Human Activity Recognition Using Machine Learning

Article · June 2023

CITATIONS
0

READS
25

4 authors, including:

Ömer Şükrü Akbulut
Dokuz Eylul University
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Derya Birant
Dokuz Eylul University
134 PUBLICATIONS **2,867** CITATIONS
SEE PROFILE

DOKUZ EYLÜL UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

HUMAN ACTIVITY RECONGITION USING MACHINE LEARNING

by
Özcan ELMACI
Ömer Şükrü AKBULUT
Umut İNANÇ

Advisor
Prof. Dr. Derya BİRANT

June, 2023
İZMİR

HUMAN ACTIVITY RECONGITION USING MACHINE LEARNING

A Thesis Submitted to the

Dokuz Eylül University, Department of Computer Engineering

In Partial Fulfillment of the Requirements for the Degree of B.Sc.

by

Özcan ELMACI

Ömer Şükrü AKBULUT

Umut İNANÇ

Advisor

Prof. Dr. Derya BİRANT

June, 2023

İZMİR

ACKNOWLEDGEMENTS

We want to thank Prof. Dr. Derya BİRANT for her continuous support, tolerance, patience, guidance, morale, and motivation from the beginning to the end of the project.

Umut İNANÇ
Özcan ELMACI
Ömer Şükrü AKBULUT

HUMAN ACTIVITY RECONGITION USING MACHINE LEARNING

ABSTRACT

Human Activity Recognition (HAR) is an interdisciplinary research field that focuses on developing techniques and algorithms to automatically identify and understand human activities from sensor data. With the proliferation of wearable devices, smart environments, and the Internet of Things (IoT), HAR has gained significant attention due to its potential applications in various domains, such as healthcare monitoring, sports analysis, security systems, and human-computer interaction.

In order to increase the precision and effectiveness of activity identification systems, this thesis gives a thorough examination into the subject. In order to achieve reliable and real-time activity recognition, the research focuses on machine learning algorithms using accelerometer sensor data from a smartphone.

The thesis explores the uses of HAR in practical contexts and their potential ramifications. It emphasizes the value of HAR in strengthening human-computer interaction, fostering individualized healthcare, and promoting human wellbeing. The integration of deep learning methods, the creation of adaptive recognition systems, and the investigation of HAR in dynamic and complex situations are all identified as future research objectives.

Overall, this thesis advances human activity recognition by offering a thorough study of current methods, suggesting a safe Android mobile application, and resolving significant research issues. The conclusions and recommendations made in this thesis have the potential to lead to additional improvements in HAR methods and to the growth of intelligent systems that are capable of precisely comprehending and interpreting human activity.

HUMAN ACTIVITY RECONGITION USING MACHINE LEARNING

ÖZET

İnsan Aktivite Tanıma (İAT), sensör verilerinden insan faaliyetlerini otomatik olarak tanımlamak ve anlamak için teknikler ve algoritmalar geliştirmeye odaklanan disiplinler arası bir araştırma alanıdır. Giyilebilir cihazların, akıllı ortamların ve Nesnelerin İnterneti'nin (IoT) yaygınlaşmasıyla birlikte, İAT, sağlık izleme, spor analizi, güvenlik sistemleri ve insan-bilgisayar etkileşimi gibi çeşitli alanlarda potansiyel uygulamaları nedeniyle önemli bir ilgi görmektedir.

Bu tez, aktivite tanıma sistemlerinin hassasiyetini ve etkinliğini artırmak amacıyla konuya kapsamlı bir inceleme sunmaktadır. Güvenilir ve gerçek zamanlı aktivite tanıma için araştırma, akıllı telefonlardan elde edilen ivme ölçer sensörünün verilerini kullanarak makine öğrenmesi algoritmalarına odaklanmaktadır.

Tez, İAT'nin pratik bağlamlardaki kullanımlarını ve potansiyel etkilerini araştırmaktadır. İAT'nin insan-bilgisayar etkileşimini güçlendirmedeki, bireyselleştirilmiş sağlık hizmetini teşvik etmedeki ve insan refahını desteklemedeki önemini vurgulamaktadır. Derin öğrenme yöntemlerinin entegrasyonu, uyarlamalı tanıma sistemlerinin oluşturulması ve İAT'nin dinamik ve karmaşık durumlardaki araştırılması, gelecekteki araştırma hedefleri olarak belirlenmektedir.

Genel olarak, bu tez, mevcut yöntemlerin kapsamlı bir çalışmasını sunarak insan aktivite tanıma alanını ilerletmektedir. Ayrıca, güvenli bir Android mobil uygulama önerisi sunarak önemli araştırma konularını çözmektedir. Bu tezde ortaya konan sonuçlar ve öneriler, İAT yöntemlerinde ek iyileştirmelere ve insan faaliyetlerini doğru bir şekilde anlayabilen ve yorumlayabilen akıllı sistemlerin gelişimine yol açabilecek potansiyele sahiptir.

TABLE OF CONTENTS

Page

TABLE OF FIGURES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background Information

Recognition of human activity is one of the active research areas in machine learning for various contexts such as safety surveillance, healthcare, and human-machine interaction. And we know that humans have several activities, and they act differently. ‘Standing’, ‘sitting’, ‘laying’, ‘walking’, ‘walking downstairs’, ‘walking upstairs’ might be given as example.

All mentioned human activities have different graphics and values from each other when they are measured by using accelerometer. Recognizing these activities of humans benefits in some ways. To talk about what an accelerometer is, an accelerometer sensor is a tool that measures the acceleration of any body or object in its instantaneous rest frame. It is not coordinate acceleration. Accelerometer sensors are used in many ways, such as in many electronic devices, smartphones, and wearable devices, etc. So, the accelerometers in smartphones will be used for this project.

1.2 Problem Definition

The lack of application that recognizes the human activities according to dataset which is given externally is a problem.

Another problem is not being an application like API for adapting the other systems and being used for different purposes. Human Activity Recognition project benefits for different situations. For example, older people could have neural illness and they may experience loss of balance and fall. Following soldier activities and suggesting people to do exercise could be also given as example.

1.3 Motivation/Related Works

1.3.1 Motivation

Life is getting a little monotonous and the monotony of life drives people to laziness. So, people do not do enough exercise and it causes health disorders. We want tosuggest people do some exercise for a healthier life.

As mentioned above in the “Problem Definition” part, we want to follow and recognize the activities of older people for their safety. We want to benefit the Military by following the activity of soldiers. According to all these good, motivated ideas, our motivation is to create a platform that achieves them.

1.3.2 Related Works

The recognition of human activity is mainly used in health systems installed in theresidential environment, hospitals, and rehabilitation centers. It is widely used to monitorthe activities of the elderly staying in rehabilitation centers for chronic disease management and disease prevention.

1.4 Goal/Contribution

1.4.1 Goal

There are many applications that could take advantage of the detection and categorization of what the user of the application is doing physically, at any given moment. An example of such an application would be a medical one, which would be to monitor the elderly in case there is a fall and alert the caretakers at a nursing home. Another example would be military application.

One could monitor what the trainees are doing, or even use it on soldiers in combat. The goal of the project is to develop an interface in Python from which such applications can be developed, which internally detects and categorizes the physical events the user has done from a smart phone using machine learning. The internal machine learning model should be externally trainable. It

is also intended to build an application which uses the interface for visual purposes.

1.4.2 Contribution

In this study, physical activities of human individuals will be detected and categorized using machine learning models. Due to the externally trainable nature of the interface the project encompasses, multiple different machine learning models will be tested for better accuracy, with different data sets.

1.5 Project Scope

The study involves the training of machine learning models using human activitydata gathered from gyroscopes of smart phones. This data will be taken from already existing sources and used to train the machine learning models. The same data will be used to train multiple different machine learning algorithms, and they will be compared for accuracy, storing the results.

Training data will consist of accelerometer measures for all directions and will also contain the respective activity definition.

Interface will live inside a smart phone application. The interface will encapsulate the machine learning model for which the real time gyroscope accelerometer data will be fed. The interface will then invoke callbacks or events to return the output of the model to the main application which will then store and visualize the collected data for the purpose of testing the accuracy of different machine learning algorithms later. Interface will have the ability to be configured externally.

1.6 Methodology/Tools/Libraries

1.6.1 Methodology:

The data will be read via electronic devices such as mobile phones. This data willbe processed and converted into a data frame by using the pandas library.

The reason for this is to use data more conveniently and comfortably in machine learning when libraries are kept as data frames in pandas. Before the data has been read and loaded to a data framedata is kept in csv. format because it is one of the best ways to load it to data frames.

After data is gathered and saved in csv. format then it will be loaded to the dataframe where pandas library is used. After this operation data will get preprocessed using scikit learn and target column and descriptive columns will be chosen. In this scenario our Target Column will be 'Activity' and the other data collected from sensors will be our descriptive features. After collecting the data if needed the data will also be normalized.

When the data is prepared this data will be separated with a 90-10, 80-20 or 70- 30 percentages into separate train-test dataframes. Both train and test dataframes will be used in different machine learning algorithms to train a model using skicit library whicherror scores will also be evaluated. These scores will be compared and after this operationthe best machine learning model will be chosen.

After these operations have been completed the decision of the best algorithm will be made and a user Interface will be written in Python using Tkinter library.

1.6.2 Tools:

VSCode

1.6.3 Libraries:

1. Machine Learning Library: Sckit learn, sklearn

   This library is one of the best Machine learning libraries and oneof the easiest to use library, so our selection is this way.

2. Data frame operations: Pandas dataframe

   Pandas library is the best library to manipulate data using python.

3. Visualization: Matplotlib

   Matplotlib has lots of visualization techniques which will be veryhelpful while comparing model scores.

