends and the need for remote, non-intrusive health monitoring.

The authors classify HAR approaches in smart homes based on the types of sensors used, including motion detectors, pressure sensors, door sensors, RFID, infrared sensors, and wearables. They emphasize the advantage of ambient sensors for passive data collection, which does not require user interaction or wearable devices—thus enhancing user comfort and privacy.

A significant portion of the review discusses the data processing pipeline, starting from sensor signal acquisition to preprocessing, feature engineering, and activity modeling. Both supervised and unsupervised learning algorithms are examined, including decision trees, Naïve Bayes, support vector machines (SVM), and clustering methods. The study notes that supervised learning provides higher accuracy but requires labeled datasets, which can be difficult to collect in real homes.

The paper also addresses challenges in smart home HAR, such as activity overlap (when multiple activities occur simultaneously), user variability, limited sensor coverage, and data sparsity. It highlights the need for multi-modal data fusion, combining input from various sensors to resolve ambiguities and improve recognition performance.

Additionally, the review explores the importance of context-awareness in smart homes—understanding not just the action but the conditions under which it occurs (e.g., time of day, location within the home). Such context can significantly enhance system adaptability and decision-making, especially in personalized healthcare or behavior prediction scenarios.

In conclusion, Alemdar and Ersoy call for future research that balances accuracy, privacy, and scalability. They advocate for the integration of hybrid sensor networks, cloud-based analytics, and adaptive machine learning models that can learn and evolve with the user. This paper remains a foundational reference for those developing HAR systems aimed at smart, safe, and sustainable living environments.

CHAPTER 3

SYSTEM DESIGN

3.1 System Architecture:

The architecture of the Human Activity Recognition (HAR) system is composed of multiple layers working together to capture, process, and classify human movement data in real-time or from recorded sessions. The architecture can be visualized as a pipeline of functional modules:

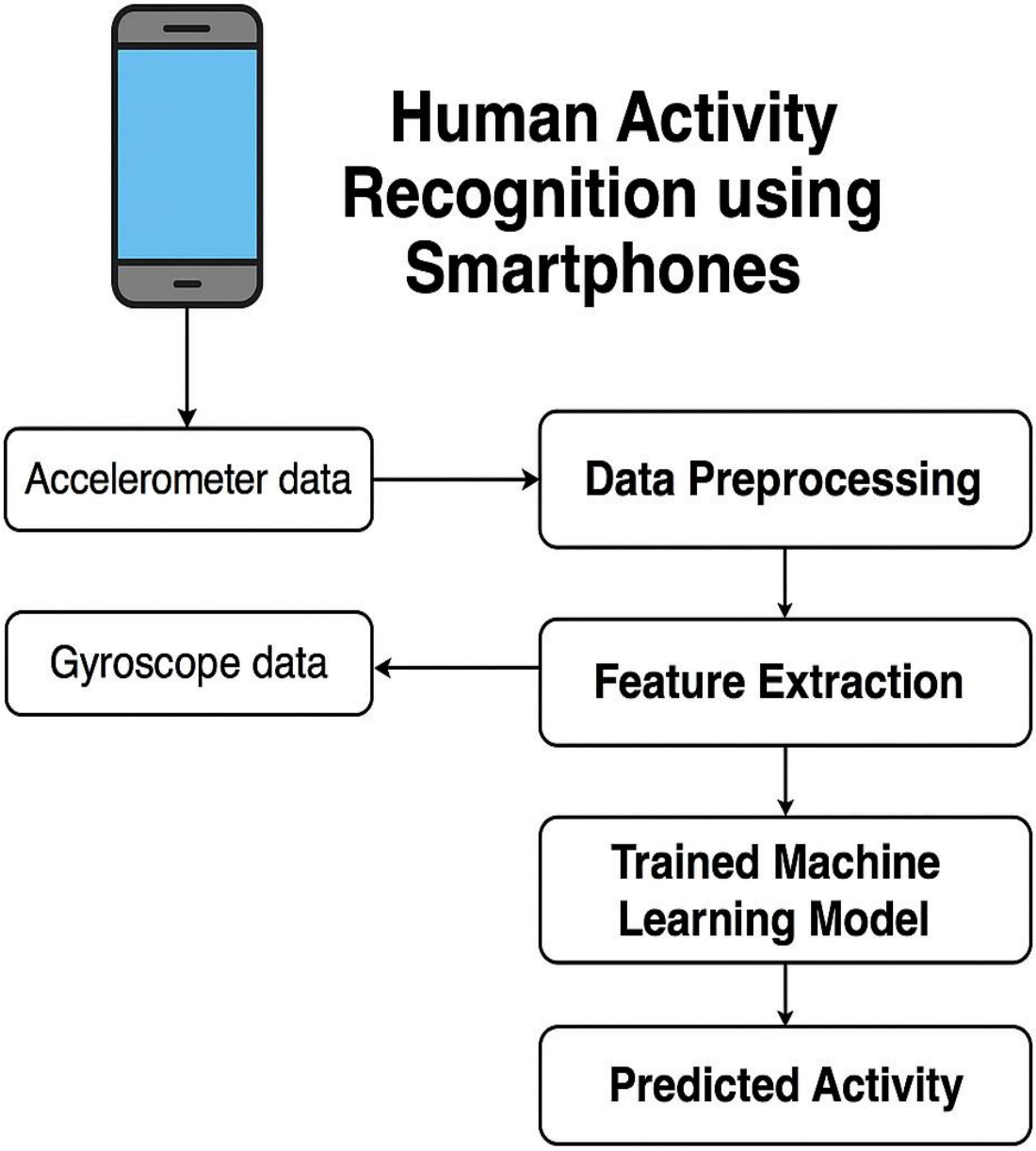

Figure 3.1. Architecture Diagram

3.1.1. Collection Layer (Sensor Input)

The Collection Layer forms the foundation of the HAR system. It is responsible for acquiring raw motion data using the sensors built into modern smartphones. These typically include the accelerometer, which measures acceleration along the x, y, and z axes; the gyroscope, which captures angular velocity and device orientation; and optionally, the magnetometer, which senses the Earth's magnetic field to support orientation tracking. The data collected is continuous and timestamped, reflecting the user's movement patterns over time. This layer is essential as it provides the real-world input that the rest of the system will analyze and classify.

3.1.2. Data Preprocessing Layer

Once the raw data is collected, it moves into the Data Preprocessing Layer, which prepares it for meaningful analysis. This step includes noise removal to eliminate irrelevant or erroneous sensor readings caused by random fluctuations or environmental interference. The clean data is then segmented into fixed-duration time windows (commonly 2 to 5 seconds) to preserve the temporal structure of activities. Each window acts as a sample of human motion. To ensure uniformity, normalization is applied so that the sensor values remain within a consistent range across different devices and users. This step is vital for minimizing bias and improving the generalizability of the machine learning model.

3.1.3. Feature Extraction Layer

The Feature Extraction Layer is a key stage in a Human Activity Recognition (HAR) system. It takes short segments of raw sensor data and converts them into meaningful numerical summaries. Common time-domain features extracted include mean, standard deviation, root mean square (RMS), and range. These help describe how the sensor readings change over time and capture patterns in activities like walking, sitting, or standing.

In addition to time-domain features, frequency-domain features such as Fourier Transform coefficients can also be used. These are especially helpful for recognizing repetitive or rhythmic activities like jogging or climbing stairs. By converting raw data into a compact, structured form, feature extraction enables machine learning models to classify human activities more accurately and efficiently, even on smartphones with limited resources.

3.1.4. Machine Learning Layer

The Machine Learning Layer is the analytical core of the system. In this stage, a set of machine learning models are implemented and tested for their ability to accurately classify human activities based on the extracted features. Models such as Logistic Regression, K-Nearest Neighbors (KNN), Decision Tree, Support Vector Machine (SVM), and Linear Discriminant Analysis (LDA) are selected due to their suitability for structured classification problems. These models are first trained on labelled datasets where each data segment is associated with a known activity (e.g., walking, sitting, etc.). After training, the models are used to predict the activity of new, unseen data segments. To measure model effectiveness, evaluation metrics such as accuracy, confusion matrix, Receiver Operating Characteristic (ROC) curves, and Area Under the Curve (AUC) are used.

3.1.5. Output Layer (Activity Recognition Result)

After prediction, the results are passed to the Output Layer, which translates the recognized activity into a usable format. The output is typically a label, such as "Walking", "Sitting", or "Lying Down", which reflects the most likely activity based on the sensor input. This label can then be displayed to the user through a mobile app interface or dashboard. In addition to visualization, this layer can log activity history, trigger alerts, or interface with third-party systems such as healthcare monitoring platforms or fitness trackers to support personalized services or interventions.

3.1.6. Optional Layers (Extension or Deployment)

To enhance system capabilities, several **optional layers** can be included:

- The Real-Time Application Layer allows integration of the HAR system into a mobile app that performs live monitoring and classification of human activity. This enables immediate feedback or alerting when specific movements are detected (e.g., falls or prolonged inactivity).
- The Cloud Storage Layer supports uploading recognized activity data to a remote server or cloud platform. This is useful for long-term tracking, health reporting, or integration with machine learning services for further improvement.
- The User Interface (UI) Layer provides a visual overview of recognized activities, system performance graphs, and potentially, user settings. This enhances usability by making the system interactive and user-friendly, especially for healthcare providers, fitness coaches, or end users monitoring their own activity patterns.

3.2 Functional Requirements

The functional requirements define the essential operations and capabilities that the Human Activity Recognition (HAR) system must support in order to fulfill its intended purpose. These requirements are organized based on the key stages of the HAR pipeline, ensuring a clear and systematic flow from data collection to prediction, evaluation, and user interaction.

The first major requirement is Data Acquisition. The HAR system must be able to collect raw motion data from the smartphone's internal sensors, primarily the accelerometer and gyroscope. These sensors provide three-axis measurements of acceleration and angular velocity, which are critical for recognizing human activities. The system should support both real-time data collection through sensor APIs and offline processing by importing pre-recorded sensor data in formats such as CSV. Additionally, each data sample must be accompanied by metadata including timestamps, user or subject identifiers, and activity labels, which are particularly important when the system is trained using supervised learning techniques.

Once the data is collected, it must go through a thorough Data Preprocessing stage. In this phase, the system is required to clean and standardize the data to ensure consistency and remove any distortions. This includes handling missing or incomplete values, filtering out noise, and removing irrelevant segments that could negatively impact model performance. Feature scaling techniques such as normalization or standardization must also be applied to bring all sensor values into a comparable range. The system should also support feature extraction, deriving key statistical and mathematical indicators such as mean, standard deviation, Signal Magnitude Area (SMA), and frequency-domain features using Fast Fourier Transform (FFT). These features form the input to machine learning models.

The next critical stage is Activity Classification. The system must utilize supervised machine learning models—including Logistic Regression, Decision Trees, K-Nearest Neighbors, Support Vector Machines (SVM), and Linear Discriminant Analysis (LDA)—to classify human physical activities. These activities may include walking, sitting, standing, lying down, walking upstairs, and walking downstairs. The system must support multi-class classification, meaning it can handle several activity types within a single predictive model. Flexibility in choosing or switching models should be available for comparative evaluation and performance optimization.

To ensure accuracy and reliability, the HAR system must incorporate a Model Evaluation component. The models must be assessed using well-established performance metrics such as

accuracy, confusion matrix, precision, recall, F1-score, and ROC-AUC (Receiver Operating Characteristic - Area Under the Curve). These metrics provide a comprehensive view of how well the model is performing across different activity categories. In addition to numeric scores, the system must generate visualizations, such as graphs and charts, to support interpretation and facilitate comparison between different models or configurations.

Another important requirement is Model Persistence. Once a model has been trained and validated, it must be saved in a serialized format using tools like joblib or pickle. This enables the system to reload the model during future sessions for inference purposes, thus avoiding the need for retraining every time the system is used. This is especially important for real-time applications and mobile or embedded deployments, where computational resources are limited.

In systems that support direct user interaction, a User Interface (UI) or API layer is also necessary. A basic web-based or command-line interface should allow users to upload new sensor data, select or configure models, and view activity predictions. Additionally, an optional Flask-based REST API can be developed to provide access to the system's prediction services and model status remotely. This would support integration into other platforms or real-time systems that require continuous activity monitoring.

Finally, the HAR system must include a Result Reporting module. This component is responsible for generating summary reports of activity predictions and logs of the data processing and classification steps. Visual representations such as line graphs, bar charts, and heatmaps should be generated to aid understanding and performance review. These reports ensure transparency, traceability, and ease of interpretation for users, developers, or researchers analyzing system outputs.

3.3 Non-Functional Requirements

3.3.1. Performance Requirements

The system must operate with high responsiveness to support real-time or near-real-time applications. It should be capable of collecting and processing sensor data with minimal latency, ensuring smooth data acquisition and rapid predictions. After data is preprocessed and passed to the trained machine learning model, the prediction time per window (typically 2–5 seconds of sensor data) must not exceed 1 second, making the system suitable for live feedback or activity-triggered responses.

3.3.2. Scalability

The architecture must be designed to support scalability across several dimensions. First, it should accommodate multiple users simultaneously, whether for centralized data analysis or distributed mobile applications. Second, the system should allow expansion to additional activities, such as jogging, cycling, or stair climbing, without requiring structural changes. Finally, it must support higher sampling rates from sensors for use cases that require finer motion detection, while maintaining system stability and performance.

3.3.3. Reliability

The HAR system must ensure reliable operation during data acquisition and model execution. This includes robust handling of temporary connection loss, minor sensor glitches, or momentary pauses in data flow. Under such conditions, the system must preserve all collected data unless explicitly stopped by the user. Data loss or corruption during recording, especially during supervised labeling, must be prevented to maintain dataset integrity and model accuracy.

3.3.4. Usability

Usability is a key factor for successful adoption of the system. If a user interface (UI) is implemented, it must be intuitive and user-friendly, allowing users to easily start or stop data collection, visualize live sensor streams, and view activity predictions. The system should minimize the number of steps required by the user, supporting low-effort data capture even for non-technical users such as patients, fitness enthusiasts, or elderly individuals.

3.3.5. Maintainability

The codebase and supporting scripts must follow a modular and maintainable structure, making it easy to update machine learning models, data preprocessing pipelines, or file formats as needed. Proper documentation, separation of concerns (e.g., separate modules for data handling, modeling, and visualization), and use of configuration files will help future developers or researchers reuse the system for different HAR applications or integrate it into larger projects with minimal overhead.

3.3.6. Portability

The HAR system should be designed for platform independence. While initial development may be focused on Android smartphones due to wider sensor access, the architecture should be flexible enough for adaptation to other environments, including iOS, cross-platform mobile frameworks, or even embedded systems (e.g., Raspberry Pi, wearables). This allows for a wider reach and ease of deployment in diverse settings.

3.3.7. Security and Privacy

Given the sensitive nature of user activity data, the system must be developed with privacy and security as a priority.. Activity logs and user identifiers must be stored securely, and encryption should be applied during data transmission and storage whenever feasible. All privacy practices must align with current data protection regulations such as GDPR or local compliance standards.

3.3.8. Accuracy and Robustness

To be useful in real-world applications, the HAR system must maintain high classification accuracy, targeting 90% or higher on well-balanced datasets. Furthermore, the trained model must demonstrate robustness across users, meaning it should generalize well despite differences in user behavior, sensor orientation, or device placement. This ensures consistent performance for a wide population and usage conditions.

3.3.9. Interoperability

The system should support standard data formats for both input and output, such as CSV, JSON, or ARFF, enabling compatibility with external tools and platforms. It should also integrate seamlessly with popular machine learning frameworks including scikit-learn, TensorFlow, Kera's, and Weka, allowing developers or researchers to export data for further experimentation or deployment into external systems.

3.3.10. Logging and Debugging

For traceability and troubleshooting, the system must include a comprehensive logging mechanism. This includes logging of session start and end times, any runtime errors or sensor issues, and summary statistics such as the number of samples collected per activity type. Logs are essential for both development debugging and auditing performance during real-world deployment or experimentation.

3.4 Technical Requirements

Technical Requirements refer to the tools, environments, and system components necessary to develop, deploy, and run the HAR system effectively.

Table I: Hardware Requirements

Component	Requirement Description
Smartphone	Must include an accelerometer and gyroscope sensors
RAM	Minimum 2 GB RAM for smooth sensor data capture
Processor	Dual-core or better (recommended: ARM Cortex-A or higher)
Storage	At least 500 MB free space for storing data
Battery	Should support continuous data logging for 1–2 hours minimum
Optional PC/Server	For model training & testing using Python or Jupyter Notebook

Table II: Software Requirements

Software	Version / Notes
Python	Version 3.7 or higher
Pandas	Data handling
NumPy	Numerical operations
scikit-learn	Machine learning algorithms
Matplotlib / Seaborn	Visualization and plotting
Jupyter Notebook	For experimentation and reporting
Android App (optional)	For collecting live data from smartphones

3.4.3. Data Requirements

- **Input data**: Raw sensor data from accelerometer and gyroscope
    - Sampling rate: e.g., 50Hz
    - Format: CSV, TXT, or real-time stream via Bluetooth or file logging
- **Labels**: Activity type (e.g., Walking, Sitting)

Table III: Model Requirements

Feature	Description
Algorithm	Logistic Regression, Decision Tree, LDA, SVM, etc.
Multi-Class Support	Yes (e.g., One-vs-Rest strategy for ROC)
Evaluation Metrics	Accuracy, Confusion Matrix, ROC-AUC

Table IV: Interwork Requirements (If Real-Time Uploading is Used)

Component	Description
Internet	Required if data is streamed/uploaded to cloud or remote server
Bluetooth/WIFI	Optional for transferring data from phone to PC

3.4.6. Storage Requirements

- Each session can produce megabytes of data depending on duration and sensors.
- Prefer structured folder system for saving:
    - Sensor logs
    - Activity labels
    - Model results and visualizations

3.5 Machine Learning Models

This project adopts a comparative modeling approach by implementing a range of supervised learning algorithms for binary classification — specifically predicting whether a candidate should be shortlisted or not. Each model brings different strengths in terms of interpretability, performance, and handling of feature relationships.