4. GUI: Tkinter

5. Other Libraries: Numpy

On the next page and the other page, Figure 1. 1 represents the flowchart which includes python code working algorithm. And, Figure 1. 2 represents the flowchart which includes our path after developing a working python code.

1.7 Flowcharts

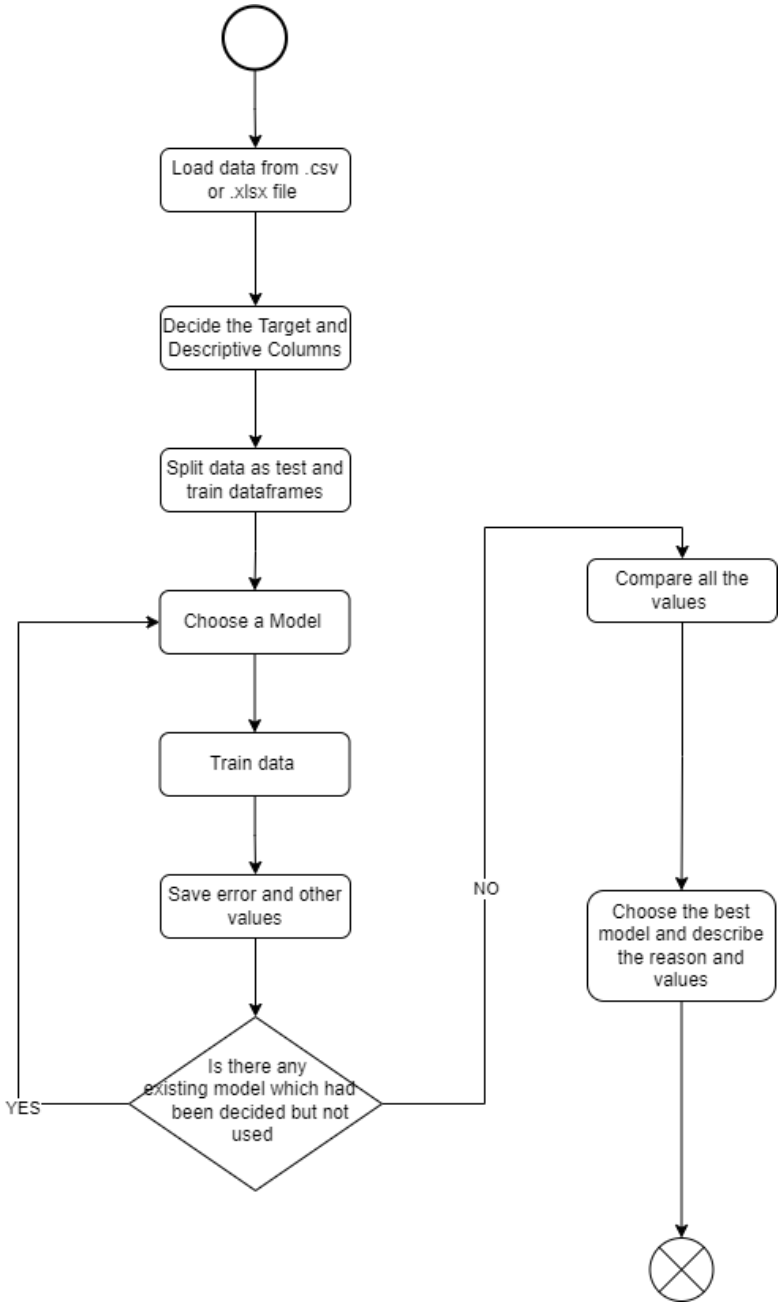

Figure 1. 1 Model comparison

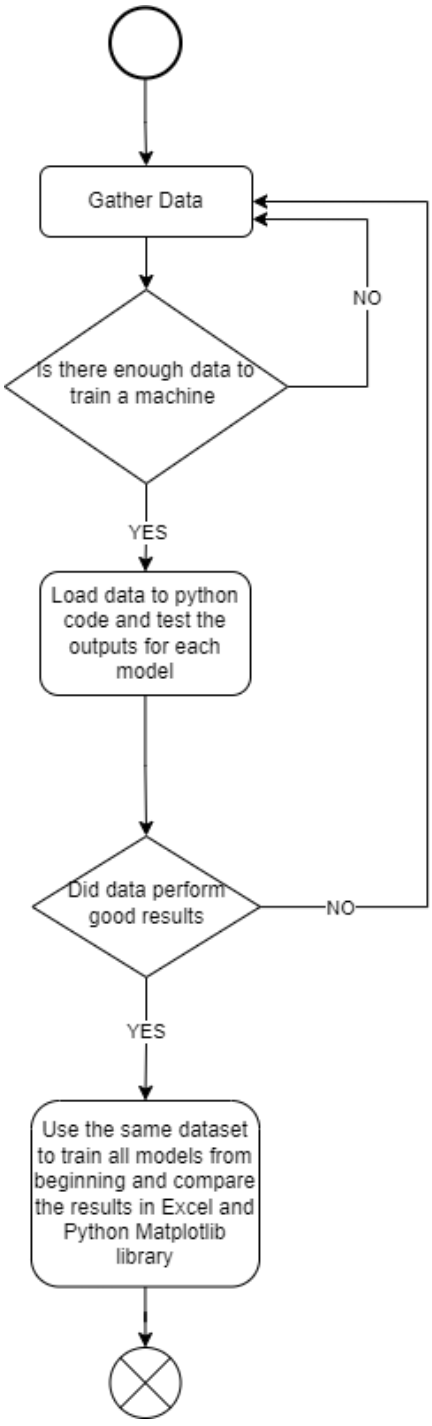

Figure 1. 2 Training model steps

CHAPTER TWO

LITERATURE REVIEW

This section presents research of literature, references used in this work and recentstudies based on a group of measurements.

This section consists of 2 subheadings: “Previous Studies” and “Technologies, Algorithms, and Tools”. In “Previous Studies” part, the previous studies that are related to our thesis are mentioned, in “Technologies, Algorithms, and Tools” part, the technologies, algorithms, and tools that we decide to use in our project are mentioned andexplained. How we have decided to use these technologies, algorithms, and tools have been explained by using the related studies below.

2.1 Previous Studies

According to Tran et al. (2016), they designed and constructed a system to identify human actions using integrated sensors in smartphones. There are six actions that are selected for recognition include: walking, standing, sitting, lying down, up the stairs, down the stairs. In this system, Support Vector Machine (SVM) is used to classify and identify action. Collected data from sensors are analyzed for the classification model - the model file. The classification models are optimized to bring the best results for the identified human activity. After forming the classify model, the model will be integrated into the system to identify the human activities. Human activities recognition system is written on Windows and Android platforms and operates in real time. The accuracy of the system depends on selected features and the quality of the training model. On the Android system running on smartphone with 248 features achieve 89.59% accurate rate. But the system still has some certain restrictions. The percentage of recognition is low insome actions. So, we have

decided to do more research to improve the performance and increase the detection capabilities of the system.

Providing accurate and opportune information on people's activities and behaviors is one of the most important tasks in pervasive computing. Innumerable applications can be visualized, for instance, in medical, security, entertainment, and tactical scenarios. Despite human activity recognition (HAR) being an active field for more than a decade, there are still key aspects that, if addressed, would constitute a significant turn in the way people interact with mobile devices. Lara et al. (2012) based on wearable sensors. A general architecture is first presented along with a description of the main components of any HAR system. They also propose a two-level taxonomy in accordance with the learning approach (either supervised or semi-supervised) and the response time (either offline or online). Then, the principal issues and challenges are discussed, as well as the main solutions to each one of them. Twenty-eight systems are qualitatively evaluated in terms of recognition performance, energy consumption, obtrusiveness, and flexibility, among others. Finally, they present some open problems and ideas that, due to their high relevance, should be addressed in future research. This thesis has affected and encouraged us.

Unintentional falls cause high numbers of death, particularly among the elderly and the children. Fall detection is hence an important challenge which is worked on by many researchers. There are two ways of approaching this problem: Visual or ambience-based approach, or wearable devices. According to Wisesa et al. (2019), visual or ambience-based approaches are expensive, despite being less intrusive. So, they have proposed a wearable device-based approach, which is more cost effective. The wearable device has an accelerometer and gyroscope, which are used for the algorithm. The algorithm they have developed makes use of Recurrent Neural Networks, unlike neural networks such as MLP (Multi-Layer Perceptron) or CNN (Convolutional Neural Networks) which accept fixed-size input and produce fixed-size output.

They have trainedthe model with the UMA Fall Dataset. In the end, they have developed a fall detection algorithm based on LSTM (Long Short-Term Memory) variant of RNN, which has promising accuracies for accelerometer and gyroscope accuracies, particularly the X-axis of 92.31% and 74.47% respectively. The result of this algorithm is promising, meaningful data can be obtained from just the accelerometer and the gyroscope, which are both present in most smart phones. It also recommends filtering at the pre-processing stage toreduce noise and improve accuracy.

The main purpose of Hassan et al. (2018) is to develop a robust human activity recognition system based on smartphone sensors' data. It seems very feasible to use smartphones for activity recognition as the smartphone is one of the most used devices by people in their daily life not only for communicating with each other but also for a very wide range of applications, including healthcare. Thus, a novel approach has been proposed here for activity recognition using smartphone inertial sensors such as accelerometers and gyroscope sensors. From the sensor signals, multiple robust features have been extracted followed by KPCA for dimension reduction. Furthermore, the robust features have been combined with deep learning technique, Deep Belief Network (DBN)for activity training and recognition. The proposed method was compared with the traditional multiclass SVM approach where it showed its superiority. The system has beenchecked for twelve different physical activities where it has obtained a mean recognition rate of 89.61% and an overall accuracy of 95.85%. On the contrary, other traditional approaches could achieve a mean recognition rate of 82.02% and an overall accuracy of 94.12% at best. Besides, it has shown its ability to distinguish between basic transitional and non-transitional activities. In future, we plan to focus on more robust features and learning for more efficient and complex activity's recognition in real-time environments.

In Wan et al. (2020), the main purpose is to compare the advantages and disadvantages of five algorithms, CNN, LSTM, BLSTM, MLP and SVM, in

the recognition of human behavior and they compared them. Through experiments, it can befound that CNN, as a classical neural network algorithm, still has important value in the field of human behavior recognition and is an excellent classification and recognition algorithm. In this paper, five of the algorithms are tested in sequence through two human behavior datasets, and the performance of these models is measured by multiple evaluation indicators. However, there are still some shortcomings in this paper. The structure of the four neural network models used in the experiment can still be further optimized, and more detailed comparison experiments can be conducted. As you see, there are several algorithms and methods for recognizing and modeling human activities.We have also chosen an algorithm that is different from them. We have used Random Forest Algorithm.

Yin et al. (2008) tells that the traditional approach to detecting abnormal activities using sensor data is the high false positive rates, particularly, when abnormalities are rare, which causes bias towards normal data. They also complain about the lack of training data, which they see as one of the solutions to the traditional approach. But it is less of anissue nowadays. They propose an alternative approach which employs a one-class supportvector machine (SVM) that is trained on commonly available normal activities, which filters out the activities that have a very high probability of being normal. They then derive abnormal activity models from a general normal model via a kernel nonlinear regression (KNLR) to reduce the false positive rate in an unsupervised manner. They demonstrate the effectiveness of their approach using real data collected from sensors attached to a human body. The results show that the alternative approach achieves a bettertradeoff between the false alarm rate and detection rate. Another advantage is thealgorithm generates its own abnormal models in its absence. However, this is also seen as a risk in case the abnormal becomes the norm one day. The paper explains the algorithm they developed rigorously, which helps us gain insight on how we might apply classification in our own project in the absence of training data.

2.2 Technologies, Algorithms, and Tools

2.2.1 Scikit Learn

Scikit-learn is a machine learning library built in Python that has grown rapidly in recent years. It allows algorithms in machine learning such as preprocessing, feature selection, pipelining, model assessment, and other variety of other activities.

2.2.2 Machine Learning Algorithm

Machine Learning splits into 3 topics which are:

- Supervised Learning: When the amount of data is less, and it is clearly labeled.
- Unsupervised Learning: This learning type is better to use in conditions where the data is larger, and labeling is harder.
- Reinforcement Learning: This learning type is in the semantic of "Good dog, bad dog". The program decides and someone evaluates as correct prediction or not.

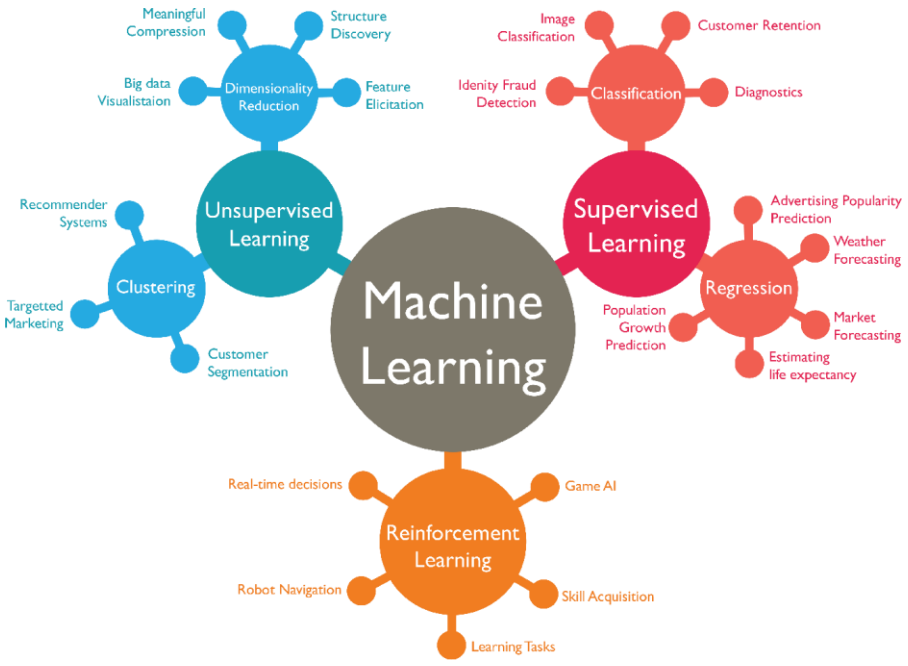

Figure 2. 1 Machine learning algorithms

Machine Learning Algorithms which is useful for this project:

- Logistic Regression
- Random Forest Algorithm
- Regression Algorithm (Boosted Forest, Random Forest, Linear Regression)

After researching about this topic, it has been recognized that in case Mean Absolute Error (MAE) for this project is higher than aimed, because of extreme/exceptional values then Huber Regression may be used for this part. And these Extreme or Exceptional values will be discarded and trained this way. Huber Regression is one of the regression techniques which is robust to outliers.

2.2.3 Accelerometer:

We were inspired by this, and we can write as if we thought that using an accelerometer would be logical and cheap at the same time, and that our work would be even simpler with a smartphone.

CHAPTER THREE

REQUIREMENTS/REQUIREMENT ENGINEERING

3.1 Functional Requirements

3.2.1 Collecting Data

The recognition of human's activities for individual requires a dataset. The users encounter a screen that they can enter necessary information. All information is used to predict the individual's activity. The machine learning model has already been trained by using the dataset that mentioned earlier chapters.

3.2.2 Analyzing Data

Analyzing the data is a critical section in the whole process of training the model and predicting. So, the aim of the project is to take instance data and predict the individual's activity (sitting, standing, walking, running, climbing stairs).

The given HAR (Human Activity Recognition) dataset is analyzed to create a model. A new instance which was taken from users is analyzed. According to the information, the program analyzes data and predicts the result. Then, the result is shownon the screen.

3.2.3 Machine Learning Model Evaluation Metrics

The core of the system is the evaluation process for creating an effective model. The model predicts and gives a result that can be important for the reliability of the model. Different evaluation metrics are used for different problems. Accuracy, Precision, and Recall are used for categorical data. The dataset that is used for this project consists of categorical data. Therefore, after building the machine learning model, these metrics will be used to understand

the model and continue to improve the results.

To explain these 3 concepts ‘Accuracy’, ‘Precision’, and ‘Recall’; Accuracy tells you how many times the ML model was correct overall. Precision is how good the model is at predicting a specific category. Recall tells you how many times the model was able to detect a specific category. So, the created model uses these concepts and predicts the result.

3.1.1 Model Connection

The desktop application and the machine learning model have been built on different platforms. The model has been trained in Azure Platform. Python Programming Language has been used for the project. Tkinter Library has been used for UI Design. Then, communication has been provided between Python Desktop Application and the trained model in Azure Platform. Thus, the system will need an internet connection for the model to work properly.

3.2 Non-Functional Requirements

3.2.4 User-Friendly Interface

The system consists of training models and generating predictions from those models. The users will be expected to see the models, their specification, and accuracies. They will also need to be able to select a model and get a prediction from a data set given by the users and train a model if its needed. A properly made user interface is needed for the users to perform these actions with ease.

Tkinter, a python language library for creating simplistic, lightweight graphical user interfaces will be used to make the interface for the system. A table-like structure and a file browser prompt to see and add models and data sets; radio buttons and normal buttons to configure and train models and generate predictions will be present.

3.2.5 Processing Time

There needs to be a balance between the accuracy of the model and performance. When the complexity of the model increases, it may increase the accuracy, but it will also increase the required processing time. This will cause suboptimal response times for the user. Since the application will calculate and return the results of the trained models on input data after user command, it should have a feedback time of 3 seconds at most.

Since it is a desktop program with model training and prediction capability, the system should at least have decent hardware resources, but most modern computers have more than enough computing power. It must be responsive and should not lag. The prediction results must be calculated on the program and stored so that they may be returned quickly.

3.2.6 Availability

Stored trained models should be available to the user on demand immediately for predictions over the given user data. For training, database or data storage must always be available to save the trained model. The program's usability and performance must not depend on time of day, or dates and be consistent always.

3.2.7 Reliability

Reliability is the application of data analytics, including AI machine learning, to predict when an asset will fail or otherwise deteriorate, so that it can be serviced or replaced before failing. The project should not have any output errors. The software should be error-free and produce consistent results. But we know that there is no software that has any errors. These characteristics and requirements must be met.

When a machine learning model is used, the predictions about human activities should provide optimal results and accurate information for users.

3.2.8 Security

Security has always been a top priority because it is important that data is not stolen and is not used for different purposes.

A UI has been created for the project. For providing security, none of the users allow to access or edit dataset. Just admin can access and edit dataset.

CHAPTER FOUR

DESIGN

4.1 Architectural View

The general architecture of the design is shown in the Figure 4. 1.