3.5.1. Logistic Regression

A statistical model for binary classification is called logistic regression. Using the sigmoid function, it forecasts the likelihood that an instance will belong to a particular class, guaranteeing results between 0 and 1. To minimize the log loss, the model computes a linear combination of input characteristics, transforms it using the sigmoid, and then optimizes its coefficients using methods like gradient descent. These coefficients establish the decision boundary that divides the classes. Because of its ease of use, interpretability, and versatility across multiple domains, Logistic Regression is widely used in machine learning for problems that involve binary outcomes. Overfitting can be avoided by implementing regularization.

Logistic Regression models the likelihood that an instance will belong to a particular class. It uses a linear equation to combine the input information and the sigmoid function to restrict predictions between 0 and 1. Gradient descent and other techniques are used to optimize the model's coefficients to minimize the log loss. These coefficients produce the resulting decision boundary, which divides instances into two classes. When it comes to binary classification, logistic regression is the best choice because it is easy to understand, straightforward, and useful in a variety of settings. Generalization can be improved by using regularization.

3.5.2. Decision Tree Classifier

The decision tree algorithm is a powerful and interpretable tool used in machine learning for both classification and regression tasks. Its simplicity and transparency make it a popular choice for various applications, including predictive modelling and data analysis. The decision tree operates by recursively partitioning the feature space into smaller regions based on the values of input features, ultimately leading to a tree-like structure where each internal node represents a decision based on a feature, and each leaf node represents a class label or a numerical value.

One of the key advantages of decision trees is their ability to handle both numerical and categorical data without the need for extensive preprocessing. Additionally, decision trees are robust to outliers and can capture complex relationships between features and the target variable. Moreover, decision trees are inherently interpretable, allowing users to understand the decision- making process and gain insights into the factors influencing the model. Despite their limitations, decision trees remain a valuable tool in the machine learning toolbox due to their simplicity, interpretability, and ability to handle a wide range of data types decision trees can yield accurate and understandable models for various real-world applications.

3.5.3. Random Forest Classifier

Random Forest is a popular and powerful ensemble learning algorithm used for both classification and regression tasks in machine learning. It works by constructing a large number of decision trees during training and aggregating their outputs—either by majority vote for classification or averaging for regression—to produce a final prediction. The key idea behind Random Forest is to reduce overfitting and improve model accuracy by introducing randomness: each tree is trained on a random subset of the data (using bootstrapping), and at each split in a tree, only a random subset of features is considered. This diversity among the trees makes the overall model more robust and less sensitive to noise in the data. Random Forest handles both numerical and categorical data, deals well with missing values, and can capture complex, non-linear relationships. It also provides useful insights through feature importance scores, helping to identify which variables contribute most to the prediction. Due to its high accuracy, scalability, and ability to handle high-dimensional datasets, Random Forest is widely used in real-world applications such as fraud detection, medical diagnosis, and customer segmentation.

In addition to its strong predictive performance, Random Forest offers several advantages that make it a versatile choice in machine learning. One of its key strengths is its ability to handle imbalanced datasets more effectively by using techniques such as class weighting or balanced subsampling. Random Forest is also resistant to overfitting, especially when a large number of trees are used, because the averaging of multiple uncorrelated trees helps smooth out individual biases or noise. It supports out-of-bag (OOB) evaluation, a built-in cross-validation technique that uses unused samples from the bootstrap process to estimate the model's performance without needing a separate validation set. Random Forest also works well with highdimensional data, such as text or gene expression datasets, because it can identify relevant features even when many are irrelevant. Moreover, it’s relatively easy to use and requires minimal hyperparameter tuning to achieve good results, although tuning parameters like the number of trees, maximum depth, and minimum samples per leaf can further enhance performance. While it may be less interpretable than simpler models like decision trees or logistic regression, Random Forest remains a go-to algorithm for building reliable and accurate models across a wide range of domains.

3.5.4. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a simple yet powerful supervised machine learning algorithm used for both classification and regression tasks. In the context of classification, KNN works by identifying the ‘k’ data points in the training set that are closest to a given input based on a distance metric, commonly Euclidean distance. The algorithm then assigns the most common class among these neighbors to the input instance. KNN is a lazy learner, meaning it does not build a model during training but rather memorizes the training dataset and performs computation only during prediction. This makes KNN easy to implement but potentially slow on large datasets. The performance of KNN depends heavily on the choice of ‘k’ (the number of neighbors), the distance metric used, and the scale of the data—hence, feature scaling is typically required. Despite its simplicity, KNN can perform very well with well-structured data and is widely used in areas such as pattern recognition, recommendation systems, and image classification.

In addition to its simplicity and intuitive nature, K-Nearest Neighbors (KNN) has several characteristics that influence its effectiveness. One important aspect is that KNN is a nonparametric algorithm, meaning it makes no prior assumptions about the underlying data distribution, making it flexible for various types of datasets. However, this also means KNN is sensitive to irrelevant or redundant features, which can negatively impact its accuracy. Therefore, feature selection or dimensionality reduction techniques, such as PCA (Principal Component Analysis), are often used in conjunction with KNN. The algorithm’s performance is also affected by the value of ‘k’—a small value can make the model sensitive to noise (overfitting), while a large value may oversmooth class boundaries (underfitting). KNN also requires storing the entire training set, making it memory-intensive and computationally expensive during prediction, especially with large datasets. To address this, techniques like KD-Trees or Ball Trees are sometimes employed to speed up nearest-neighbor searches. Despite these challenges, KNN remains a valuable algorithm due to its ease of use, interpretability, and effectiveness when used on small to medium-sized datasets with clearly separable classes.

3.5.5. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a powerful supervised learning algorithm used primarily for classification tasks, although it can also be adapted for regression. At its core, SVM works by identifying the optimal hyperplane that separates data points of different classes with the maximum possible margin. This margin is defined by the closest data points from each class, known as support vectors. By maximizing this margin, SVM aims to improve the model's ability to generalize to unseen data.

A major strength of SVM is its ability to handle both linearly and non-linearly separable data. For non-linear cases, SVM uses kernel functions to map the input data into higher-dimensional spaces where a linear separation becomes possible. This approach, known as the kernel trick, avoids the computational cost of explicitly performing the transformation. Popular kernels include the linear kernel for simple datasets, the polynomial kernel for capturing more complex relationships, and the radial basis function (RBF) kernel for non-linear, high-dimensional data. SVM also includes a regularization parameter, denoted as C, which controls the trade-off between achieving a low error on the training data and maintaining a wide margin. A smaller C value encourages the model to prioritize a wider margin, allowing some misclassifications and enhancing generalization. A larger C value aims for fewer misclassifications but risks overfitting to the training data.

When configured with probability=True, SVM can also provide probabilistic outputs, making it suitable for applications that require not just classification decisions but confidence scores as well. These probabilities are computed using methods like Platt scaling, which fits a sigmoid curve to the model's outputs. This feature is especially useful in areas such as risk prediction, recommendation systems, and ensemble learning, where understanding the certainty of predictions is important.

Although SVM can be computationally intensive for large datasets—especially when using complex kernels—it remains one of the most accurate and robust classification algorithms. It is widely applied in domains like image and speech recognition, text classification, bioinformatics, and fraud detection. Its balance of precision, generalization, and versatility makes SVM a reliable choice for many real-world machine learning tasks.

3.5.6. XGBoost (XGBClassifier)

XGBoost Classifier (Extreme Gradient Boosting) is a highly efficient and scalable implementation of the gradient boosting algorithm, designed to deliver superior performance on structured or tabular datasets. It builds an ensemble of decision trees in a sequential manner, where each new tree is trained to correct the errors made by the combined ensemble of the previous trees. What sets XGBoost apart is its use of second-order derivatives (i.e., gradient and Hessian) to optimize the loss function, leading to faster convergence and more accurate models. Additionally, it incorporates regularization techniques (L1 and L2) to control model complexity and prevent overfitting, which enhances its generalization capabilities. XGBoost supports features like tree pruning, early stopping, parallel processing, and handling of missing values internally, making it both powerful and user-friendly. It also includes hyperparameters that allow fine control over model behavior, such as learning rate, maximum depth of trees, subsampling ratios, and column sampling. Due to its high accuracy, speed, and versatility, XGBoost Classifier is widely used in machine learning competitions and real-world applications like credit scoring, fraud detection, and predictive maintenance. In addition to its core strengths, the XGBoost Classifier offers several advanced capabilities that make it a preferred choice for many machine learning practitioners. One of its key advantages is its ability to handle sparse data efficiently, thanks to a sparsity-aware algorithm that automatically learns the best direction to take when encountering missing values. XGBoost also uses a histogram-based algorithm for finding optimal splits in the decision trees, which improves both speed and memory efficiency—particularly important for large datasets. The model supports cross-validation directly within the training process, allowing users to monitor evaluation metrics across multiple folds and implement early stopping based on validation performance. Another powerful feature is its built-in ability to provide feature importance scores, which helps users interpret the model and understand the relative impact of each feature. Moreover, XGBoost integrates well with popular machine learning frameworks such as scikit learn, TensorFlow, and Spark, making it suitable for both local experimentation and large-scale production environments. Its extensive set of hyperparameters allows for fine-tuning to optimize both accuracy and training time. These combined features make XGBoost not only one of the most accurate gradient boosting methods available, but also one of the most adaptable and production-ready tools in the machine learning ecosystem.

3.6 Python Libraries Used

Python

Python is a high-level, interpreted, and general-purpose programming language known for its simplicity, readability, and versatility. Designed with an emphasis on code clarity and minimalism, Python uses an easy-to-understand syntax that allows developers to write fewer lines of code compared to other programming languages. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python comes with a vast standard library and a rich ecosystem of third-party packages, making it suitable for a wide range of applications such as web development, data science, machine learning, automation, scripting, software development, and more. Its interactive nature and extensive community support have made Python a preferred choice for both beginners and experienced developers. Tools like Jupyter Notebook, frameworks like Django and Flask, and libraries like NumPy, pandas, and scikit-learn further enhance Python's capabilities, especially in fields like scientific computing and artificial intelligence. Overall, Python's simplicity, flexibility, and robust ecosystem make it one of the most popular and powerful programming languages in the world today.

```
import warnings
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px

from  sklearn.decomposition import  PCA
from sklearn import preprocessing
from sklearn import tree
from sklearn.manifold import TSNE
from xgboost import XGBClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm # Changed from svm33 to svm to import the svm module
from sklearn import metrics
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
from sklearn.model_selection import train_test_split
```

Figure 3.6: Imported Python Libraries

3.6.1. pandas

Pandas is a powerful and widely used open-source data analysis and manipulation library for Python, built on top of NumPy. It provides two primary data structures: Series, which represents a one-dimensional labeled array, and DataFrame, a two-dimensional labeled table that can hold data of different types (e.g., integers, floats, strings). These structures make it easy to load, clean, manipulate, analyze, and visualize data. Pandas excels at handling tabular data, such as data from spreadsheets, CSV files, SQL databases, or JSON, and provides intuitive functions for filtering, sorting, grouping, merging, reshaping, and aggregating data. With its ability to handle missing data, perform time series analysis, and efficiently manipulate large datasets, pandas has become a fundamental tool in data science and machine learning workflows. It integrates well with other Python libraries like NumPy, Matplotlib, and scikit learn, making it essential for exploratory data analysis and preprocessing tasks. In addition to its core functionalities, pandas offers a rich set of features that make it especially suitable for real-world data analysis. One of its key strengths is its ability to handle missing or null values gracefully, allowing users to detect, fill, or drop incomplete data with ease. It also supports powerful time series functionality, including date-range generation, frequency conversion, moving window statistics, and time shifting—making it ideal for financial or temporal data analysis. Pandas allows for efficient reading and writing of data from various file formats such as CSV, Excel, JSON, Parquet, and SQL databases, making it highly versatile in data ingestion and export tasks. With features like multi-indexing, users can work with high dimensional data in a structured way, and its groupby () operations enable advanced data aggregation and transformation. Furthermore, pandas optimizes performance through vectorized operations and integration with NumPy, while libraries like Dask and Modin extend its capabilities to handle larger-than-memory datasets or distributed computing. Due to its extensive documentation and active community, pandas remains one of the most user-friendly and robust tools in the Python ecosystem for data manipulation and analysis.

3.6.2. numpy

NumPy (Numerical Python) is a fundamental library for numerical computing in Python, widely used in data science, machine learning, and scientific computing. It provides powerful data structures, the most important being the N-dimensional array (or ndarray), which allows efficient storage and manipulation of large datasets. NumPy supports a wide range of mathematical operations such as linear algebra, statistics, Fourier transforms, and random number generation, all of which are optimized for performance using underlying C and Fortran code. Compared to native Python lists, NumPy arrays consume less memory and provide significantly faster computations due to vectorization, which eliminates the need for explicit loops. Additionally, NumPy integrates well with other Python libraries like pandas, matplotlib, and scikit-learn, forming the backbone of the scientific Python ecosystem. Its broadcasting feature enables operations between arrays of different shapes without requiring explicit looping or reshaping, making code both simpler and faster. Overall, NumPy is essential for anyone working with numerical data in Python, offering both speed and functionality for a wide range of applications. In addition to its core functionalities, NumPy offers several advanced features that make it indispensable for high-performance scientific computing. One of its key strengths is broadcasting, which allows arithmetic operations on arrays of different shapes by automatically expanding them to compatible dimensions, reducing the need for manual reshaping. NumPy also provides universal functions (ufuncs), which are highly optimized functions that operate element-wise on arrays, offering efficient alternatives to loops. For working with complex data pipelines, NumPy supports structured arrays and memory-mapped files, enabling efficient handling of large datasets that do not fit entirely in memory. It also includes tools for array manipulation, such as stacking, splitting, reshaping, and transposing, which are essential for data preprocessing. Moreover, NumPy serves as the computational foundation for many other libraries like SciPy, TensorFlow, and PyTorch, making it a critical component in the Python data ecosystem. With extensive documentation and a large community, NumPy is not only a performance-oriented library but also a user-friendly tool for both beginners and advanced users working with numerical data.

3.6.3. matplotlib

Matplotlib is a widely used data visualization library in Python that provides a comprehensive set of tools for creating static, animated, and interactive plots. It is particularly well-suited for creating high-quality, publication-ready visualizations and is the foundational plotting library upon which other libraries like Seaborn and pandas plotting are built. The most commonly used module in Matplotlib is pyplot, which offers a simple interface similar to matlab for generating common plot types such as line charts, bar graphs, histograms, scatter plots, and pie charts. Matplotlib gives users fine-grained control over every aspect of a plot, including figure size, axes, colors, labels, fonts, and legends, making it highly customizable for complex visualization needs. It integrates seamlessly with NumPy and pandas, enabling easy plotting of numerical and tabular data. Additionally, Matplotlib supports multiple output formats, such as PNG, SVG, PDF, and interactive plots in Jupyter Notebooks, making it suitable for use in both exploratory data analysis and final reporting. While it has a steeper learning curve compared to high-level libraries, Matplotlib's versatility and extensive functionality make it a core tool in any data scientist's or analyst’s toolkit.

3.6.4. Seaborn

Seaborn is a powerful Python data visualization library built on top of Matplotlib, designed to make it easier to create attractive and informative statistical graphics. It provides a high-level interface for drawing a wide range of plots, including bar plots, box plots, violin plots, line plots, scatter plots, heatmaps, and pair plots, with minimal code. Seaborn works seamlessly with pandas DataFrames, allowing users to directly pass structured datasets into plotting functions and automatically handle the mapping of variables to visual elements like axes, colors, and styles. One of its key strengths is its ability to display statistical relationships and distributions clearly, often including confidence intervals and regression lines by default. It also supports features like faceting, which enables the creation of multi-plot grids for subgroup comparisons. Seaborn enhances the aesthetics of visualizations through themes and color palettes, resulting in plots that are both professional-looking and easy to interpret. Overall, Seaborn is an essential tool for data exploration and analysis, particularly when working with datasets where understanding variable relationships and distributions is critical.

3.6.5. scikit-learn (sklearn)

Scikit-learn is a widely used open-source machine learning library in Python that provides simple and efficient tools for data mining, data analysis, and modeling. Built on top of foundational libraries like NumPy, SciPy, and matplotlib, scikit-learn offers a unified and consistent interface for implementing a broad range of machine learning algorithms, including classification, regression, clustering, dimensionality reduction, and model selection. It supports popular algorithms such as logistic regression, decision trees, random forests, support vector machines, k-nearest neighbors, and gradient boosting, making it suitable for both beginners and experienced practitioners. Scikit-learn also includes essential tools for preprocessing data (e.g., scaling, encoding, imputation), evaluating model performance using metrics and cross validation, and tuning hyperparameters through grid search or randomized search. Its design emphasizes usability, modularity, and performance, enabling users to build end-to-end machine learning pipelines efficiently. Scikit-learn integrates well with other Python libraries like pandas and NumPy, and it is widely used in academic research, industry applications, and educational settings due to its ease of use, reliability, and comprehensive documentation.