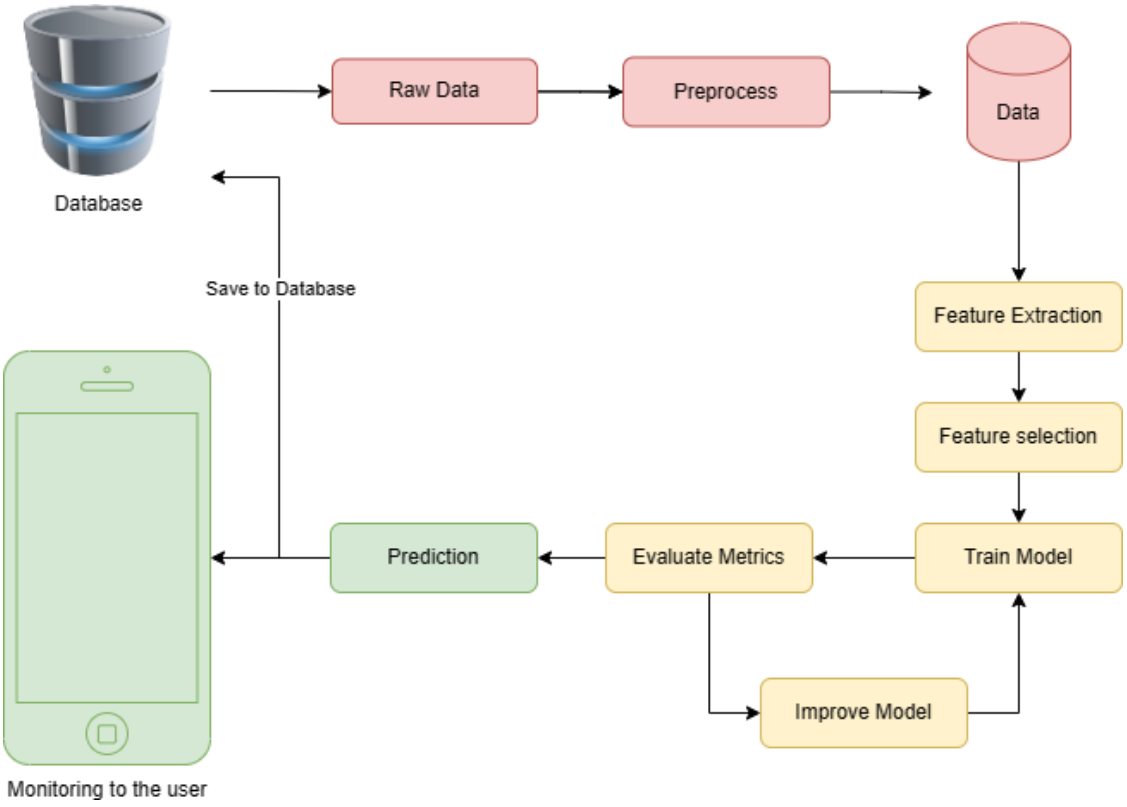

Figure 4. 1 Architectural diagram

4.2 Database Design/ER Diagram

In Figure 4. 2, the ER diagram of the database is shown.

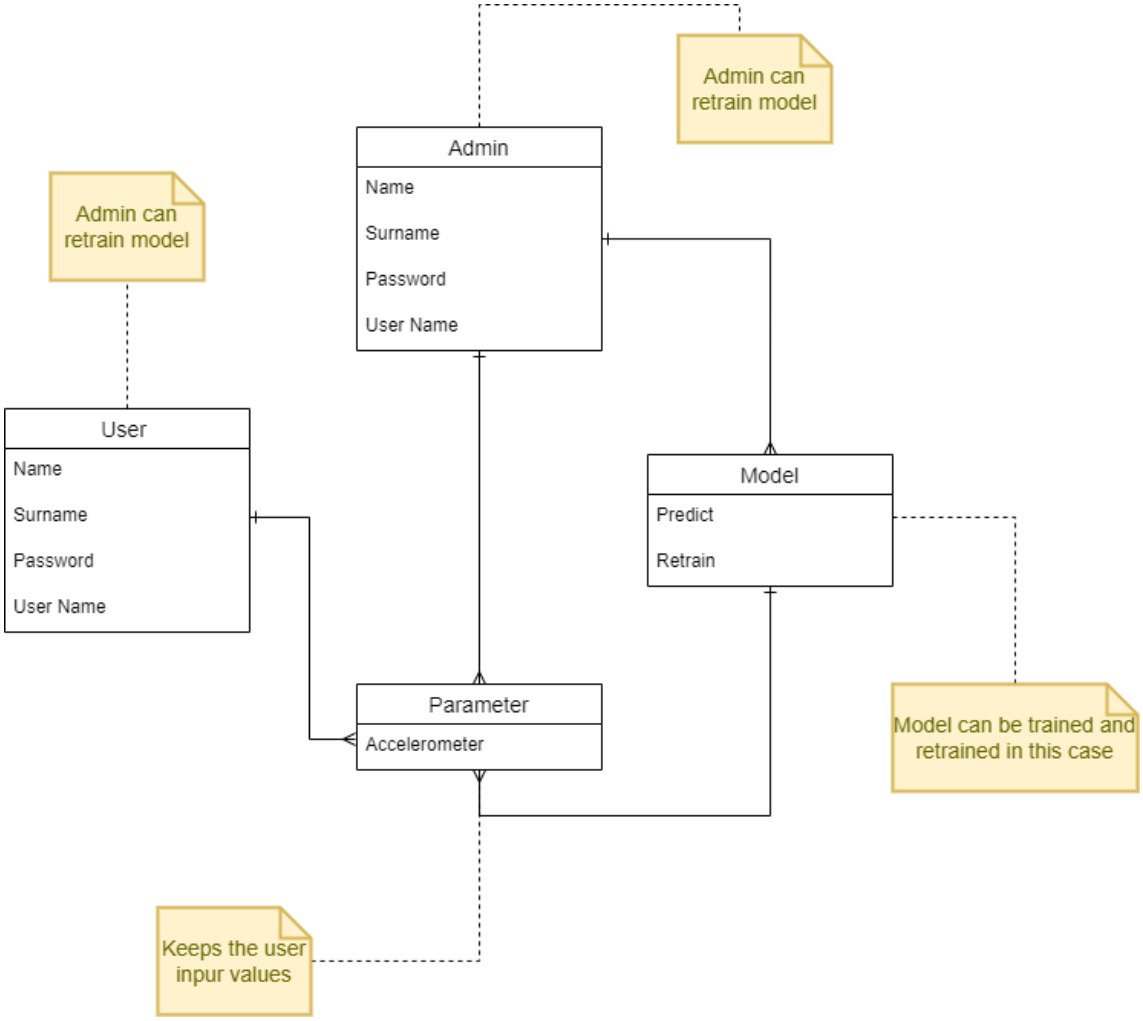

Figure 4. 2 Er diagram

4.3 UML Class Diagram

The UML class diagram is indicated in the Figure 4. 3.

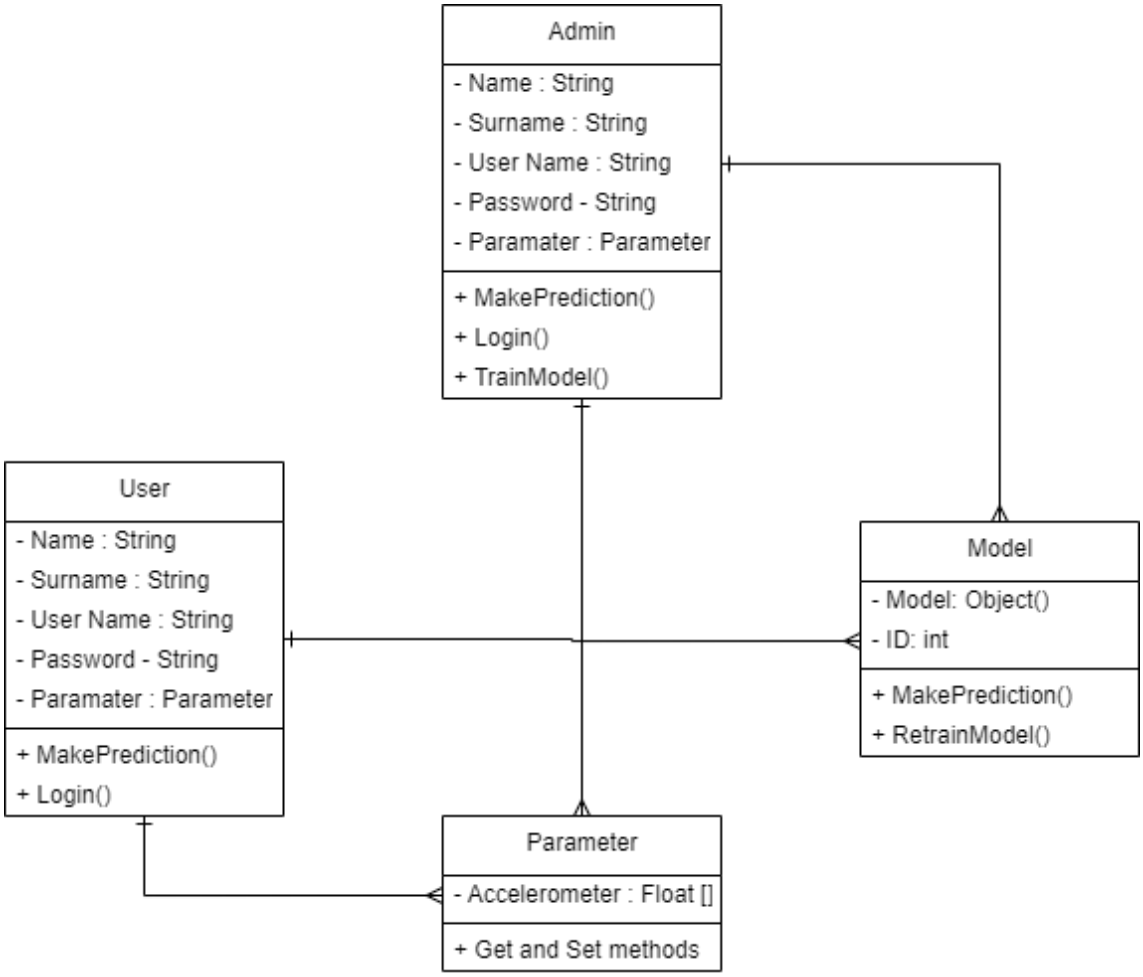

Figure 4. 3 UML class diagram

4.4 UI Design

The UI design of the mobile application is displayed in Figure 4. 4 below. A user can log in and make predictions using the model created by the program.

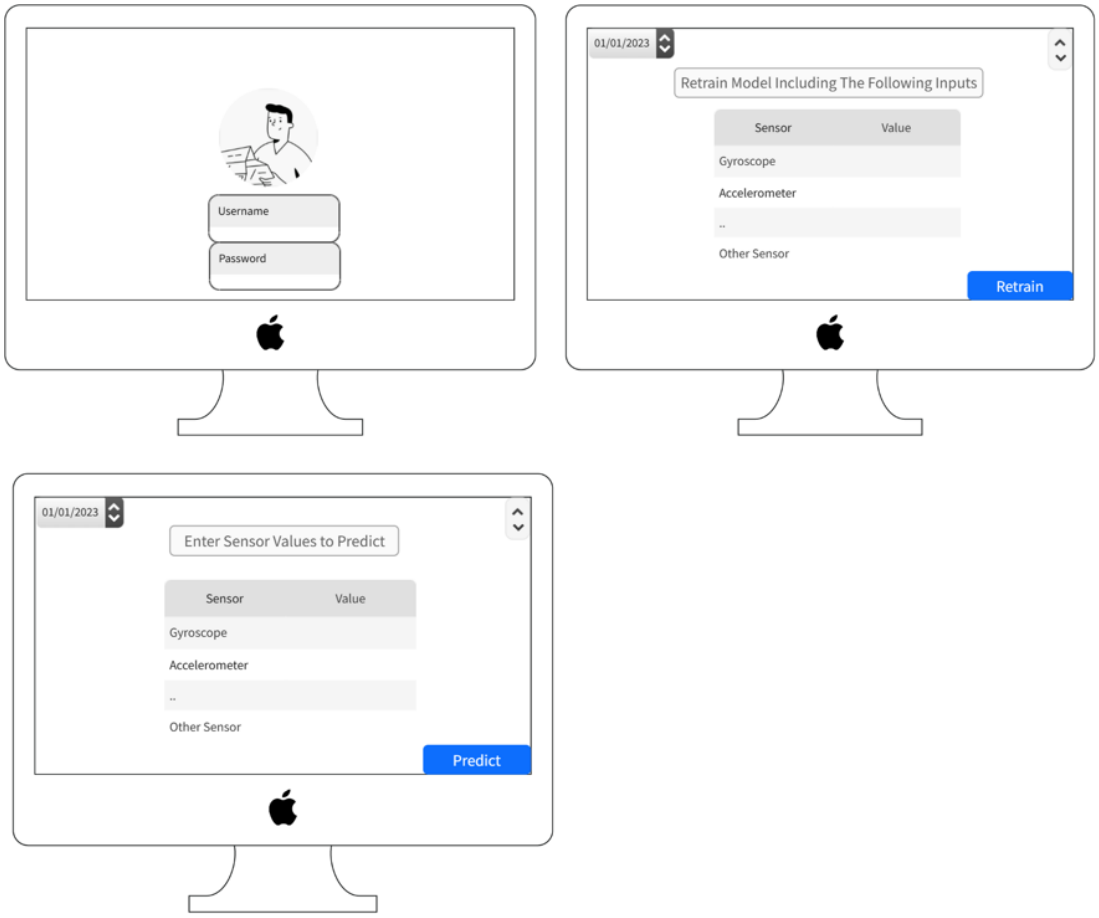

Figure 4. 4 UI design

4.5 Use Cases

The use case diagram is shown in the Figure 4. 5 as follow:

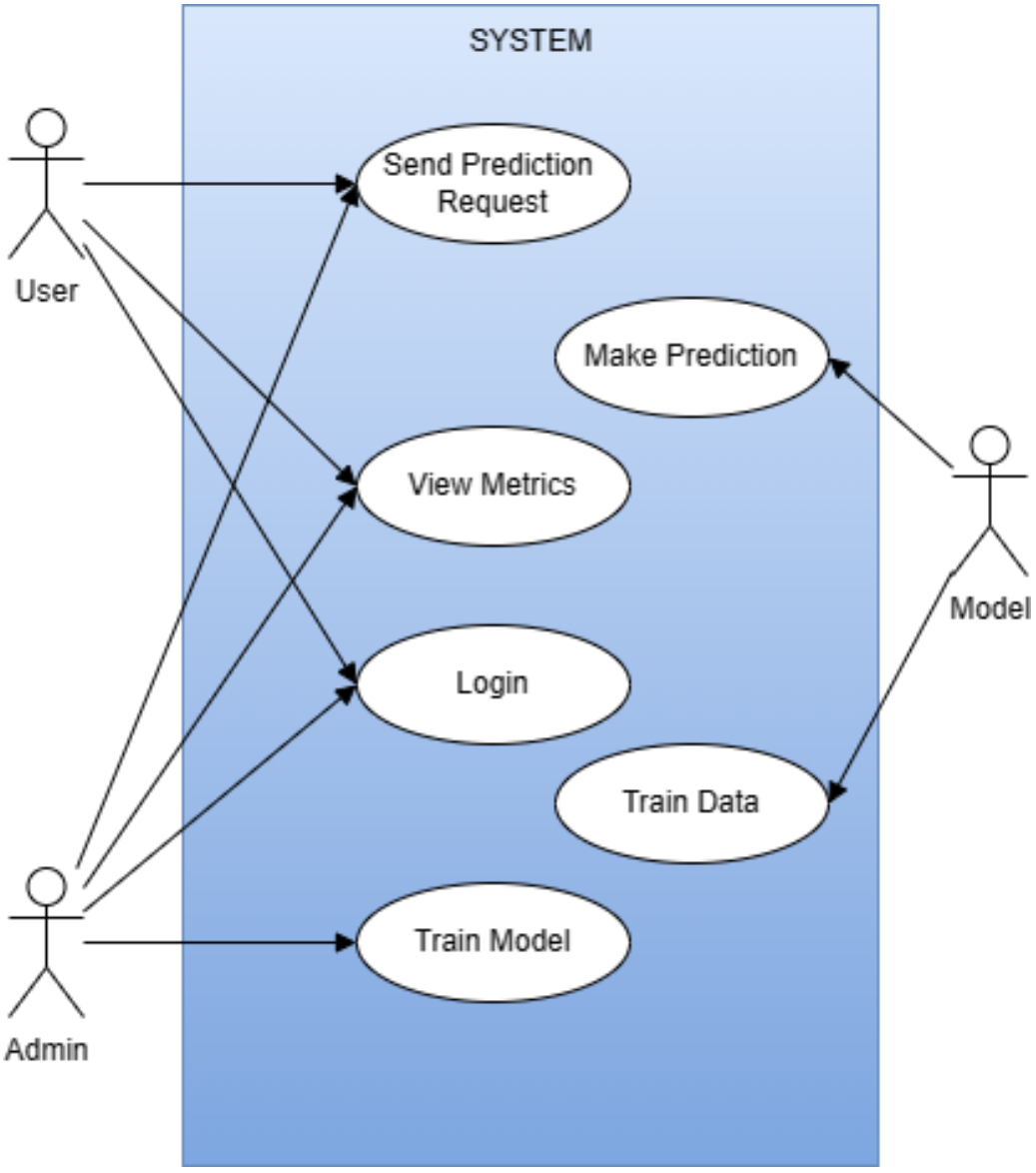

Figure 4. 5 Use case diagram

4.6 Sequence Diagram

The Figure 4. 6 examines the order of the possible user activities in the mobile application.

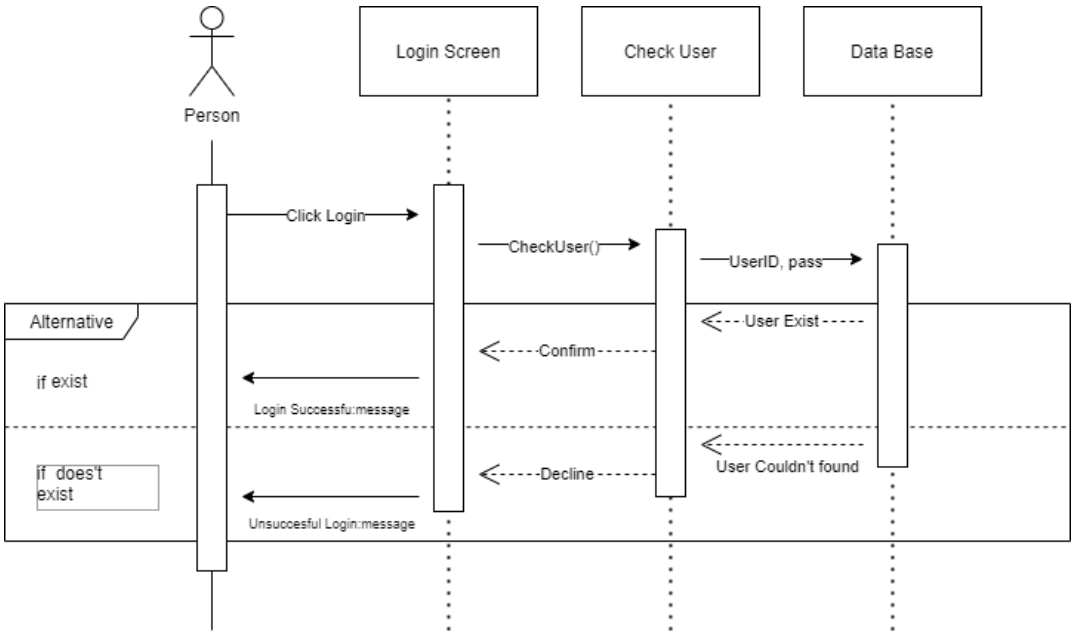

Figure 4. 6 Sequence diagram

4.7 Activity Diagram

The activity diagram as shown in the Figure 4. 7 below indicates decision path from start to end.