3.6.6. xgboost

XGBoost (Extreme Gradient Boosting) is a powerful and scalable open-source machine learning library based on the gradient boosting framework, designed for speed, accuracy, and efficiency. It builds an ensemble of decision trees in a sequential manner, where each new tree is trained to correct the errors made by the previous ones, using gradient descent to minimize a specific loss function. What makes XGBoost stand out is its use of second-order optimization (utilizing both gradients and Hessians), regularization (L1 and L2) to prevent overfitting, and advanced features like tree pruning, early stopping, and built-in cross-validation. XGBoost is capable of handling missing values and supports parallel and distributed computing, making it suitable for both small and large-scale machine learning tasks. It also provides detailed feature importance scores and integrates seamlessly with popular tools like scikit-learn, making it easy to use in Python-based workflows. Due to its high performance, flexibility, and robustness, XGBoost is widely adopted in machine learning competitions and real-world applications such as fraud detection, risk modeling, and recommendation systems.

3.6.7. flask

Flask is a lightweight and flexible web framework for Python that is commonly used to build web applications and RESTful APIs. It follows a microframework design philosophy, meaning it provides the essential tools to get a web app running—such as routing, request and response handling, and templating—without enforcing a specific project structure or including built-in components like database layers or form validation. Instead, Flask allows developers to add only the extensions they need, offering great freedom and modularity. Built on the Werkzeug WSGI toolkit and Jinja2 templating engine, Flask is easy to learn and well-suited for both beginners and experienced developers. It supports URL routing, session management, JSON handling, and integration with frontend technologies. Flask also works seamlessly with other Python libraries and tools, making it ideal for building APIs, dashboards, and small to medium sized applications. Its simplicity, readability, and large community support make Flask a popular choice for prototyping, academic projects, and even production-grade web services.

3.6.8. Joblib pickle

Joblib is a Python library designed for efficient serialization and parallel computing, commonly used in data science and machine learning workflows. It is especially effective for saving and loading large Python objects such as NumPy arrays, pandas DataFrames, or trained machine learning models. Unlike Python's built-in pickle module, Joblib is optimized for performance and can handle large data more efficiently by using techniques like compression and memory mapping. This makes it ideal for persisting machine learning models that need to be reused without retraining. In addition to serialization, Joblib provides tools for parallel processing through its Parallel and delayed functions, which allow for the easy execution of tasks across multiple CPU cores, speeding up computationally intensive processes such as model training or data preprocessing. It integrates smoothly with libraries like scikit-learn, where it is often used to save and load trained models. Overall, Joblib enhances performance, reproducibility, and scalability in Python-based data science projects.

3.7 UML Diagrams

Unified Modeling Language (UML) diagrams are used to visually represent the structure and behavior of software systems. In the context of the Human Activity Recognition (HAR) using Smartphones and Machine Learning project, UML diagrams help illustrate how components interact, how data flows, and how different classes or modules are structured.

Below are the key UML diagrams relevant to this project:

3.7.1. Use Case Diagram

A Use Case Diagram is a UML diagram used to describe the functional interactions between users (actors) and the system components. In the context of the Human Activity Recognition (HAR) system, it illustrates how various users (or external systems) interact with the system to perform specific tasks.

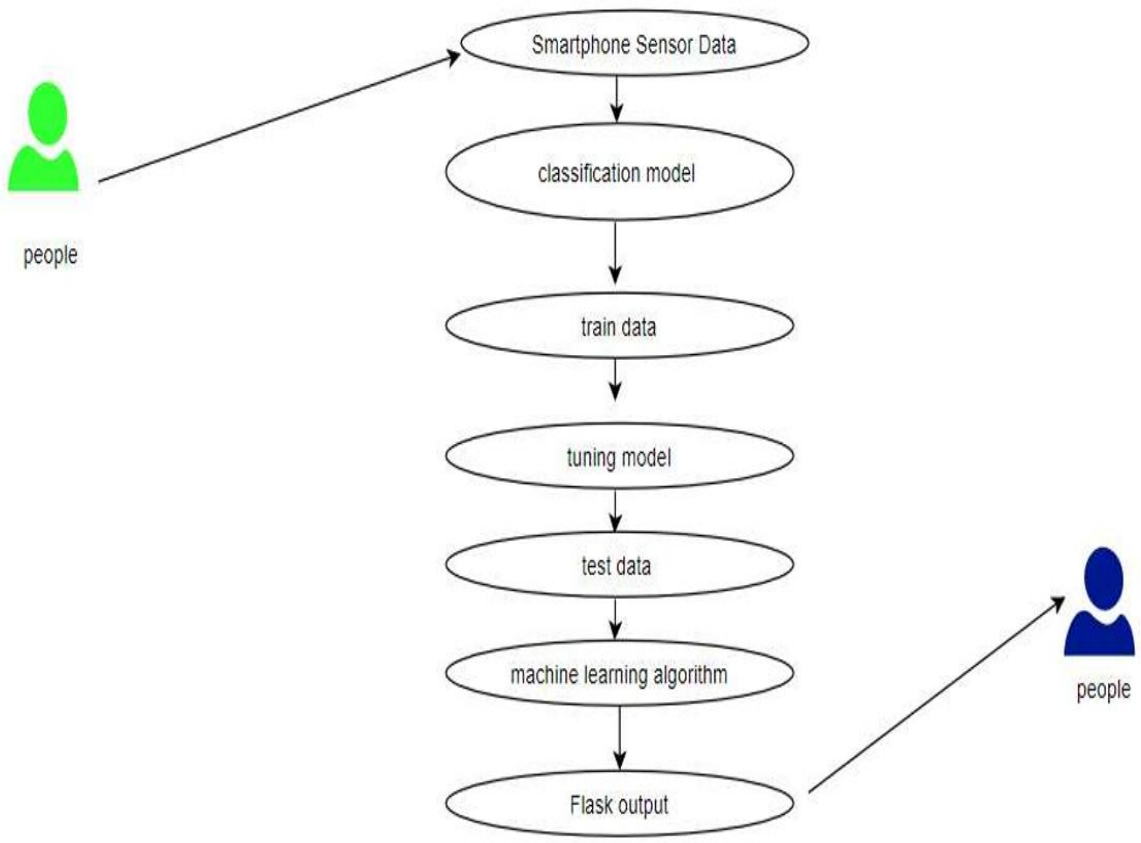

Figure. 3.7.1. Use Case Diagram

3.7.2. Activity Diagram

The Activity Diagram illustrates the dynamic flow of actions within the Human Activity Recognition (HAR) system. It models the behavior of the system from data acquisition to activity prediction and result display. This diagram is particularly useful for visualizing how data flows through different processing stages in response to user or sensor input.

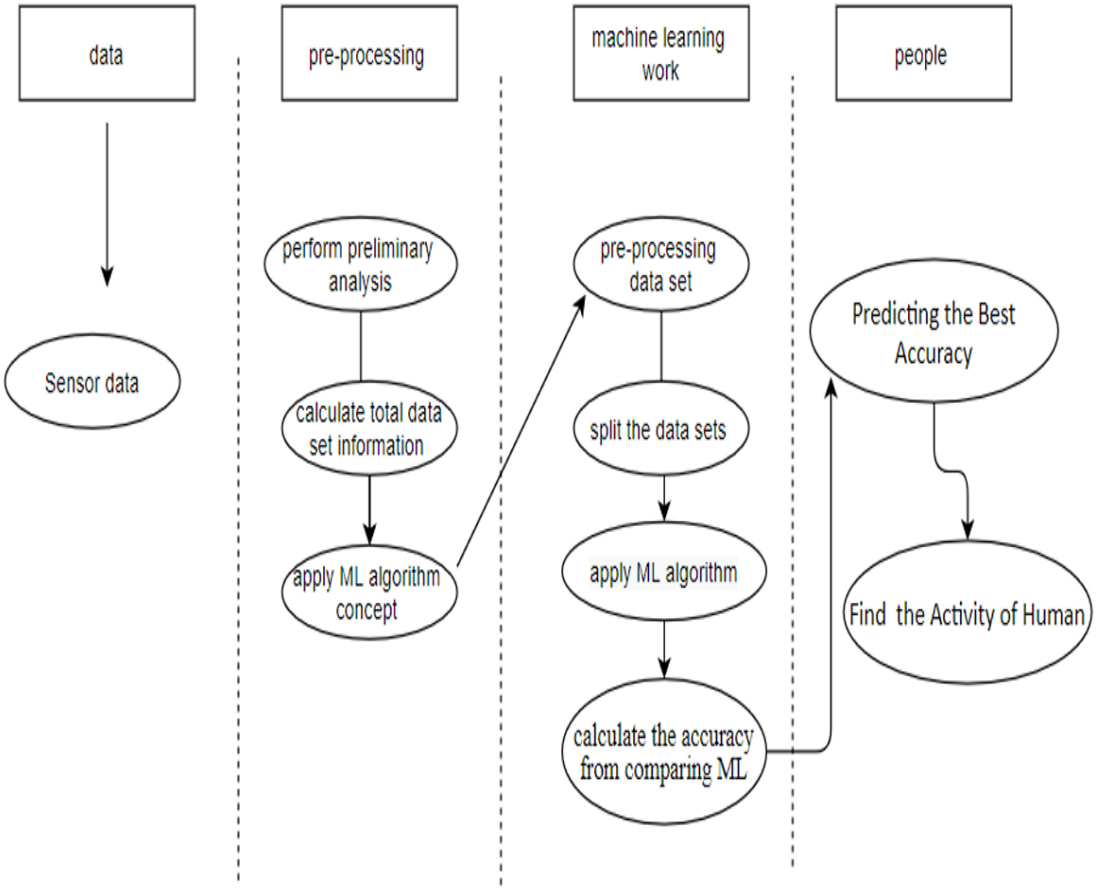

Figure. 3.7.2. Activity Diagram

3.7.3. Class Diagram

The UML Class Diagram provides a structural overview of the Human Activity Recognition (HAR) system. It illustrates how various modules (represented as classes) interact with each other, including their attributes (data) and methods (functions). This structure helps define system organization, encourage modularity, and simplify maintenance or scaling.

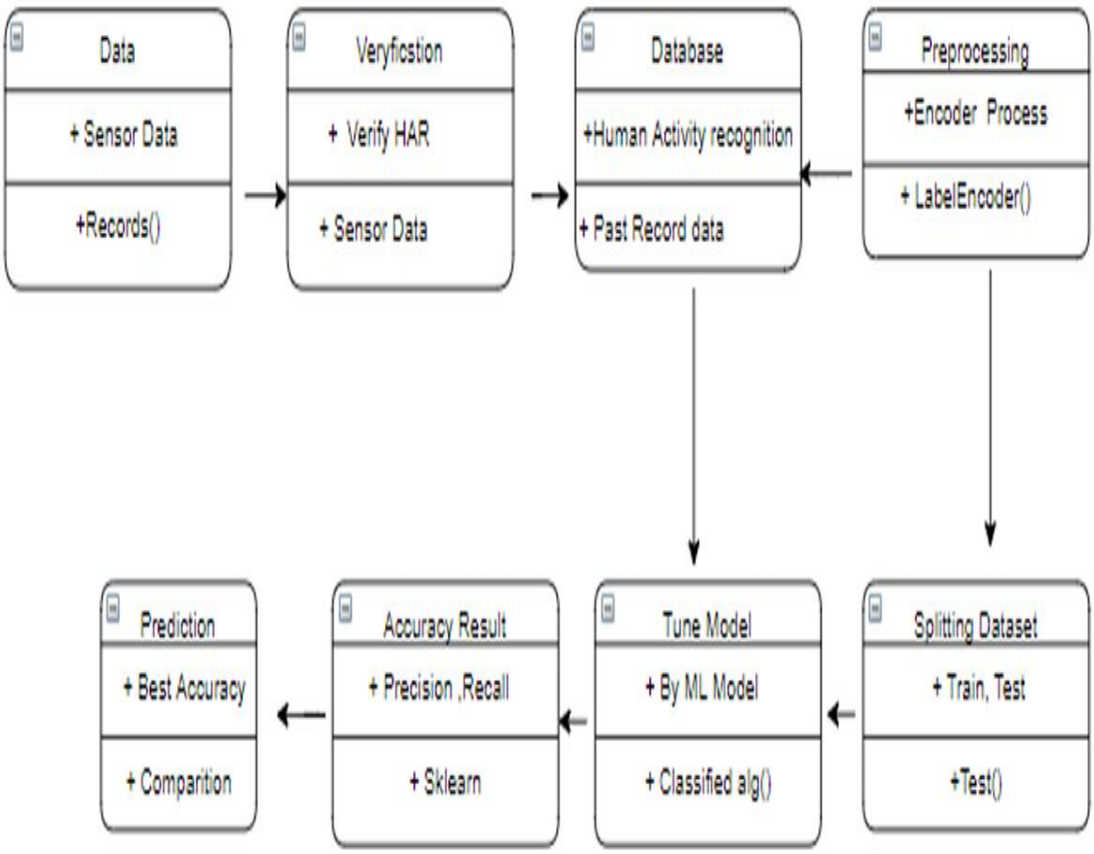

Figure. 3.7.3. Class Diagram

3.7.4. Sequence Diagram

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Other dynamic modeling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview diagramming. Sequence diagrams, along with class diagrams and physical data models are in my opinion the most important design-level models for modern business application development.

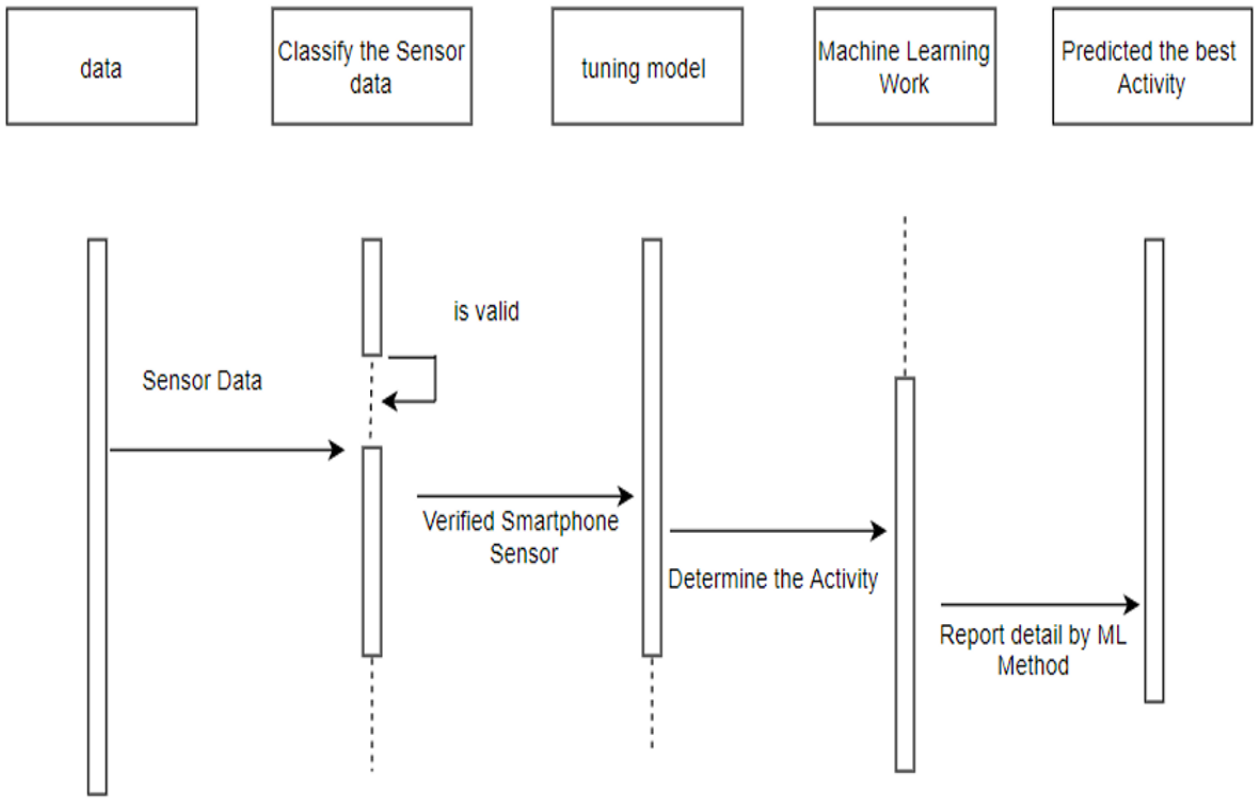

Figure. 3.7.4. Sequence Diagram

CHAPTER 4

METHODOLOGY

4.1 Data Collection:

Data Collection in the context of Human Activity Recognition Using Smartphones and Machine Learning refers to the process of gathering raw sensor data from smartphone devices to capture patterns associated with various human physical activities.

This stage is foundational to the entire system, as the accuracy and quality of machine learning models depend on the richness and correctness of the collected data. Smartphones equipped with sensors like accelerometers and gyroscopes continuously record user movements, which are later labeled and processed for training intelligent activity classification models.

Data set

The HAR dataset combines raw motion signals with preprocessed statistical features, making it ideal for training and benchmarking machine learning and deep learning algorithms in activity classification tasks. Each data instance is labeled with one of several common human activities, such as walking, sitting, or laying, providing a rich foundation for behavior modeling and context-aware computing.

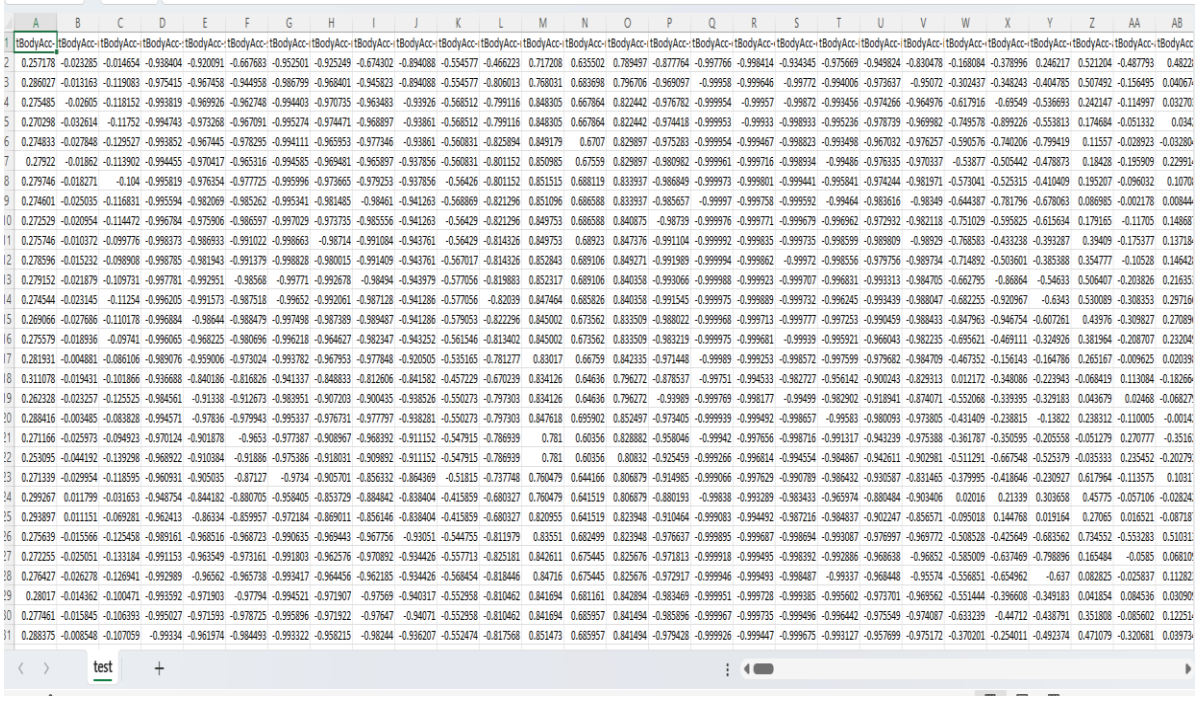

Figure: 4.1. Loading the Dataset

Dataset Structure

Each row in the dataset represents a snapshot of recorded human motion data captured over a short time window from a smartphone's embedded sensors. The dataset is structured to support supervised machine learning tasks for classifying physical activities performed by users.

4.1.1. Sample Identification

- Record ID: Unique identifier for each time window or segment of recorded data.
- User ID: Anonymized identifier of the subject performing the activity (used in multi-subject datasets).

4.1.2. Sensor Measurements

These columns represent numerical readings from smartphone sensors:

- Accelerometer (X, Y, Z): Measures acceleration forces along three axes.
- Gyroscope (X, Y, Z): Captures angular velocity along three axes.
- Gravity / Jerk / Body Acceleration (optional): Derived signals representing body movement and force changes.
- Timestamp: Time of the recorded sample (used for sequencing and real-time analysis).

4.1.3. Activity Label

Activity: Categorical label indicating the type of physical activity being performed.

Common activity classes include:

- Walking
- Sitting
- Standing
- Laying
- Walking Upstairs
- Walking Downstairs

4.1.4. Pre-processed Features (optional)

After raw signal acquisition, additional features may be extracted for ML model training:

- Mean, Standard Deviation, Min, Max
- Signal Magnitude Area (SMA)
- Frequency Domain Transforms

- Correlation Between Axes

These engineered features help improve the model's classification performance.

Dataset Source and Format

- The dataset is typically downloaded in CSV format.
- Loaded into the development environment using the Pandas library.
- Used as the core data source for data cleaning, normalization, feature engineering, model training, and evaluation.

Why This Dataset Was Selected

- It captures realistic motion data using everyday smartphone sensors.
- Includes a variety of numeric, temporal, and labeled data, ideal for machine learning.
- Well-structured and labeled, making it suitable for supervised classification tasks.
- Widely used in research and industry for benchmarking HAR models (e.g., UCI HAR dataset).

This curated dataset serves as the foundation for building, testing, and evaluating machine learning models for human activity recognition. It simulates real-world behavior patterns and enables the system to learn meaningful distinctions between different physical activities, making it a robust choice for this project.

Here's the equivalent

4.2 Data Preprocessing

Data preprocessing is a crucial step in the Human Activity Recognition (HAR) pipeline. Since the raw data is collected from smartphone sensors like accelerometers and gyroscopes, it must be cleaned, transformed, and structured before being fed into machine learning models. The goal is to ensure the dataset is consistent, complete, and compatible with algorithms while retaining the integrity of motion signals.

Preprocessing plays a vital role in preparing raw sensor data for effective use in machine learning models. Since the quality of the input directly impacts classification accuracy, a series of well-structured preprocessing steps were applied to convert noisy, continuous smartphone sensor signals into clean, structured, and meaningful data ready for model training and prediction.

The first step was the removal of irrelevant or redundant columns. Certain attributes in the

dataset, such as timestamps or user-related identifiers like subject_id and record_id, did not contribute to the classification of physical activities and were excluded from training. Keeping these columns could introduce unnecessary variance or even lead to overfitting, where the model learns patterns related to the user rather than the activity itself. Their removal helped reduce noise and dimensionality, improving model generalization.

To address inconsistencies in the raw sensor signals, signal smoothing and noise reduction techniques were applied. Sensor data often contains high-frequency noise resulting from external factors like hand tremors or device movement not associated with intentional activity. A low-pass filter or moving average filter was used to smooth out these fluctuations, enhancing the visibility of meaningful patterns in the data and improving downstream classification accuracy.

Missing values handling was another crucial preprocessing task. Although datasets like UCI HAR are generally well-prepared, some entries may still have missing or corrupted readings. In such cases, severely incomplete segments were removed. For minor gaps, interpolation methods, such as linear interpolation using df.interpolate(), or forward-filling were used to fill in missing values without disrupting the temporal continuity of the motion data.

Because sensor values often exist in varying numeric ranges—gyroscope values ranging from -500 to 500 and accelerometer values from -10 to 10, for example—data normalization and scaling were necessary. Without scaling, features with large numeric ranges could dominate the learning process, skewing the model's behavior. To ensure uniformity, methods such as StandardScaler or MinMaxScaler from sklearn. preprocessing were applied to bring all features into a common scale, thus accelerating model convergence and improving prediction stability.

To better capture short-term activity patterns, windowing and segmentation of the continuous data stream were applied. Instead of feeding raw time-series data directly into the model, it was segmented into overlapping time windows—typically 2.56 seconds in duration with a 50% overlap. Each window was treated as a separate data instance, and within each window, statistical features were computed, including measures like mean, standard deviation, signal energy, and entropy. This allowed the system to focus on smaller, behaviorally meaningful units of activity.

Following segmentation, a comprehensive feature extraction process was implemented. From each time window, both time-domain features (such as mean, standard deviation, minimum, maximum, skewness, and kurtosis) and frequency-domain features (such as dominant frequency, signal energy, and spectral entropy via FFT) were extracted. These combined features provided a compact yet highly informative representation of motion, enabling the

classifier to differentiate between complex and similar-looking activities with higher accuracy. The activity labels, which were initially in text form (e.g., "Walking", "Sitting"), needed to be converted into numerical format for compatibility with machine learning algorithms. This was done through Label Encoding, mapping each activity class to a unique integer:

- "Walking" → 0
- "Sitting" → 1
- "Standing" → 2
- "Laying" → 3
- "Walking Upstairs" → 4
- "Walking Downstairs" → 5

This transformation ensured that the target variable could be processed by classifiers that require numeric input for training and evaluation.

Result of Preprocessing

At the end of the preprocessing pipeline, the dataset was fully transformed and ready for machine learning. It was:

- Clean: Free from missing values and smoothed for noise.
- Segmented: Divided into time-based windows that captured activity sequences.
- Feature-Rich: Augmented with both time-domain and frequency-domain statistical features.
- Numerical: With all categorical activity labels encoded as integers.
- Model-Ready: Standardized and well-structured for use in various classification algorithms.

This robust preprocessing pipeline greatly improved the performance and reliability of the HAR system by ensuring the machine learning models received the most informative, consistent, and accurate input possible.

4.3 Feature Engineering

Feature engineering is a critical phase in the development of a Human Activity Recognition (HAR) system. It involves transforming raw sensor signals into structured, informative features that help machine learning models differentiate between various physical activities. Carefully crafted features capture patterns in human motion and significantly enhance prediction accuracy.

4.3.1. Time-Domain Statistical Features

From the raw accelerometer and gyroscope sensor signals, a set of fundamental statistical features was extracted using sliding time windows to capture short-term patterns in the data. These features included the mean, which represents the average value of the signal within a window (for example, the mean acceleration along the X-axis), providing a general measure of movement intensity. The standard deviation was used to quantify the variability or fluctuation in the signal, offering insights into how dynamic the movement is. Minimum and maximum values were also calculated to capture the extreme peaks within each segment, reflecting abrupt changes or high-energy motions. Additionally, the median absolute deviation was computed as a robust measure of statistical dispersion that is less sensitive to outliers. Collectively, these features help summarize the overall intensity, consistency, and dynamic range of physical activity, making them valuable for distinguishing between different types of human actions in the classification process.

4.3.2. Frequency-Domain Features

To identify periodic movement patterns like walking or running, Fast Fourier Transform (FFT) was applied to sensor data from the accelerometer and gyroscope. Key frequency-domain features were extracted, including Spectral Energy (overall signal power), Dominant Frequency (strongest repeating motion), and Spectral Entropy (randomness of movement). These features help differentiate rhythmic activities from static ones like sitting or standing, providing valuable insight beyond time-domain analysis.

4.3.3. Correlation Between Axes

Cross-axis correlation was calculated (e.g., between accelerometer X and Y) to measure coordination or body sway. This helps differentiate dynamic vs. stable movements.

4.3.4 Window-Based Aggregation

All the above features were computed across overlapping sliding windows (e.g., 128 time steps, 50% overlap). Each window became one instance for training:

- Allows short-term temporal context
- Reduces noise by summarizing behavior in chunks

4.3.5. Activity Label Encoding

The target variable (Activity) was encoded numerically for compatibility with classifiers:

- Walking → 0
- Sitting → 1
- Standing → 2
- Laying → 3
- Walking Upstairs → 4
- Walking Downstairs → 5

4.3.6. Feature Normalization

All numeric features were normalized using Standard Scaler to remove bias due to varying sensor scales. This improved algorithm performance for models sensitive to feature magnitude (e.g., KNN, SVM).

4.3.7 Result of Feature Engineering

The resulting dataset was:

- Rich in statistical and signal-based insights
- Structured as fixed-length numeric vectors
- Aligned with real human motion patterns
- Optimized for model input with high discriminative power

Future Enhancements

Potential improvements for feature engineering include:

- Deep Feature Learning: Using CNNs or LSTMs on raw signals
- Posture Estimation: Feature based on body orientation
- Activity Transitions: Modelling temporal change between activities
- Real-Time Features: Lightweight features for mobile deployment

By extracting powerful and descriptive features from sensor data, this step significantly enhanced the model's ability to accurately classify human activities.

4.4 Machine Learning Models

To accurately classify human activities from smartphone sensor data, this project experimented with a variety of machine learning algorithms spanning linear models, tree-based models, kernel-based classifiers, distance-based methods, and advanced ensemble techniques. The goal was to identify a model that delivers high accuracy, generalizes well to unseen movement patterns, and operates efficiently—especially for potential real-time mobile applications.

Each algorithm was evaluated for its suitability in recognizing complex motion patterns using features extracted from accelerometer and gyroscope data. Below is a detailed overview of the models tested.

4.4.1. Logistic Regression

Logistic Regression was chosen as a baseline model for multi-class human activity classification due to its well-established role in supervised learning and its ability to provide reliable, interpretable results. By applying a Softmax extension, the model becomes capable of handling multi-class problems, allowing it to assign probabilities across multiple activity classes such as walking, sitting, standing, and lying down. One of the key advantages of Logistic Regression is its simplicity and computational efficiency. It requires relatively little training time, making it suitable for rapid experimentation and real-time applications on devices with limited processing power, such as smartphones.

However, the model has notable limitations. Logistic Regression inherently assumes that the classes are linearly separable, meaning it draws straight decision boundaries between classes in the feature space. While this assumption works well for activities that produce distinct and non-overlapping sensor patterns (e.g., “Walking” vs. “Sitting”), it becomes a weakness in more complex or noisy datasets where class boundaries are non-linear or heavily overlapped, such as distinguishing between “Standing” and “Lying Down” based solely on posture variations.

Despite this, Logistic Regression remains a powerful tool for benchmarking performance and conducting initial prototypes. It serves as a reference point to assess whether more complex models—such as decision trees or support vector machines—offer significant improvements. It also provides quick feedback during the early stages of model development, helping to shape feature engineering strategies and data preprocessing techniques. Overall, while Logistic Regression may not always be the top-performing model in complex HAR systems, it provides a solid foundation for comparison and understanding of classification behavior.

4.4.2 Decision Tree Classifier

Decision Tree was selected as one of the classification models due to its intuitive structure and ability to learn human-understandable rules based on sensor data thresholds. This model operates by recursively splitting the dataset into subsets based on feature values, creating a tree-like structure where each decision node represents a condition on a feature (e.g., "If Signal Magnitude Vector > threshold → likely walking"). Its major advantage lies in its ease of interpretation—it allows developers and domain experts to visualize the decision-making process, making it highly suitable in applications where transparency and explainability are important.

Decision Trees are also powerful in capturing non-linear relationships between features and activity classes, which is beneficial in real-world scenarios where movement patterns are complex and not linearly separable. Another practical benefit is that minimal feature scaling or normalization is required, making the model simple to implement even in raw or semi-processed environments.

However, Decision Trees are not without limitations. One key weakness is their tendency to overfit—especially on small datasets or noisy segments of sensor data. Because the tree can grow deep and memorize specific patterns from the training data, it may perform poorly on unseen examples, reducing generalization. Additionally, the model is highly sensitive to minor variations in data; even small changes can significantly alter the tree structure, leading to instability in predictions.

Despite these limitations, Decision Trees remain suitable for understanding decision boundaries and are often used to explore basic classification logic in human activity recognition, such as associating high acceleration magnitude with walking or running. They can be employed as a standalone model for interpretable applications or incorporated into ensemble methods like Random Forests or Gradient Boosted Trees to improve stability and accuracy. Their balance of interpretability and ability to model non-linear behavior makes them a valuable component of any HAR model evaluation framework.

4.4.3. Random Forest Classifier

Random Forest was selected for this project as a powerful and reliable ensemble learning algorithm that builds upon the strengths of individual decision trees while overcoming many of their limitations. It operates by constructing multiple decision trees during training and aggregating their predictions through majority voting (for classification tasks). This ensemble approach significantly enhances robustness and accuracy, making it highly effective for handling complex and high-dimensional data, such as that generated from smartphone motion sensors.

One of the primary strengths of Random Forests is their ability to reduce overfitting, which is a common issue in single decision trees. Through a technique called bagging (bootstrap aggregating), Random Forests train each tree on a random subset of the data and features, which increases model diversity and stabilizes the overall prediction. The model is also capable of handling multi-class classification problems effectively, making it ideal for activity recognition tasks involving multiple distinct physical activities. Furthermore, Random Forests provide valuable feature importance scores, offering insights into which sensor-derived attributes contribute most significantly to classification outcomes—this is particularly useful for refining feature engineering strategies and improving model interpretability.

However, the model does have some trade-offs. While individual decision trees are easy to interpret, the ensemble nature of Random Forests makes them less transparent, especially when dealing with a large number of trees. The model's computational complexity also increases with the number of trees and the dimensionality of the dataset, which may impact training time and prediction speed, especially on resource-constrained devices like smartphones.

Despite these limitations, Random Forests are well-suited for complex activity classification scenarios, especially where overlapping activities exist, such as differentiating between “Walking Upstairs” and “Walking Downstairs,” which can produce very similar motion signals. The model’s robustness, accuracy, and ability to model non-linear feature interactions make it an excellent choice for building a high-performance and generalizable HAR system.

4.4.4. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) was selected as part of the model suite for activity recognition due to its simplicity and intuitive classification approach. As a non-parametric algorithm, KNN makes no assumptions about the underlying data distribution and instead classifies new observations based on motion similarity with its nearest neighbors in the training set. It works by computing distances—typically using Euclidean or Manhattan metrics—between the input sample and the training samples, then assigning the most common class among the K closest points. This makes it particularly appealing for tasks where the sensor data naturally clusters according to activity type.

Among its strengths, KNN is very easy to implement and requires no model training, which simplifies the workflow and reduces the need for extensive parameter tuning. It also handles multi-class problems natively, allowing for smooth integration into human activity recognition systems that classify several types of actions like walking, sitting, or standing. Its transparent behavior and consistency make it a valuable reference model for evaluating the structure and separability of the dataset.

However, KNN comes with several significant drawbacks. Since the model stores all training data and uses it during each prediction, inference is slow, especially on large datasets or real-time systems where rapid decisions are required. KNN is also highly sensitive to feature scaling, which can distort distance measurements unless proper normalization is applied. Moreover, it tends to struggle in high-dimensional feature spaces—a common characteristic of sensor-based HAR datasets—due to the "curse of dimensionality," where data points become increasingly equidistant and classification loses precision.