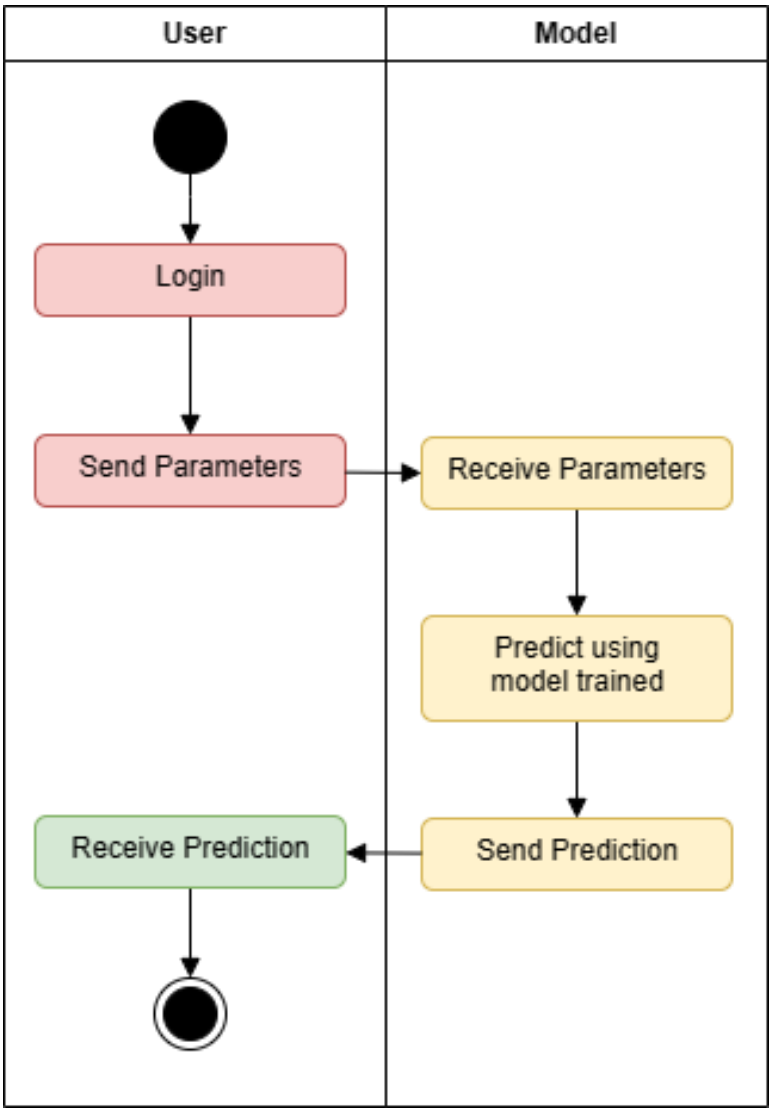

Figure 4. 7 Activity diagram

CHAPTER FIVE

IMPLEMENTATION

This section attempts to provide a quick overview of the development process while highlighting the most important topics.

5.1 Technical Implementation

5.1.1 Frontend Implementation

Project HAR (HUMAN ACTIVITY RECOGNITION) is a mobile application combined by Python. The frontend part of the project has been developed by using Android Studio. Figure 5. 1 shows the Android Studio IDE. The mobile app contains several pages.The pages will be developed day by day. These application pages are demo pages to showthe main schema of the project. These sections will be navigated by using the buttons andsome actions.

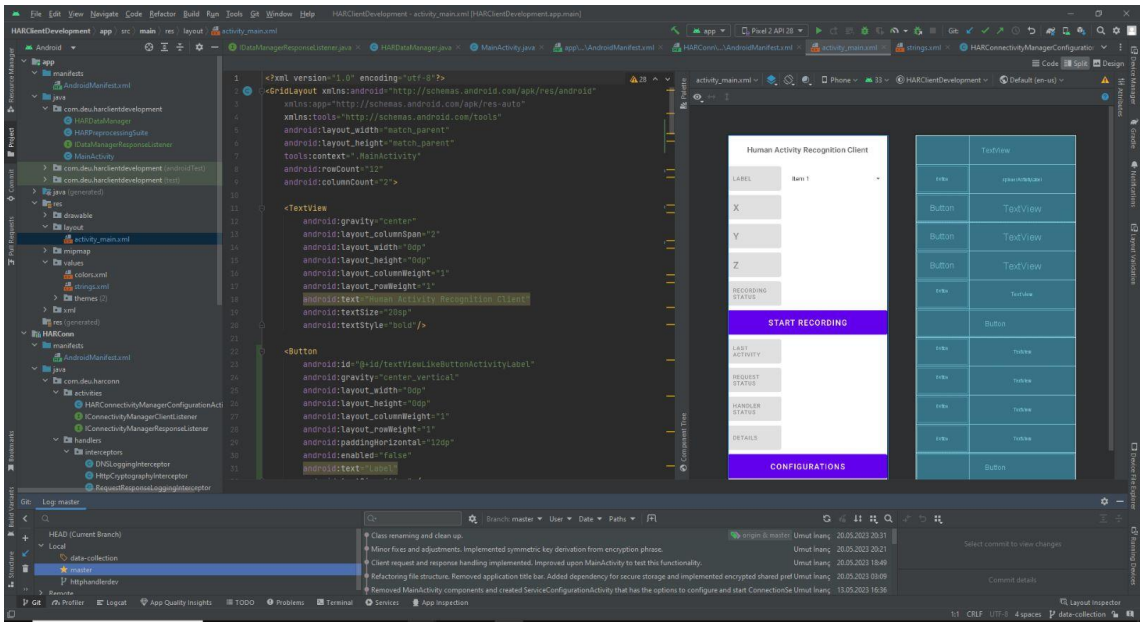

Figure 5. 1 Android studio ide

The added figures of the pages on the mobile application are shown below. Figure 5. 2 shows the first screen of the mobile application. You can see the live values of the accelerometers of three position (x axis, y axis, z axis). There is a condition. You can start recording data from the accelerometer of the mobile phone in case you set the convenient configuration settings from configuration page. Configuration page is shown in Figure 5. 3. The predicted result by the machine learning model is shown “REQUEST STATUS” part in the page. Users can reach the configuration page very easily by using the button. Configurations Button will navigate the user to the page.

Additionally, if you select the label as shown in the Figure 5. 2, your data is added into database and used for retrain the model.

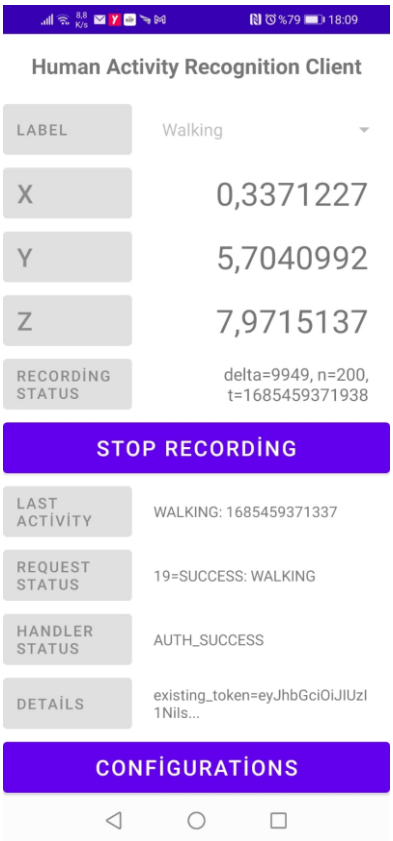

Figure 5. 2 Starting page of the application

The figure of the Configuration Page is shown at Figure 5. 3. On this screen, users should select the method. There are two methods named “HTTP” and “HTTP_WITH_NGROK”. If the user selects “HTTP”, the user should also enter a URL. If the user selects “HTTP_WITH_NGROK”, the user should provide a NGROK API KEY because of the encryption. If provided information is true, user should log in the system with the correct username and password. As mentioned before page of the thesis, you can start recording your data in case you set the convenient configuration settings in Configuration Page.

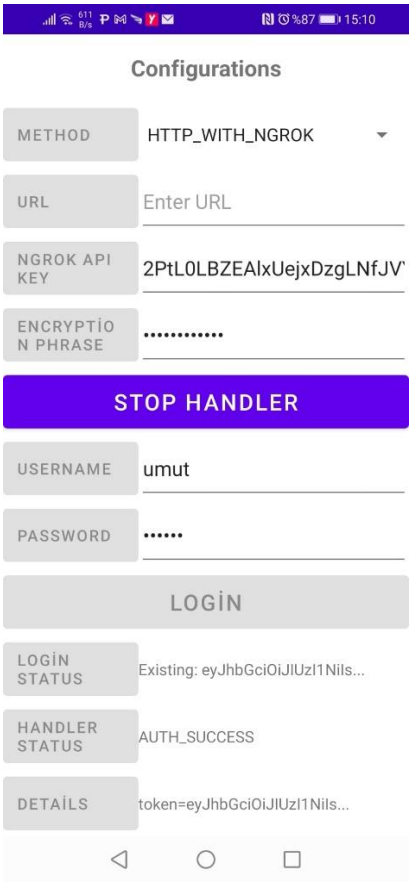

Figure 5. 3 Configuration page

The android version of the Mobile Application supports Android 7 and upper version with 94%. The Virtual Machine Device is shown in Figure 5. 4. The Android version of the device is Android 12, and all development

processes have been tested with this device.