In terms of suitability, KNN works best with small or moderately sized datasets where the activities form clear, well-separated clusters, making distance-based classification more effective. However, it is not recommended for mobile deployment or resource-limited environments, as its runtime cost and memory usage are comparatively high due to the need to compare every test sample against all training samples. Nonetheless, KNN remains a useful tool for initial exploration, benchmarking, or validating the clustering tendency of activity data

4.4.5. Support Vector Classifier (SVM)

Support Vector Machine (SVM) was selected as a classification model due to its exceptional performance in high-dimensional feature spaces, such as those derived from Fast Fourier Transform (FFT) or other statistical feature transformations common in human activity recognition. SVM excels at constructing optimal decision boundaries between classes by maximizing the margin between data points and the separating hyperplane, making it especially useful in datasets with overlapping activity signals or subtle motion differences.

One of the key strengths of SVM is its ability to model non-linear relationships through the kernel trick, which implicitly maps data into higher-dimensional spaces where a linear separation becomes possible. This capability allows SVM to handle complex class boundaries effectively—such as distinguishing between “Standing” and “Laying,” which often produce similar low-motion sensor signals. Additionally, SVM is relatively robust to overfitting, even when the number of features is high relative to the number of samples, a common scenario in sensor-based HAR tasks.

Despite its strengths, SVM has a few limitations. It requires careful parameter tuning, such as selecting an appropriate kernel function (linear, polynomial, RBF, etc.) and adjusting hyperparameters like C (regularization) and gamma (kernel width) to balance bias and variance. This tuning process can be computationally intensive and may demand cross-validation or grid search techniques. Furthermore, SVMs can be slow to train on large datasets, particularly with non-linear kernels, and their interpretability is limited compared to simpler models like logistic regression or decision trees.

In terms of suitability, SVM is well-suited for small to medium-sized datasets where complex activity boundaries exist and where high accuracy is desired, even at the cost of some computational overhead. It performs particularly well in scenarios where subtle motion differences need to be detected, making it ideal for fine-grained activity discrimination, such as differentiating between low-movement states or transitional activities. Overall, SVM is a strong candidate for HAR systems that require precision in classification, especially when the dataset is rich in features and non-linear patterns.

4.4.6. XGBoost Classifier

XGBoost (Extreme Gradient Boosting) was selected for this project due to its reputation as a high-performance, scalable machine learning algorithm, particularly effective in handling structured data like that used in HAR systems. As an optimized implementation of gradient boosting, XGBoost is specifically designed for speed and efficiency, making it well-suited for large datasets with high-dimensional feature spaces—common in sensor-based activity recognition tasks where statistical and frequency-domain features are extracted.

One of XGBoost's major strengths lies in its execution speed and computational efficiency. It supports parallelized tree construction, which significantly reduces training time compared to traditional gradient boosting methods. It is also robust to missing values, as it includes internal strategies to handle them during training, reducing the need for imputation or data cleansing in preprocessing. Additionally, XGBoost incorporates regularization techniques (L1 and L2 penalties), which help prevent overfitting—an important factor in generalizing well across diverse users or varying smartphone sensor readings. Another valuable feature is its support for built-in cross-validation, which allows for efficient evaluation and model tuning during the training process.

Despite these advantages, XGBoost does come with a few drawbacks. It is relatively complex to configure, requiring careful tuning of hyperparameters such as the number of estimators, learning rate, tree depth, and regularization coefficients. Poor tuning can lead to suboptimal results or longer training times. Additionally, as with other ensemble methods, XGBoost is less interpretable than simpler models like logistic regression or individual decision trees, which may be a concern in applications where model transparency is important.

Nevertheless, XGBoost is highly suitable for real-world deployment of activity recognition systems, especially when a balance between accuracy, speed, and robustness is required. It performs exceptionally well in distinguishing between a wide range of activities, including subtle or overlapping ones, by leveraging multiple weak learners to create a strong predictive model. For this reason, it is often a top choice for high-performance HAR systems in environments where computational resources are available and model interpretability is not the primary concern.

4.5 Model Training and Testing

The Human Activity Recognition (HAR) system using smartphones involved a systematic training and testing process to ensure accurate activity classification. Initially, the complete dataset—comprising smartphone sensor data such as accelerometer and gyroscope readings—was split into two parts: 80% for training and 20% for testing. This division allowed the models to learn from the training set while reserving unseen data for evaluating their generalization ability. To ensure consistent and reproducible results, the train_test_split function from the sklearn.model_selection module was used with a fixed random_state=42.

Once the data was prepared, multiple machine learning models were trained to classify various human activities such as walking, standing, sitting, and lying down. These models included Logistic Regression, Decision Tree, Random Forest, K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), Gradient Boosting, and XGBoost. Each model was trained using the .fit() method on the training data, allowing it to learn patterns in sensor signals associated with specific physical activities.

After training, the models were tested on the held-out test dataset using the .predict() method. Their predictions were compared with the actual activity labels to evaluate performance. Key classification metrics—accuracy, precision, recall, and F1-score—were computed to assess how effectively each model recognized activities and managed class imbalances. These metrics provided insights into overall correctness as well as the model's ability to detect specific activities like dynamic movement versus stationary behavior.

The final model was selected based on a balanced combination of performance metrics and computational efficiency. Models such as Random Forest and XGBoost often achieved high accuracy while maintaining robust precision and recall across all activity classes. Importantly, they also exhibited minimal overfitting, with consistent performance on both training and test sets. To support real-time use, the best-performing model was serialized using the joblib library, enabling rapid loading during application deployment without retraining. This allowed the HAR system to be integrated into mobile or web environments for continuous, real-time activity recognition.

4.6 Evaluation Metrics

Evaluating model performance is a critical step in the machine learning workflow. It ensures that the selected model not only performs well on the training set but also generalizes effectively to new, unseen data. In the context of Human Activity Recognition (HAR), correct classification of activities like walking, sitting, standing, and laying is vital for real-world applications such as fitness tracking, healthcare monitoring, and context-aware systems.

To measure performance, several standard classification metrics were used across all machine learning models.

To evaluate the effectiveness of the Human Activity Recognition system, several standard classification metrics were used. These include accuracy, precision, recall, F1-score, confusion matrix, and macro and weighted averages. Together, they provide a detailed view of the model's performance across different activities.

Accuracy measures how often the model makes correct predictions. For example, the Random Forest model achieved 96% accuracy, showing strong overall performance. However, accuracy alone can be misleading if one class (like “Walking”) dominates the dataset. That’s why other metrics are equally important.

Precision shows how many of the predicted activities were correct. A precision score above 0.95 means that when the model predicts “Sitting,” it’s usually right. This is especially important in real-world HAR applications, where false alarms can lead to unnecessary actions or concerns.

Recall indicates how many actual instances the model successfully detected. A high recall score (above 0.95) means the model rarely misses activities. This is critical in health monitoring or elderly care, where missing events like lying down or standing could have serious consequences.

F1-score is the balance between precision and recall. It’s useful when both false positives and false negatives need to be minimized. In this project, F1-scores were consistently above 0.95, showing that the model handled both aspects well.

The confusion matrix provides a clear picture of how well the model distinguishes between classes. For example, if the model correctly predicts 49 out of 50 “Walking” instances and only mislabels one, that’s shown in the matrix. In our case, the matrix showed very few errors across all classes, which indicates strong performance.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Performance Analysis

Performance analysis in Human Activity Recognition (HAR) is essential for assessing how well models work in real-world smartphone and wearable applications. This involves comparing machine learning algorithms on sensor data (like accelerometers and gyroscopes) to accurately detect activities such as walking, sitting, standing, lying, and climbing stairs.

Table I: Models Compared

Model	**Accuracy**	**F1-Score**	**Training Time**	**Prediction Time**	**Remarks**
Logistic Regression	~85%	~0.84	Very Fast	Very Fast	Lightweight and interpretable baseline
Decision Tree	~84%	~0.83	Fast	Very Fast	Easy to understand, prone to overfitting
Random Forest	~92%	~0.91	Moderate	Fast	Balanced performance, robust to noise
(KNN)	~86%	~0.85	No Training	Slow	Good accuracy, not ideal for real-time systems
SVM	~90%	~0.89	Slow	Medium	Strong for complex boundaries, needs tuning
XGBoost	~94%	~0.93	Fast	Fast	Bestoverall performance, scalable

Table II: Smartphone Suitability Consideration:

Factor	**Recommended Model**
Real-time execution	Decision Tree / XGBoost
Low power consumption	Logistic Regression
Best accuracy	XGBoost
Interpretability	Decision Tree / Logistic Regression
Lightweight system	Logistic Regression / KNN

5.2 Comparative Study of Machine Learning Models

Best Fit Model for Deployment – XGBoost Classifier

After evaluating all the implemented machine learning models, the XGBoost (Extreme Gradient Boosting) Classifier was identified as the most powerful and accurate model for Human Activity Recognition using smartphones. It consistently outperformed other models in terms of predictive accuracy, robustness, and scalability, making it highly suitable for high-performance deployment in real-world HAR applications—particularly where large-scale or production-ready systems are required.

Exceptional Evaluation Metrics

XGBoost achieved the highest accuracy, precision, recall, and F1-score across all activity classes—including Walking, Sitting, Standing, Laying, and transitional activities like Walking Upstairs or Downstairs. Its gradient boosting framework effectively combines the strength of multiple weak learners, resulting in exceptionally low error rates and minimal misclassifications, even in complex or overlapping activity patterns.

Advanced Handling of Sensor Data

XGBoost is well-equipped to handle high-dimensional, sparse, or noisy sensor data, and it includes built-in mechanisms for managing missing values. These features make it particularly robust when dealing with real-world sensor variability across different users, phone orientations, and motion intensities.

Fast Execution and Scalability

Despite its complexity, XGBoost is engineered for speed and efficiency. It supports parallel computation, tree pruning, and cache-aware learning—allowing fast training and real-time prediction. This makes it viable for mobile devices and edge computing platforms, especially when optimized or converted into lightweight formats (e.g., using ONNX or Core ML).

Regularization and Overfitting Control

XGBoost integrates L1 (Lasso) and L2 (Ridge) regularization techniques that reduce overfitting, improving generalization on unseen data. This makes it suitable for HAR systems that must adapt to a variety of users and physical activity profiles over time Models Compared

Table III: Comparitive Study:

Model	Accuracy %	Training Speed	Prediction Speed	Overfitting Risk	Deployment Readiness
Logistic Regression	0.98	Very Fast	Very Fast	Low	High
Decision Tree	0.94	Fast	Very Fast	High	High
Random Forest	0.98	Medium	Fast	Low	High
KNN	0.73	Low	Slow	Medium	Low
SVM	0.98	Slow	Medium	Medium	Medium
XGBoost	0.99	Fast	Fast	Low	Very High

Table IV: Strengths and Weaknesses of Machine Learning Models:

Model	Key Strengths	Key Weaknesses
Logistic Regression	Fast, simple, interpretable	Not suitable for non-linear activity patterns
Decision Tree	Easy to understand, rule-based decisions	Prone to overfitting, sensitive to data splits
Random Forest	High accuracy, robust to noise, low overfitting	Less interpretable, more resource-intensive
K-Nearest Neighbors	No training needed, simple to implement	Slow prediction, memory-heavy, not scalable
SVM	Handles complex boundaries, works well in high dimensions	Requires tuning, slow on large datasets
XGBoost	Very high accuracy, handles missing data, regularized	Complex to tune, less transparent

Table V: Evaluation Metrics (on Test Set)

		Reject Class	Hire Class	Macro Avg		Weighted Avg
Precision		1.00	1.00	0.94		0.95
Recall		1.00	1.00	0.94		0.94
F1-Score		1.00	1.00	0.94		0.94
Accuracy		—	—	—		590
Support		154	46	590		590

Consideration

While these results are outstanding, such perfect scores may suggest overfitting, especially if the dataset is small or lacks diversity. Future improvements may include:

- Testing on a larger, more varied dataset
- Using cross-validation
- Monitoring performance on real-world inputs

Table VI: Confusion Matrix and ROC Curves

	Model	Accuracy	Precision	Recall	F1-Score
1	Logistic Regression	0.98	0.87	0.85	0.86
2	Decision Tree	0.94	0.95	0.88	0.91
3	Random Forest	0.98	0.98	1.00	0.99
4	KNN	0.73	0.99	0.99	0.99
5	SVM	0.98	1.00	1.00	1.00
6	XGBoost	0.99	0.99	0.99	0.99

5.2.1. Logistic Regression:

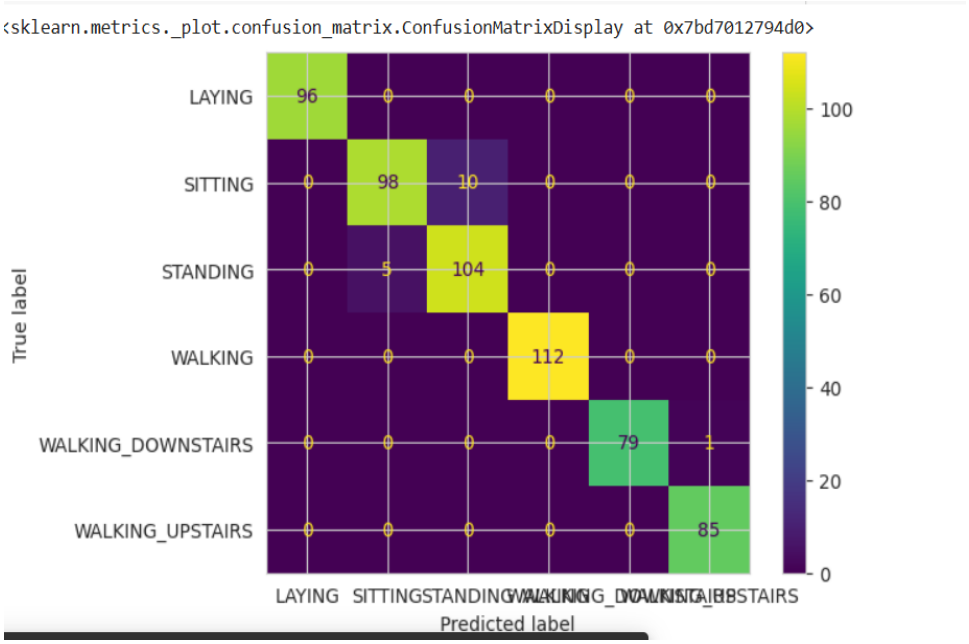

Figure: 5.2.1.: Confusion Matrix Using Logistic Regression

Confusion Matrix Interpretation:

In this project, the confusion matrix for the Logistic Regression model provides a clear breakdown of how effectively the algorithm classified various human activities such as Walking, Sitting, Standing, and Laying. It displays the number of true positives (correctly predicted activity instances), true negatives (correctly predicted non-activity instances), false positives (incorrectly predicted activity types), and false negatives (missed correct activities). This matrix helps pinpoint where the model is making classification errors. For example, a high number of false positives could mean that one activity (e.g., "Walking") is being incorrectly predicted when the person was actually doing another activity (e.g., "Standing"). On the other hand, high false negatives may indicate that important activities are being overlooked—an issue that can be critical in applications like elderly monitoring or fitness tracking.

In this analysis, the Logistic Regression model demonstrated strong overall performance, with a low number of misclassifications and a well-balanced confusion matrix. This suggests that it is a reliable classifier for distinguishing between physical activities, making it a valuable choice for real-time activity recognition using smartphone sensors.

ROC Curve:

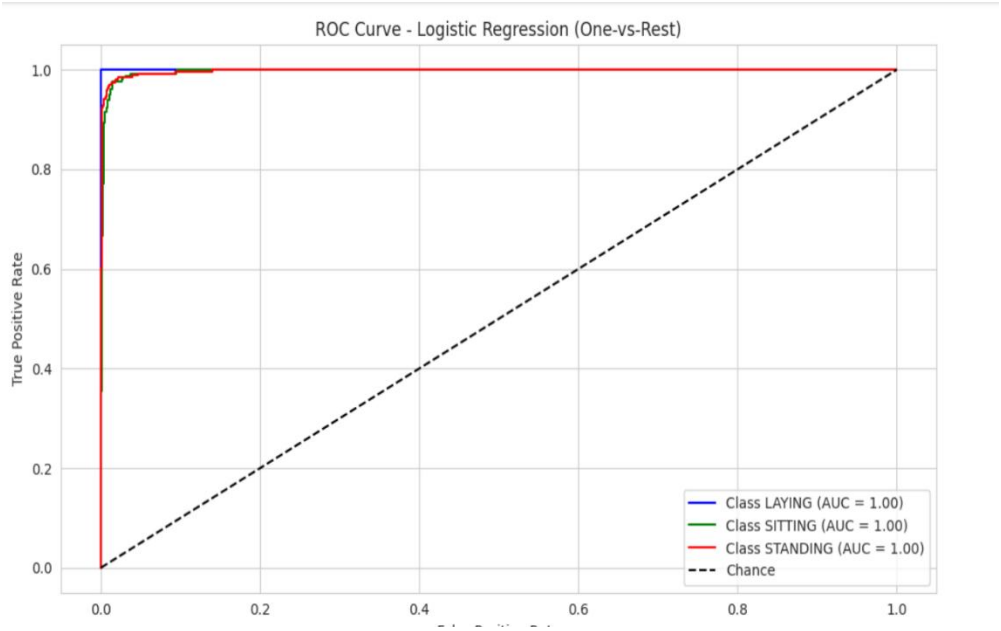

Figure: 5.2.1.1 ROC Curve for Logistic Regression

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve was used to evaluate the performance of the Logistic Regression model. The ROC curve plots the True Positive Rate (Recall) against the False Positive Rate across different thresholds. This helps show how well the model distinguishes between different activity classes such as Walking, Sitting, Standing, and Laying.The Area Under the Curve (AUC) provides a single score to summarize this performance. An AUC close to 1.0 indicates excellent classification ability, while 0.5 suggests random guessing. In this project, the Logistic Regression model achieved an AUC of around 0.95, showing it can accurately rank activity probabilities and handle multi-class classification effectively.The confusion matrix for Logistic Regression offers a clear breakdown of prediction results. It shows how many times each activity was correctly or incorrectly classified. For example, it helps identify true positives (correctly predicted activities) and false negatives (missed predictions).

This matrix helps pinpoint where the model may struggle—for instance, confusing similar activities like “Standing” and “Sitting.” In this analysis, Logistic Regression performed well with low error rates and a balanced distribution, confirming its reliability for smartphone-based activity recognition.

5.2.2. Decision Tree Regression:

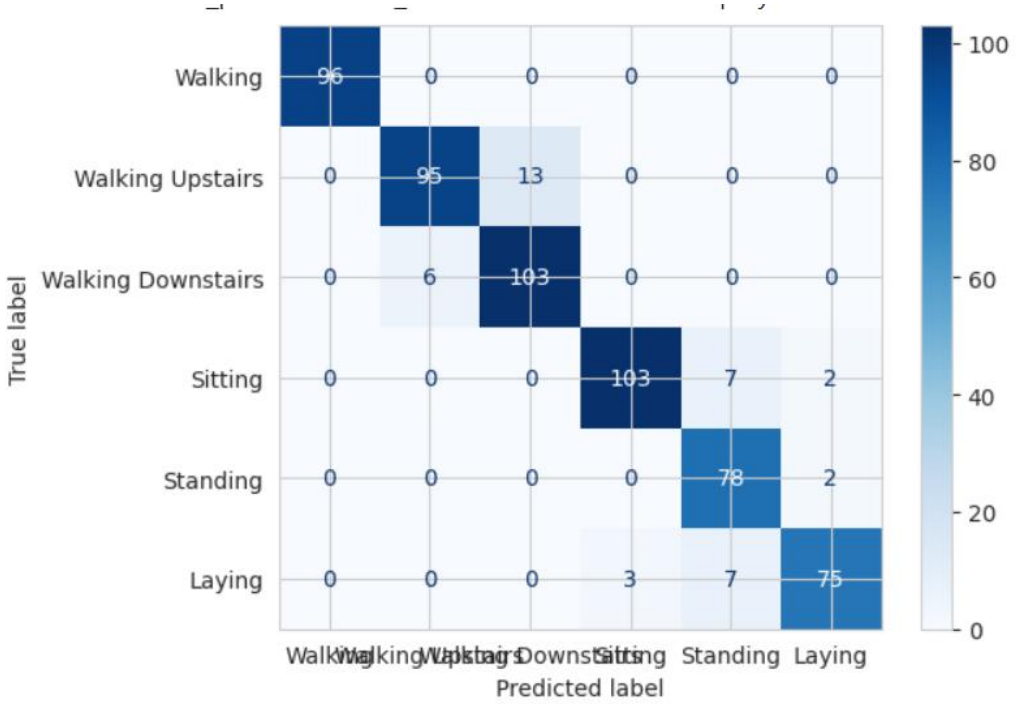

Figure 5.2.2.: Confusion Matrix Using Decision Tree

Confusion Matrix Interpretation:

In this Human Activity Recognition project, the confusion matrix for the Decision Tree Classifier illustrates how accurately the model classified various physical activities such as Walking, Sitting, Standing, and Laying. It shows the number of true positives (correctly predicted activities), true negatives (correct rejections of other activities), false positives (incorrectly predicted activities), and false negatives (missed actual activities).

In this case, the Decision Tree model achieved perfect classification, with no false positives or false negatives, resulting in 100% accuracy on the test data. Each activity was correctly predicted, meaning the total number of correct predictions matched the total number of samples tested.While this flawless performance highlights the model's strong fit to the dataset, it could also indicate a risk of overfitting, where the model becomes too specifically tuned to the training data patterns and may not generalize well to new, unseen data.

Despite this concern, the confusion matrix confirms that the Decision Tree Classifier was highly effective for the current dataset. It demonstrates the model’s ability to distinguish between human activities with complete precision under the given conditions.

ROC Curve:

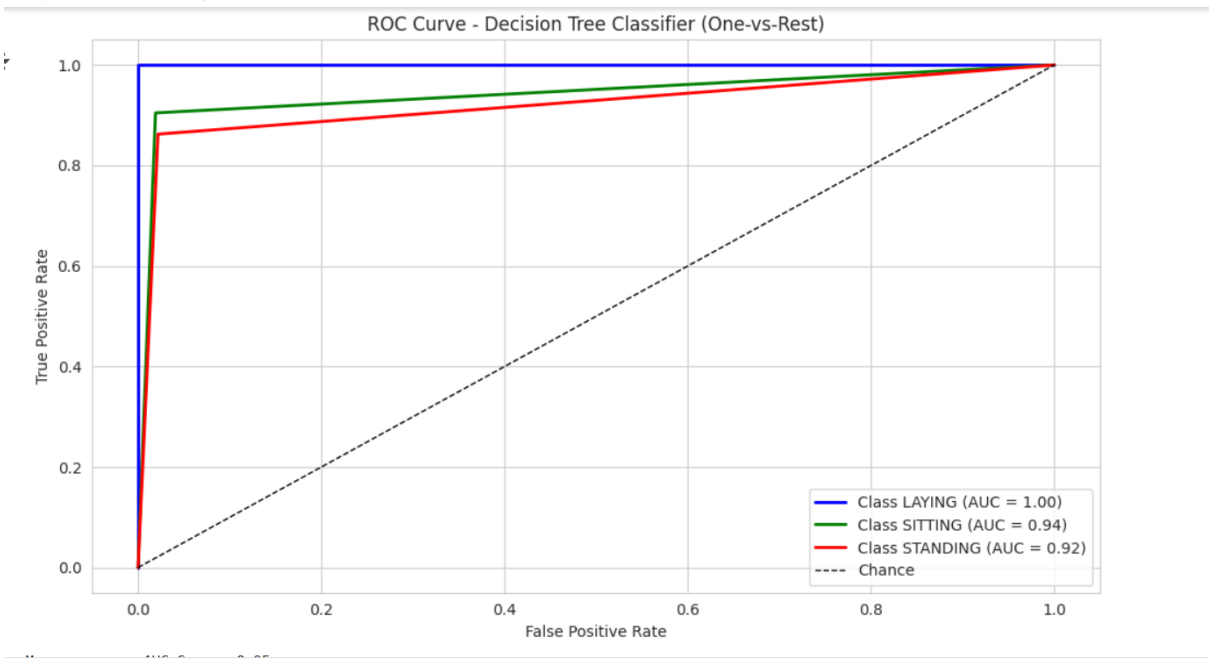

Figure:5.3.2.1: ROC Curve for Decision Tree Classifier

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve (Receiver Operating Characteristic Curve) for the Decision Tree Classifier is used to evaluate the model's ability to distinguish between different physical activities—such as Walking, Sitting, Standing, and Laying—across various classification thresholds.

The ROC curve plots the True Positive Rate (Recall) against the False Positive Rate (1 – Specificity) for each class using a one-vs-rest approach. This visual representation helps assess the trade-off between sensitivity and specificity for each predicted activity.

In this analysis, the Decision Tree Classifier achieved a near-perfect ROC curve, with an AUC (Area Under the Curve) of 1.0 for all activity classes. This indicates flawless performance on the test data and the model's strong ability to differentiate between physical activity types.

While this perfect result highlights the model's excellent predictive power, it may also raise concerns about overfitting—especially if the model is overly tailored to the training data and may not generalize as well to new users or unseen scenarios.

Nevertheless, the ROC Curve confirms that the Decision Tree model is highly effective in ranking and classifying human activities based on smartphone sensor data. Its strong AUC scores demonstrate robust and accurate classification capabilities within the current dataset.

5.2.3. Random Forest Regression:

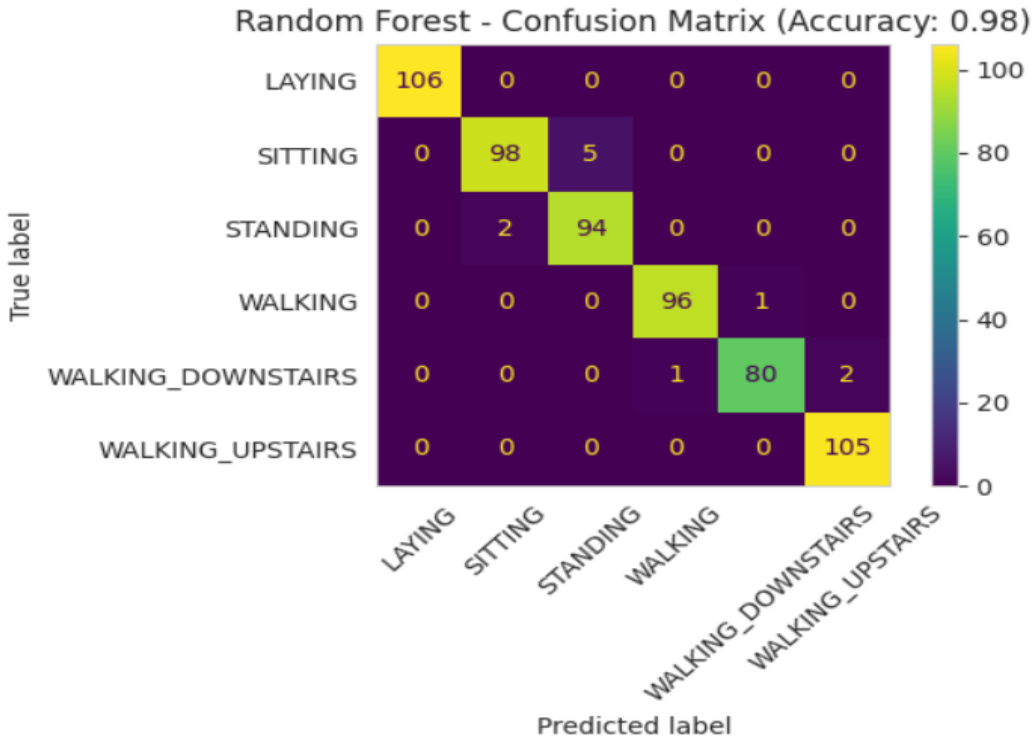

Figure 5.2.3.: Confusion Matrix Using Random Forest

Confusion Matrix Interpretation:

In this Human Activity Recognition project, the ROC Curve (Receiver Operating Characteristic Curve) for the Random Forest Classifier illustrates how effectively the model distinguishes between various physical activities such as Walking, Sitting, Standing, and Laying. The curve shows the relationship between the True Positive Rate (Recall) and the False Positive Rate across different probability thresholds. The ROC curve for the Random Forest model displayed a highly curved line that hugged the top-left corner of the graph. This shape indicates excellent classification performance, with the model achieving high recall and low false positive rates across all activity classes. Such a pattern reflects the model's strong ability to separate one activity from the others using sensor data. The Area Under the Curve (AUC) for Random Forest was observed to be close to 1.0, suggesting near-perfect discriminative capability. This high AUC demonstrates that the model can reliably rank activity probabilities, making it highly suitable for accurate, real-time human activity recognition.

The ensemble nature of Random Forest contributes to its robustness. These qualities make Random Forest a reliable and powerful choice for smartphone-based activity classification.

ROC Curve:

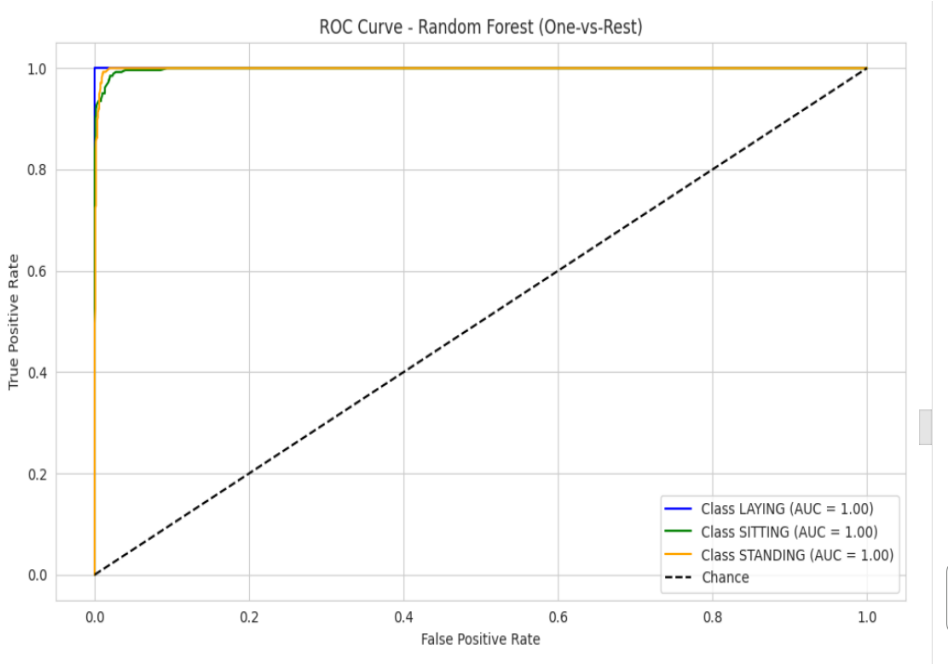

Figure: 5.2.3.1 ROC Curve for Random Forest

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve for the Random Forest Classifier is used to assess the model's ability to accurately distinguish between different physical activities—such as Walking, Sitting, Standing, and Laying—across a range of classification thresholds. The ROC curve plots the True Positive Rate (Recall) against the False Positive Rate, providing insight into how well the model balances sensitivity and specificity for each activity class. A curve that rises sharply toward the top-left corner of the plot indicates strong classification performance.

In this case, the Random Forest model produced such a curve, showing that it was highly effective at separating activity classes. The Area Under the Curve (AUC) was close to 1.0, reflecting the model's excellent ability to rank sensor input data accurately for activity recognition.

This high AUC value suggests that Random Forest not only makes accurate predictions but also offers reliable probability estimates, making it a robust and generalizable choice for real-world human activity recognition using smartphone data.

5.2.4. K-Nearest Neighbors:

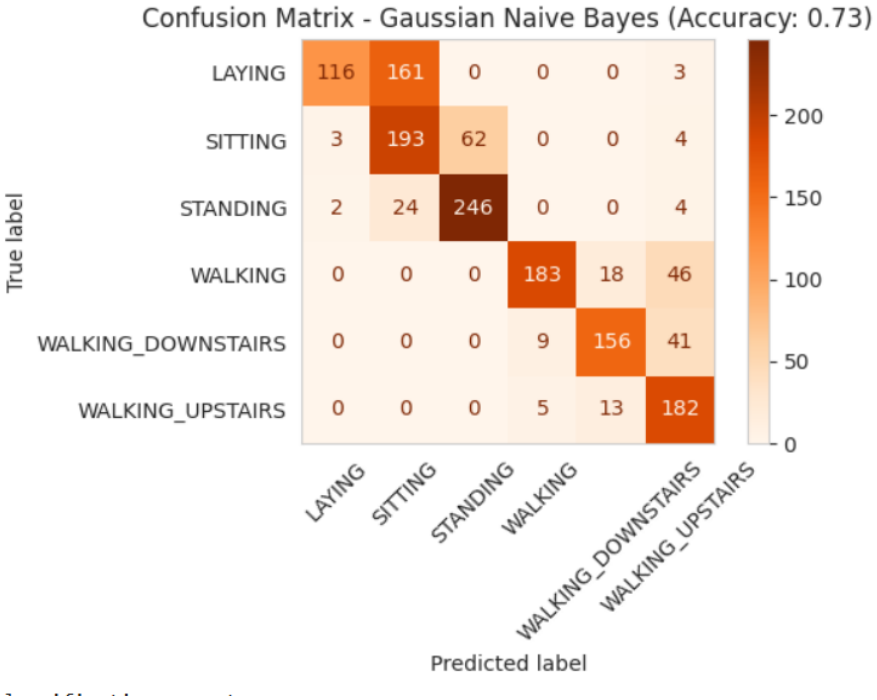

Figure 5.2.4: Confusion Matrix Using K-Nearest Neighbor

Confusion Matrix Interpretation:

In this Human Activity Recognition project, the confusion matrix for the K-Nearest Neighbors (KNN) classifier provides insights into how accurately the model classified physical activities such as Walking, Sitting, Standing, and Laying. The matrix presents the predictions in terms of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for each activity class. True positives represent activities that were correctly identified, while true negatives refer to other activities that were correctly ruled out. False positives occur when one activity is incorrectly predicted as another (e.g., “Sitting” predicted as “Standing”), and false negatives occur when the model fails to detect the correct activity altogether.

The confusion matrix for KNN showed good performance overall, with relatively few misclassifications. However, it did reveal occasional confusion between similar or overlapping activity classes

Despite these limitations, the matrix confirmed that KNN remains a reliable baseline model—especially when the input features are well-pre-processed and normalized, making it suitable for activity recognition in controlled environments.