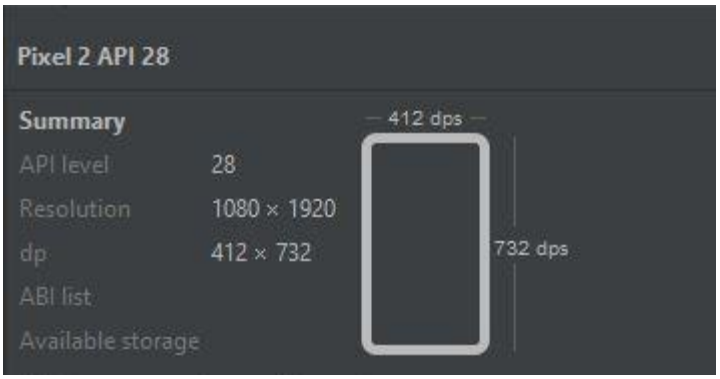

Figure 5. 4 Virtual machine device

5.1.2 Backend Implementation

Backend implementation consists of the networking and database management between the mobile phone application recording HAR-related data and the machine learning application residing on a personal computer. Machine learning application use the HAR-related data to test or train its model and to achieve this, there has to be information flow from the mobile phone and the personal computer. Three key technologies are used for these purposes:

Table 5. 1 Technologies and purpose

Technology	**Purpose**
Flask Microframework	To create and power the server API which will listen to the mobile phone application for data over HTTP.
SQLite	Database which will serve as a data storage and IPC (Inter-process Communication) method between the Flask-powered web server and machine learning application.

Ngrok	A tunneling service used to expose the web server to the wider internet in networks where port forwarding and serving requests are not possible due to ISP-side rules.

Backend architecture can be split into two parts. The first part is the network between Flask Web Server API and mobile phone application and second part is Database connections with Flask Web Server API and machine learning application.

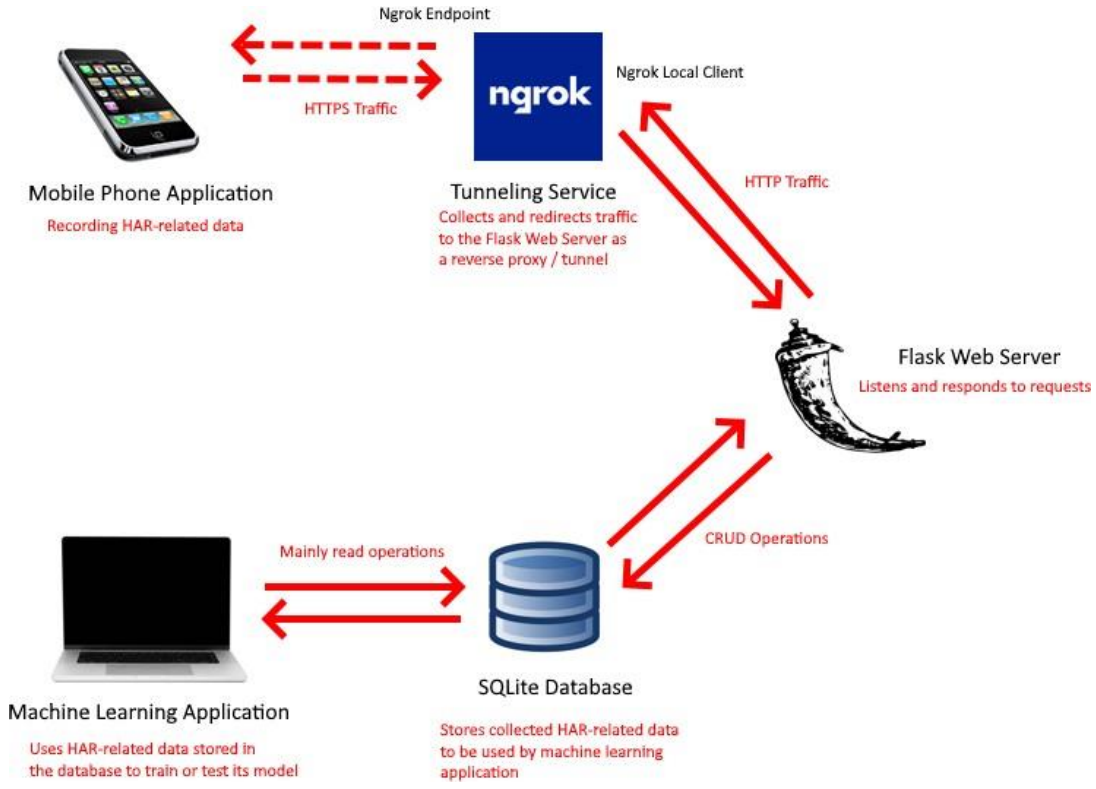

Figure 5. 5 Main architecture

Networking, which is the first part of the architecture, is facilitated over HTTP/HTTPS protocols using Ngrok service and Flask microframework. Because a mobile phone is a physically separate device from a personal

computer, wireless communication is a requirement. During the requirement phase, we have ruled out Bluetooth because not every personal computer supports it, which is a problem for some team members. USB connection was also ruled out because the purpose is to record human activity, which needs room. So, we have decided to facilitate communication over the Internet.

The issue with hosting a web server in this context is since the project needs to be demonstrated in a school environment, we will not be able to set up ports and configure the firewalls within the school network such that LAN is unusable. Exposing the web server to the general internet is necessary, which is also not doable for the same reasons. In order to overcome this, an external service called Ngrok is used.

Ngrok is a tunnelling service which exposes a local area network to the Internet. It does this by acting as a reverse proxy. Ngrok's local client software is started on the personal computer which establishes a tunnel with Ngrok's own servers that acts as the reverse proxy which has a publicly accessible IP that supports HTTP/HTTPS protocols. In the free version of Ngrok's subscriptions, only the communication between user clients and Ngrok Local Client is TLS-secured with HTTPS. Communication within Ngrok's own system (between Ngrok endpoint and Ngrok Local Client) is unknown if its encrypted and messaging on the local loopback happens over HTTP, as such there are points in the system which are open to sniffing. In order to secure it, we intend to encrypt the JSON payload with AES256 with GCM mode using a symmetric key between the mobile phone application and Flask web server.

The second part of the architecture involves two separate applications with different processes, which are the Flask web server and machine learning application. Separating the web server part from machine learning part instead of going for a multithreaded, singular application is a design decision aimed at

increasing decoupling and scalability, which are useful for development. It would also allow machine learning application to avoid using Flask framework. However, with two different processes comes the challenge of IPC (Inter-process Communication). Initially, it was thought that pipes, sockets or memory-maps within python multiprocessing library could be used but it was then decided to instead use a database like SQLite which is also Python friendly. Having a database that can serve multiple readers and writers solves the problem of IPC and provides permanent storage for future requirements. SQLite's performance is deemed to be enough for this project. Another weakness of SQLite is that while it allows multiple reads at the same time, it does not allow multiple writes, but this is not expected to cause a problem because Flask web server will be the writer most of the time.

Implementation of the mobile phones involves having a background service that runs continuously. This service facilitates the communication between the main application and the HTTP client it manages. To do this, the service starts and owns a thread with a messaging loop called HttpHandler, which manages the multitude of states a connection has in a complete state machine fashion. HttpHandler has three main phases: URL_RESOLVE, AUTH_USER, and READY. These phases occur sequentially and if there is an error in a phase, it goes back. HttpHandler can be started in Ngrok mode and non-Ngrok mode, which determine how the target URL is obtained. In the latter mode, HttpHandler keeps sending requests to Ngrok API to obtain the target URL. In the first mode, it asks the user through the service for a URL and assumes the received URL to be correct. After the URL is obtained, phase moves to AUTH_USER, in which HttpHandlerrequests credentials from the user through service unless there is already an unexpired JWT token retrieved by the service. If not, it connects to the Flask server and attempts toget a JWT token to be used in future requests. If this authentication fails, HttpHandler will keep asking for credentials, if it succeeds then HttpHandler will tell the service to store this token and it will use this token on further requests. After AUTH_USER phase,

HttpHandler enters into READY phase in which it can start processing regular user requests. When there is a connection issue, HttpHandler waits for a set time before continuing and trying again.

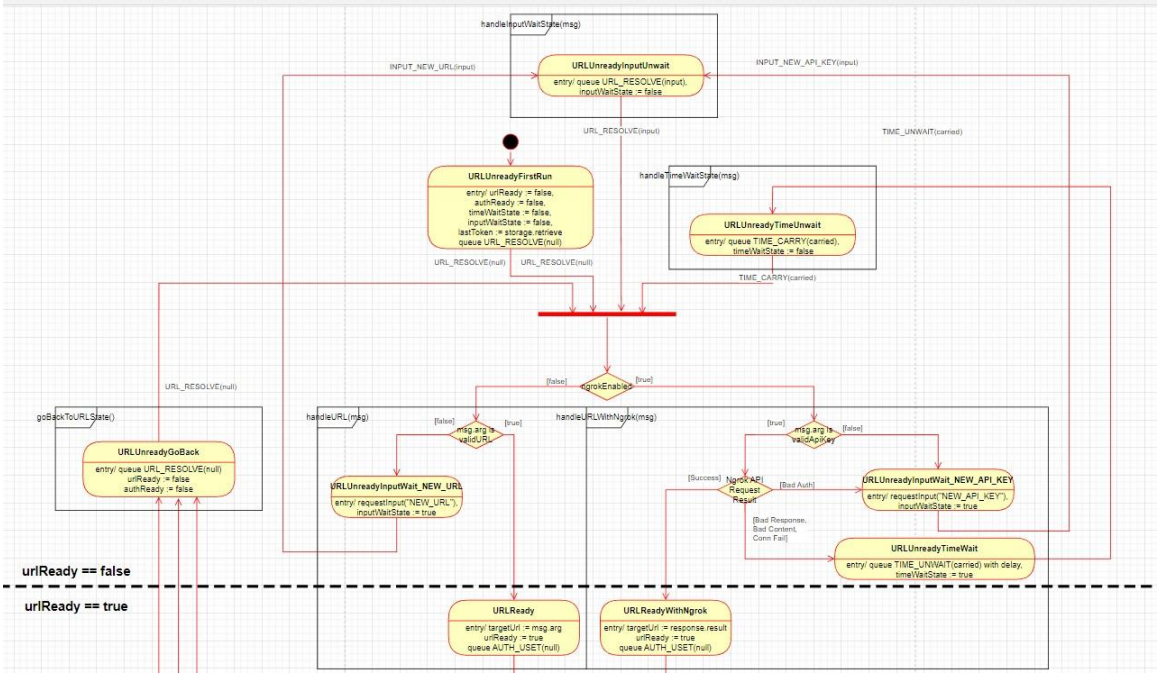

Figure 5. 6 HttpHandler state diagram (1)