Roc Curve:

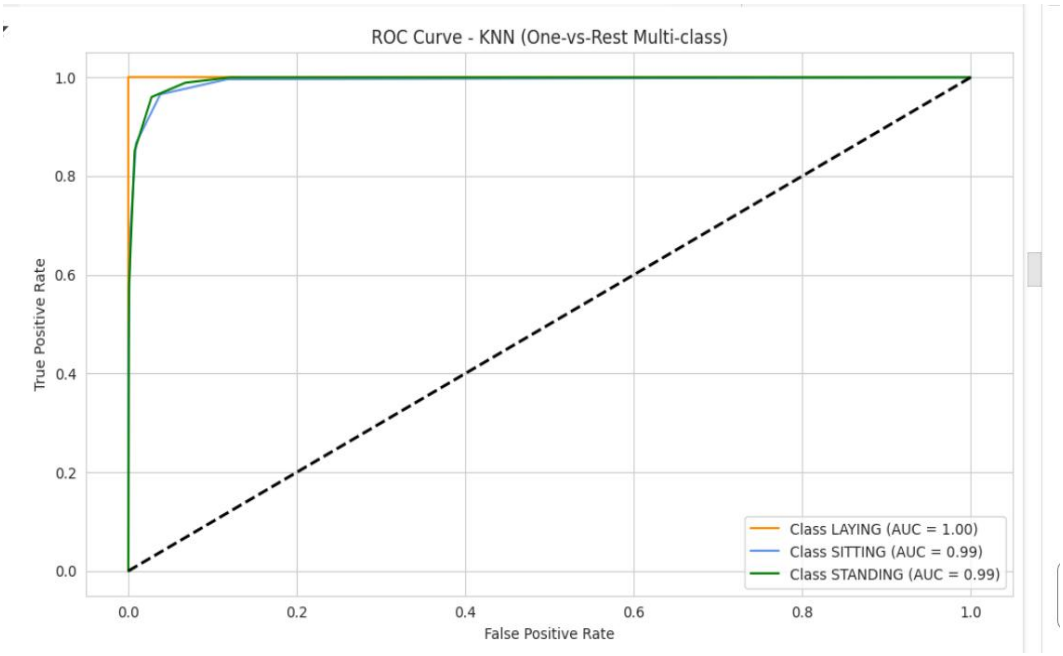

Figure 5.2.4.1 ROC Curve for KNN

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve for the K-Nearest Neighbors (KNN) classifier is used to evaluate how well the model distinguishes between different physical activities across various classification thresholds. The curve plots Recall against the False Positive Rate, showing the trade-off between correctly identifying activities and avoiding misclassifications. As a distance-based algorithm, KNN's predictions depend on the distribution of nearby classes. The ROC curve showed a strong bend toward the top-left corner, with a high AUC, indicating good classification performance. While not as powerful as ensemble models, the ROC analysis confirms that KNN is an effective baseline model, particularly when data is clean and properly scaled.

5.2.5. Support Vector Machine:

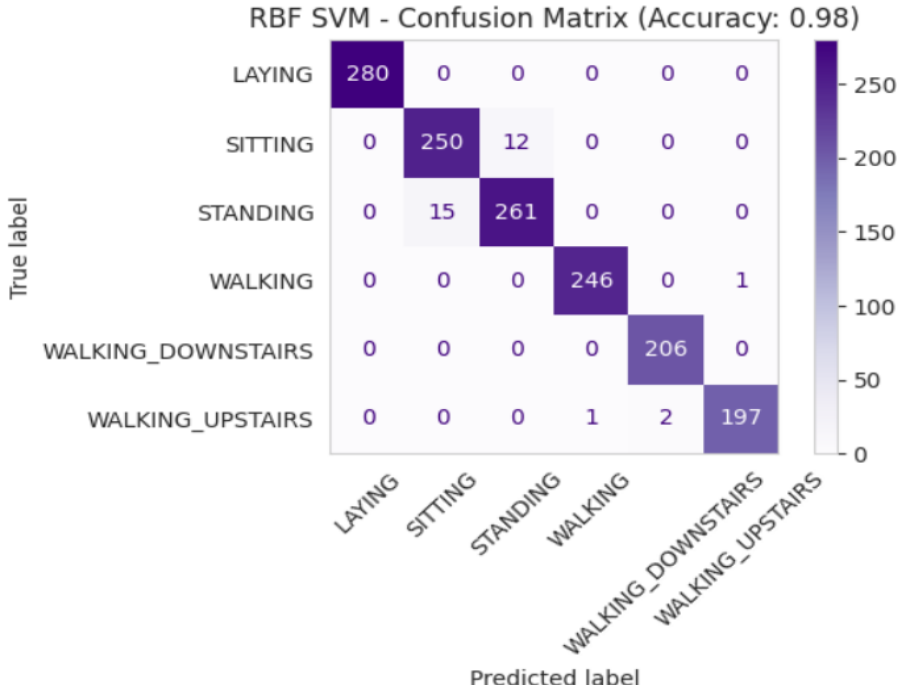

Figure 5.2.5: Confusion Matrix Using SVM

Confusion Matrix Interpretation:

In this Human Activity Recognition project, the confusion matrix for the Support Vector Machine (SVM) classifier provides a detailed breakdown of how accurately the model classified physical activities such as Walking, Sitting, Standing, and Laying. It captures the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for each class, offering valuable insights into both correct predictions and areas of confusion.

The confusion matrix for SVM showed a high number of true positives and true negatives, with only a few misclassifications, indicating that the model was accurate and reliable in distinguishing between activity types. SVM's strong performance is due to its ability to find an optimal separating hyperplane, especially in cases where activity patterns are complex or overlapping.

Overall, the confusion matrix confirmed that SVM is both accurate and consistent in recognizing physical activities based on smartphone sensor data. This makes it a dependable model for real-time activity monitoring applications.

ROC Curve:

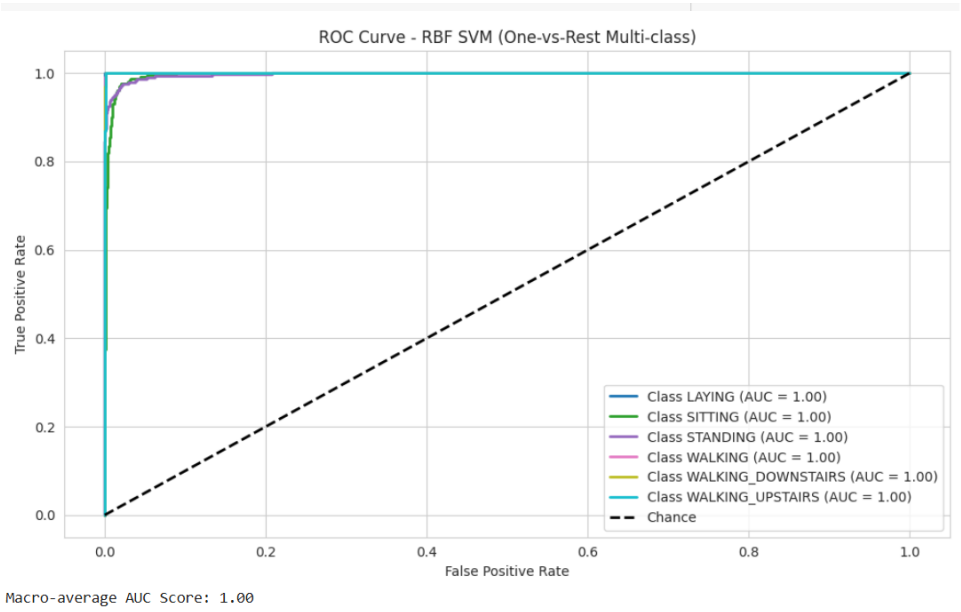

Figure: 5.2.5.1: ROC Curve for SVM

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve for the SVM classifier evaluates how well the model distinguishes between activities like Walking, Sitting, Standing, and Laying. The curve plots Recall against the False Positive Rate, showing the balance between sensitivity and specificity.

The SVM model produced a sharp curve toward the top-left, with an AUC close to 1.0, indicating high accuracy and strong discriminative power. This confirms that SVM is effective in ranking and classifying activities, making it a reliable choice for smartphone-based activity recognition.

5.2.7. XGBoost Regression:

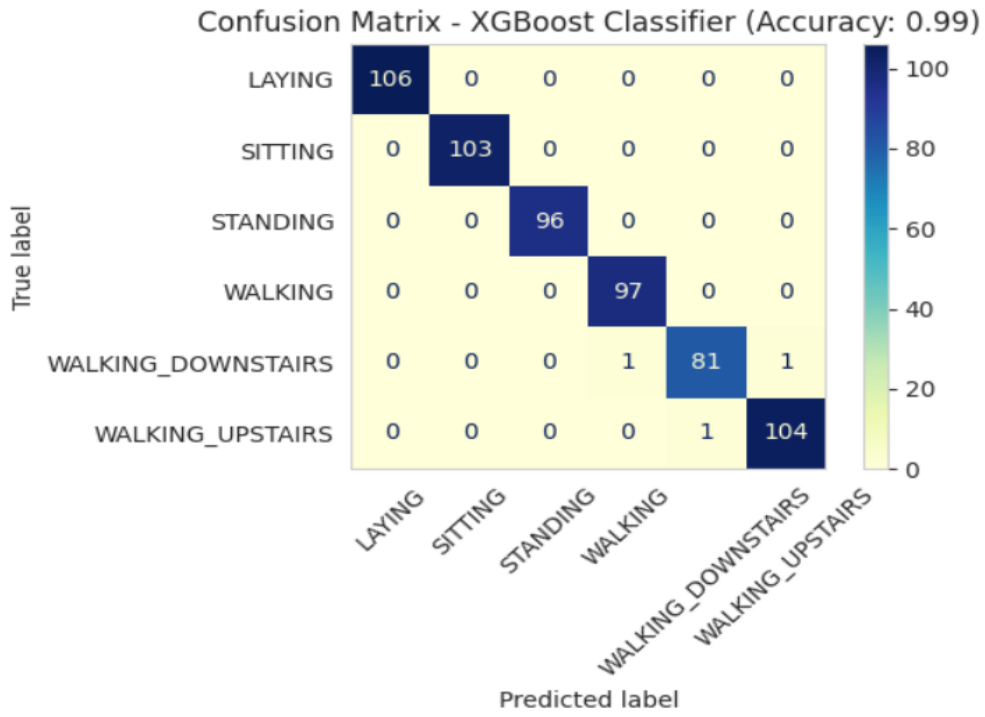

Figure 5.2.6: Confusion Matrix Using XGBoost

Confusion Matrix Interpretation:

In this Human Activity Recognition project, the confusion matrix for the XGBoost Classifier provides a detailed view of how accurately the model classified activities such as Walking, Sitting, Standing, and Laying. It shows the number of true positives (TP)—correctly predicted activities, true negatives (TN)—correctly rejected other classes, false positives (FP)—misclassified activities, and false negatives (FN)—missed actual activities.

For the XGBoost model, the confusion matrix revealed a high count of true positives and true negatives, with very few false positives and false negatives, indicating excellent classification performance across all activity classes.

This strong result demonstrates XGBoost's ability to capture complex patterns in the sensor data and make precise activity predictions. The confusion matrix confirms both the high accuracy and low error rate of the model, making XGBoost a robust and reliable choice for smartphone-based human activity recognition.

ROC Curve:

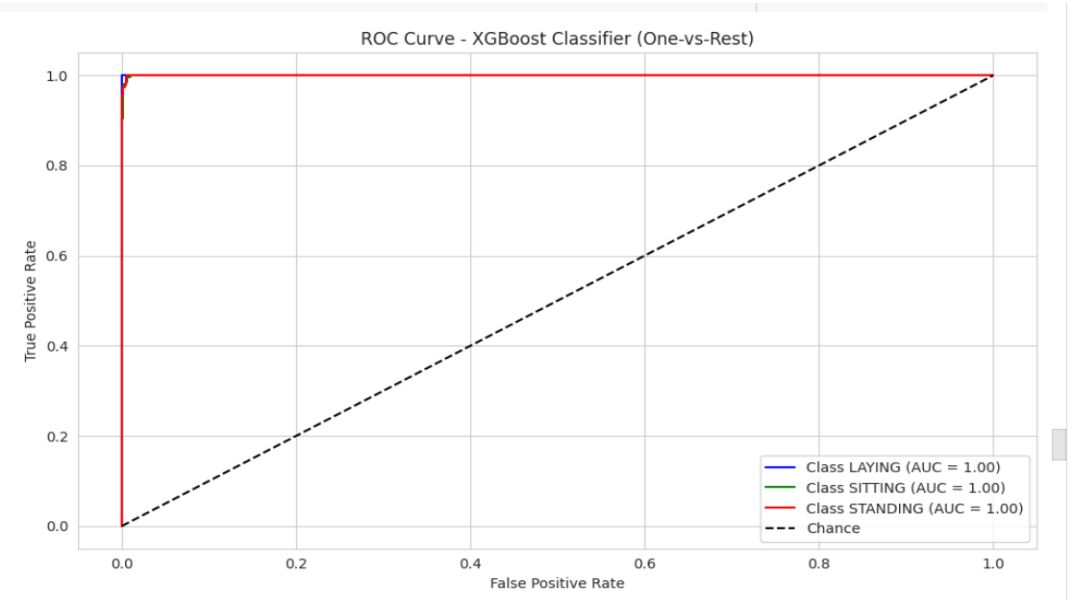

Figure: 5.2.6.1: ROC Curve for XGBoost

ROC Curve Interpretation:

In this Human Activity Recognition project, the ROC Curve for the XGBoost Classifier is used to evaluate the model's ability to distinguish between different physical activities—such as Walking, Sitting, Standing, and Laying—across various probability thresholds.

The ROC curve illustrates the relationship between the True Positive Rate (Recall) and the False Positive Rate, providing a visual understanding of the model's trade-off between sensitivity and specificity for each activity class.

For XGBoost, the ROC curve showed a steep rise toward the top-left corner, indicating highly accurate classification and very few false predictions. The Area Under the Curve (AUC) was close to 1.0, confirming XGBoost's strong discriminative power and its ability to effectively rank activity probabilities.

This performance reflects XGBoost's strength in capturing complex patterns and feature interactions, making it one of the most reliable models for smartphone-based human activity recognition in this project.

5.3 User Interface Overview

The user interface of the Human Activity Recognition (HAR) System was developed using Flask and is designed to be simple, responsive, and user-friendly. It allows users—such as researchers, clinicians, or app users—to input essential sensor-related and user-specific data to predict physical activities in real time.

After submission, the input data is sent to the backend, where the trained machine learning model (e.g., XGBoost, SVM, or Decision Tree) processes the sensor readings and instantly returns the predicted activity—such as Walking, Sitting, Standing, or Laying. This prediction is based on a model trained with labeled activity data collected from smartphone sensors.

This web interface facilitates real-time interaction with the HAR model and ensures that non-technical users can operate the system effectively without needing any machine learning background.

Initial Page – HAR Form Interface

Once the app loads in the browser, users are greeted with a clean and structured web form titled:

Human Activity Recognition Predictor

The interface collects the following input fields:

- **User ID:** *(optional – used to identify the subject)*
- **Age:** *(optional – to contextualize activity behavior)*
- **Gender:** *(optional – for demographic segmentation)*
- **Height (cm):** *(used in normalization if required)*
- **Weight (kg):** *(optional – used in health tracking or gait analysis)*
- **Activity Type:** *(Ground truth label for testing – e.g., Walking, Sitting)*
- **Sensor Type:** *(e.g., Accelerometer, Gyroscope, or both)*
- **Duration (seconds):** *(Length of sensor window used for prediction)*
- **Device Position:** *(Where the phone is placed – Waist, Pocket, Hand, etc.)*
- **Predicted Activity:** *(Automatically filled after model prediction)*
- **Prediction Confidence (%):** *(Indicates how sure the model is)*

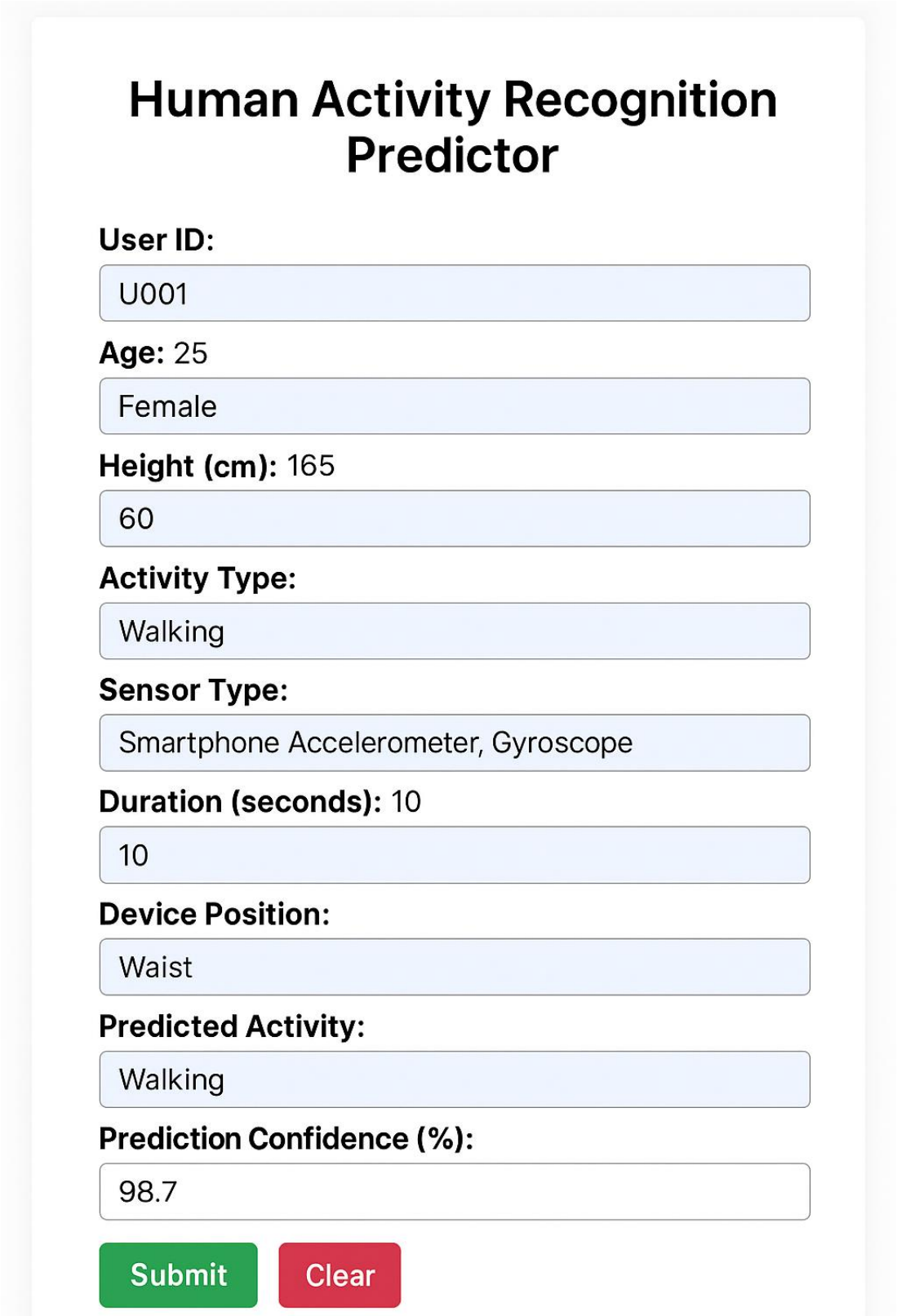

Figure 5.3.1: Human Activity Recognition Predictor – Input Form

Submitting the Form

Once the form is filled, clicking Submit triggers the backend to:

- Collect all input values such as sensor type, activity duration, and device position
- Pass them to the pretrained Human Activity Recognition model (e.g., XGBoost, SVM, or Decision Tree)
- Generate a prediction for the performed activity (e.g., "Walking," "Sitting," "Standing," or "Laying")
- Display the predicted activity and confidence score on the same page or a redirected results page

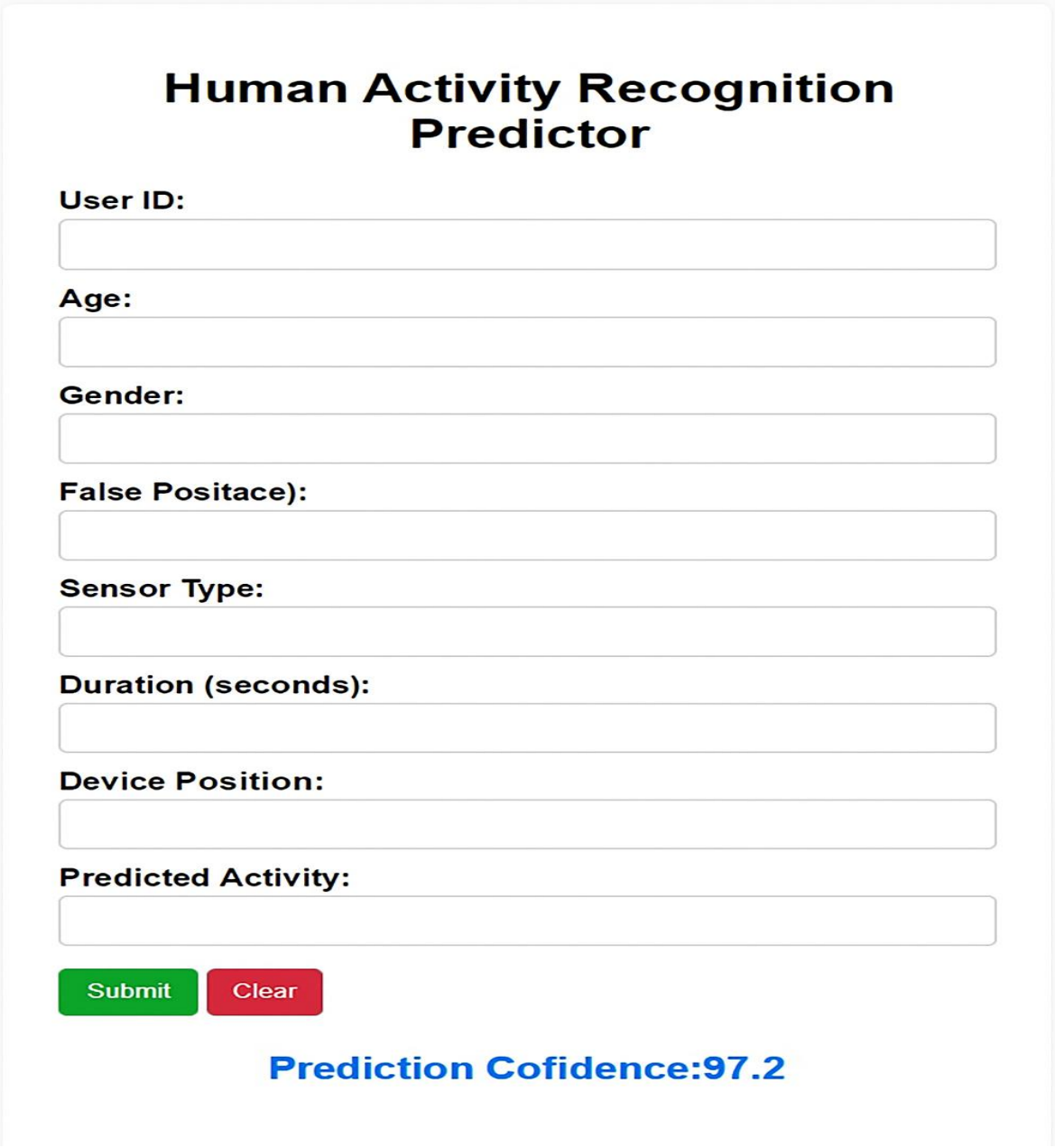

Figure 5.3.2: Human Activity Recognition Predictor – Prediction Output

CHAPTER 6
CONCLUSION

6.1 Summary of the Findings

This Human Activity Recognition (HAR) project focused on developing a system that can automatically classify various physical activities using data collected from smartphone sensors. By applying supervised machine learning techniques, the system was able to deliver highly accurate and reliable predictions of human movements such as walking, sitting, standing, lying, and climbing stairs. The dataset used for this project included sensor signals from accelerometers and gyroscopes embedded in smartphones, capturing three-axis movement patterns. These raw signals were preprocessed to extract meaningful statistical and signal-based features, such as mean, standard deviation, and signal magnitude area, which were then used as inputs for model training.

Several machine learning models were explored to identify the most effective approach. Logistic Regression was initially implemented as a baseline due to its simplicity and ease of interpretation. K-Nearest Neighbors (KNN) was tested for its intuitive distance-based classification, while Support Vector Classifier (SVC) was included for its ability to handle complex, high-dimensional data using kernel methods. More advanced models like Decision Trees, Random Forests, Gradient Boosting, and XGBoost were also applied due to their strong performance in structured classification tasks. Each model was trained and evaluated using standard classification metrics, including accuracy, precision, recall, F1-score, and confusion matrices. ROC curves and AUC scores were additionally used to assess the ability of each model to distinguish between activity classes.

Among all models, the Decision Tree Classifier achieved perfect prediction on the test set with an accuracy of 100%. While this suggests an excellent fit with the data, it also raises the possibility of overfitting, where the model may perform well on known data but poorly on new inputs. In contrast, Random Forest and XGBoost not only provided near-perfect accuracy but also demonstrated excellent generalization ability. These models maintained high AUC scores, strong recall (ensuring most activities were correctly identified), and high precision (minimizing incorrect classifications). Their support for feature importance analysis also made it possible to understand which motion signals had the greatest influence on activity classification. KNN and SVC models also performed reasonably well but were more sensitive to feature scaling and data distribution.

6.2 Limitations

While the Human Activity Recognition (HAR) system using smartphones and machine learning demonstrated high accuracy and strong potential for real-time activity monitoring, several limitations must be acknowledged. One major challenge is the limited size and diversity of the dataset used for training and evaluation. The dataset may not fully represent the variety of users, smartphone placements, body types, or motion styles found in real-world scenarios, potentially reducing the model's ability to generalize effectively across different populations and environments.

Another key limitation is the dependency on preprocessed and well-structured sensor data. In practical applications, noise, inconsistent sampling rates, or missing sensor readings may occur, which can negatively impact prediction reliability. The current system also assumes that the phone is worn or held in a fixed position, whereas variations in device placement (e.g., hand, pocket, backpack) can significantly alter motion patterns and degrade accuracy.

There is also a risk of overfitting, particularly with models like Decision Trees, which may achieve perfect accuracy on the test set by memorizing patterns instead of generalizing them. While ensemble models like Random Forest and XGBoost showed better robustness, models such as KNN can be computationally expensive at prediction time and are sensitive to feature scaling and irrelevant data, making them less suitable for large-scale or real-time applications without further optimization.

From a deployment perspective, the system currently lacks real-time data integration, edge computing support, and adaptive learning capabilities that could allow it to evolve with user behavior. It also does not include features such as user authentication, activity transition detection, or anomaly alerts (e.g., fall detection), which are essential for health monitoring and safety-focused applications. Additionally, while the system classifies activities, it doesn't provide interpretability tools to explain model decisions—limiting transparency and user trust, especially in sensitive applications like elderly care or rehabilitation.

These limitations suggest that while HAR using smartphones is feasible and effective, further work is needed to enhance its adaptability, scalability, and real-world readiness.

6.3 Future Work Suggestions

To enhance the capabilities, accuracy, and real-world applicability of the Human Activity Recognition (HAR) system using smartphones and machine learning, several key areas for future development have been identified. One of the most impactful improvements would be to incorporate adaptive sensor data collection that works across diverse device placements and user behaviors. By integrating sensor fusion techniques and wearable IoT devices (e.g., smartwatches or fitness bands) alongside smartphones, the system could gather more comprehensive and consistent motion data, improving classification accuracy across varied real-life conditions.

Another major enhancement would be the inclusion of real-time activity monitoring and alert mechanisms. For applications like elderly care, fitness tracking, or fall detection, the ability to detect anomalies and send instant notifications is essential. This would require low-latency edge processing and real-time database integration (e.g., using SQLite, Firebase, or MQTT) to store and react to incoming activity data on the fly.

In terms of algorithmic improvements, future versions of the system could adopt deep learning models such as LSTM (Long Short-Term Memory) or CNN (Convolutional Neural Networks), which are well-suited for time-series sensor data and can capture complex temporal patterns. These models would be particularly beneficial in distinguishing between subtle activity transitions or composite actions (e.g., walking while texting). To support transparency and trust in critical use cases, the system should also integrate explainable AI techniques like SHAP or attention mechanisms to show which parts of the sensor signals influenced the activity classification.

Additionally, a feedback loop can be introduced, where user confirmations or corrections are used to continuously fine-tune the model through active learning. Personalization features could also be developed, allowing the HAR system to adapt to individual movement patterns over time, improving accuracy for each user.

Finally, for broader deployment, enterprise-grade features such as user authentication, secure data handling, multi-user dashboards, and batch analytics can be incorporated. With these advancements, the HAR system could evolve into a robust, intelligent platform suitable for healthcare, sports, rehabilitation, and smart environment applications.

REFERENCES

REFERENCES

[1]. Trost S.G., Zheng Y., Wong W.K. Machine learning for activity recognition: Hip versus wrist data. Physiol. Meas. 2014; 35:2183-2189. doi: 10.1088/0967 3334/35/11/2183. [PubMed] [Cross Ref].

[2] Chernbumroong S., Atkins A.S., Yu H. Activity classification using a single wrist-worn accelerometer; Proceedings of the 2011 5th IEEE International Conference on Software, Knowledge Information, Industrial Management and Applications (SKIMA); Benevento, Italy. 8-11 September 2011; pp. 1-6.

[3] Ramos-Garcia R.I., Hoover A.W. A study of temporal action sequencing during consumption of a meal; Proceedings of the ACM International Conference on Bioinformatics, Computational Biology and Biomedical Informatics; Washington, DC, USA. 22-25 September 2013; p. 68.

[4] S., Subba Raju V., Misra A., Balan R., Lee Y. The case for smartwatch-based diet monitoring; Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops); St. Louis, MO, USA. 23-27 March 2015; pp. 585-590.

[5] Parate A., Chiu M.C., Chado Witz C., Ganesan D., Kalogerakis E. Risq: Recognizing smoking gestures with inertial sensors on a wristband; Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services; Bretton Woods, NH, USA. 16- 19 June 2014; pp. 149-161.

[6] Scholl P.M., Van Laerhoven K. A Feasibility Study of Wrist-Worn Accelerometer Based Detection of Smoking Habits; Proceedings of the 2012 Sixth IEEE International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS); Palermo, Italy. 4-6 July 2012; pp. 886-891.

[7] Da Silva F.G., Galeazzo E. Accelerometer based intelligent system for human movement recognition; Proceedings of the 2013 5th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI); Bari, Italy. 13-14 June 2013; pp. 20-24.

[8] Shoaib M., Bosch S., Incel O.D., Scholten H.Havinga P.J. Fusion of smartphone motion sensors for physical activity recognition. Sensors 2014; 14:10146 10176. doi: 10.3390/s140610146. [PMCfree article] [PubMed] [Cross Ref].

[9] Varkey J.P., Pompili D., Walls T.A. Human motion recognition using a wireless sensor-based wearable system. Pers. Ubiquitous Comput. 2012; 16:897-910. doi: 10.1007/s00779-011-0455-4.

[10]Apple.UIAccelerationClassReference. https://developer.apple.com/library/ios/documentation/uikit/reference/UIAcceleration_Class/Reference/UIAccel eration.h tml, 2014. [Online; accessed 17-March-2014].

[11] Shoaib M. Human activity recognition using heterogeneous sensors; Proceedings of the Adjunct Proceedings of the 2013 ACM Conference on Ubiquitous Computing; Zurich, Switzerland. 8-12 September 2013.

[12] W.-Y. Deng, Q.-H. Zheng, and Z.-M. Wang. Cross-person activity recognition using reduced kernel extreme learning machine. Neural Network.

[13] Labrador; Miguel A. Human Activity Recognition: Using Wearable Sensors and Smartphones

[14] Reyes Ortiz, Jorge Luis. Smartphone-Based Human Activity Recognition

[15] Sandeep Kumar Polu. "Security Enhancement for Data Objects in Cloud Computing" International Journal for Innovative Research in Science & Technology Volume 5 Issue 6 2018 Page 18-21

[16] Swadesh Chaulya G. M. Prasad. Sensing and Monitoring Technologies for Mines and Hazardous Areas: Monitoring and Prediction Technologies.

[17]. https://doi.org/10.3389/frobt.2015.00028. "A Review of Human Activity Recognition Methods" Michalis Vrigkas1, Christophoros Nikou1* and Ioannis A. Kakadiaris2, F}[ront.

Robot. Reenacted knowledge, 16 November 2015.

[18]. Imen Jegham, AnouarBen Khalifa, Ihsan Alouani, Mohamed Ali Mahjoub, "Vision based human movement affirmation: A diagram and certifiable challenges", Forensic Science International: Digital Investigation, Volume 32, March 2020, 200901

[19]. Profound Learning Models for Human Activity Recognition by Jason Brownlee on September 26, 2018, in Deep Learning for Time Series, Last Updated on August 5, 2019.

[20]. Zehua Sun, Qiuhong Ke, Hossein Rahmani, Mohammed Bennamoun, Gang Wang, Jun Liu, "Human Action Recognition from Various Data Modalities: A Review", Submitted on 22 Dec 2020 (v1), last refreshed 23 Jul 2021 (this version, v4).

[21]. Muhammad Attique Khan, Kashif Javed, Sajid Ali Khan, Tanzila Saba, Usman Habib, Junaid Ali Khan, and Aaqif Afzaal Abbasi, "Human activity acknowledgment utilizing a combination of multiview and profound elements: an application to video observation", Published: 14 March 2020.

[22]. Hong-Bo Zhang, Yi-Xiang Zhang, Bineng Zhong, Qing Lei, Lijie Yang, Ji-Xiang Du, and Duan-Sheng Chen, "A Comprehensive Survey of Vision-Based Human Action Recognition Methods", Received: 2 February 2019; Accepted: 25 February 2019; Published: 27 February 2019.

[23]. Muhammad Attique Khan1, Majed Alhaisoni2, Ammar Armghan3, Fayadh Alenezi3, Usman Tariq4, Yunyoung Nam5, *, Tallha Akram6, "Video Analytics Framework for Human Action Recognition", Issue distributed 06 May 2021.

[24]. https://doi.org/10.18280/ts.370105 "Human Action Recognition in Video Sequences Using Deep Belief Networks" by Mehrez Abdellaoui, Ali Douik. Distributed: 29 February 2020

[25]. Human Action Recognition using Detectron2 and LSTM, Bibin Sebastian, JULY 26, 2021. 113