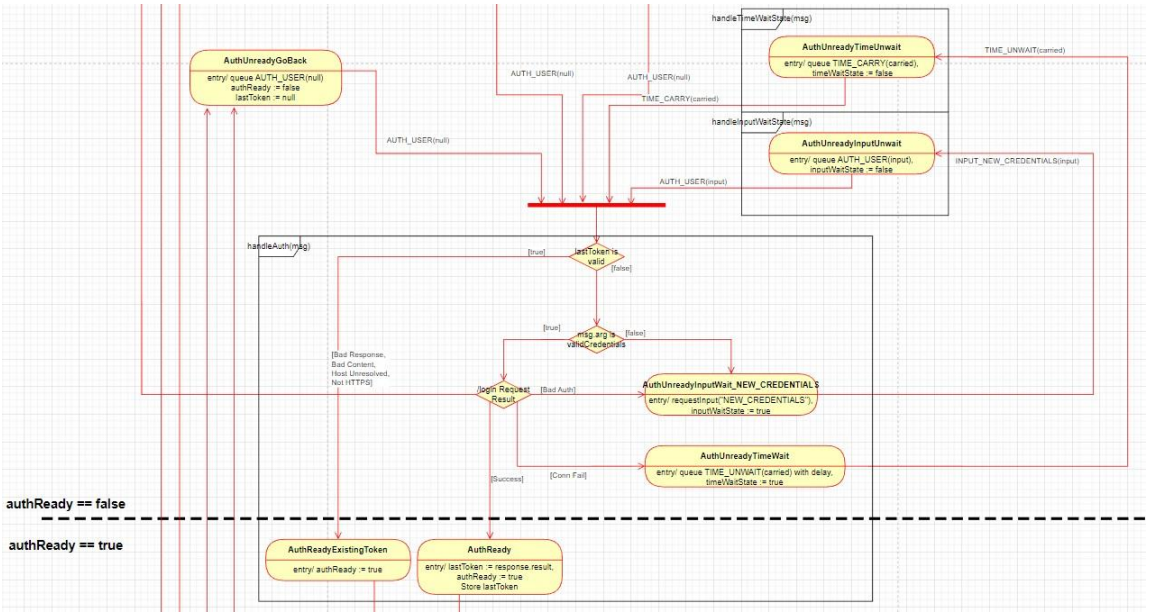

Figure 5. 7 HttpHandler state diagram (2)

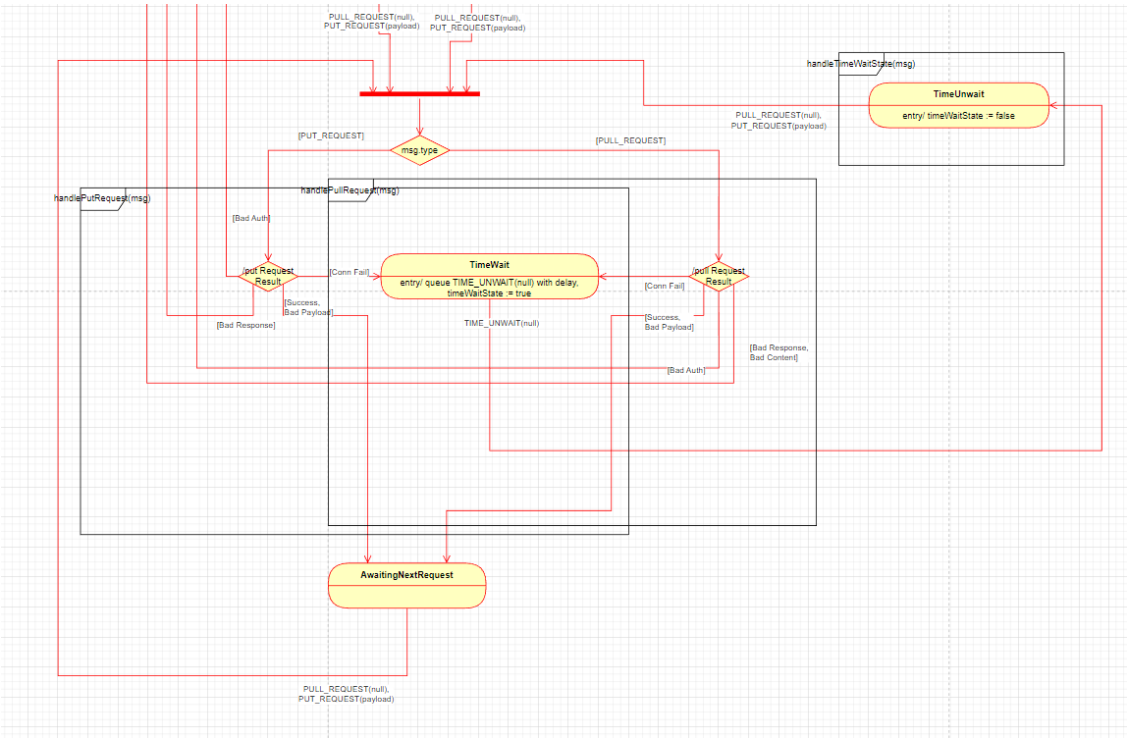

Figure 5. 8 HttpHandler state diagram (3)

HttpHandler implementation makes use of OkHttp library to make the actual requests. OkHttp allows us to set up interceptors which capture requests and responses before it is handled and allows us to modify them. AES256 encryption and decryption takes place here. IV with 12 bytes is selected randomly for the GCM mode and it is appended to the payload during encryption, the same goes for received messages from the Flask server where IV is appended to the end. MAC is set as 16 bytes. The key for the cryptographic operations will be derived from a password phrase configured by the user. Only the payload is encrypted because of viability and complexity, which still achieves enough security because data between the Flask server and HttpHandler is only exchanged on the payload of HTTP message. Only the headers can be sniffed by an attacker which is a non-issue for our case because caching is disabled, headers are discarded, and replays attacks can only occur if attacker knows the secret key.

5.1.3 Machine Learning Implementation

Python will be used for machine learning in this project. It was determined that Logistic Regression gave the best results when the model was trained using SVM Logistic regression, Gradient Boosting classifier, Random Forest Classifier Neural Network classification algorithms on the data set, as a preliminary idea before concentrating on implementation. The kfold algorithm is used because it is aimed to bring the accuracy rate closer to real life. In K-Folds Cross Validation, we divide our data into k different subsets. K-1 subsets are used to train our data and leave the final subset as test data. The average error value obtained because of k experiments indicates the validity of our model.

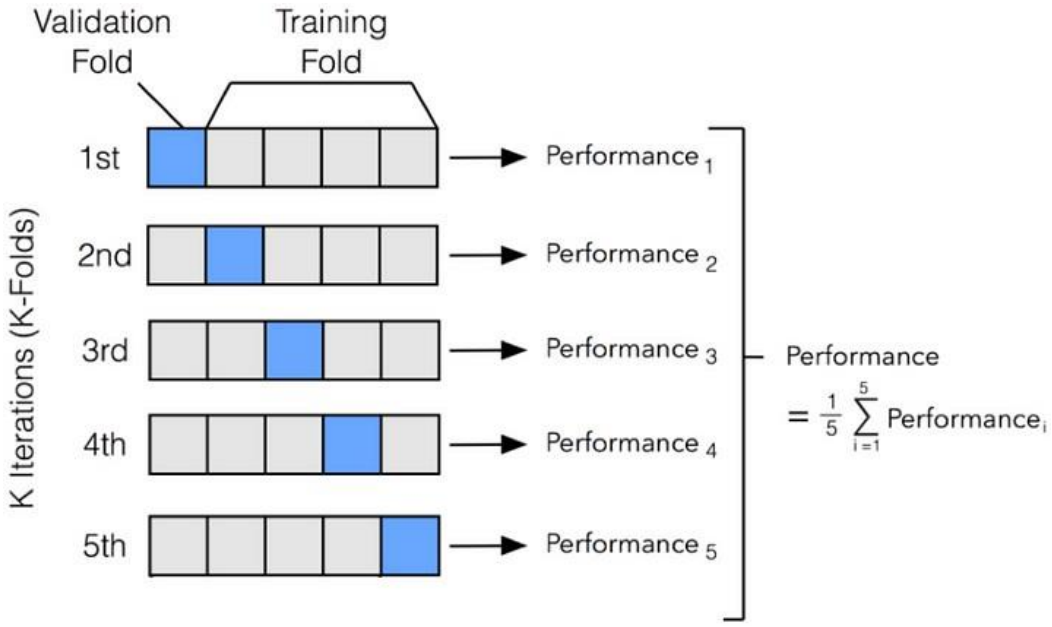

Figure 5. 9 Kfold implementation - ogundur (2021)

In addition to Logistic Regression, the use of neural network will continue, hyperparameters will be examined for hyperparameter Tuning, and finally, they will be compared with the Logistic regression model. The resulting model will thus be ready foruse.

Libraries to be used in Python are as follows:

1) Sklearn: Will be used for model training and evaluation.

2) Pandas: Will be used for Dataframe structure to deal with data more easily

3) Numpy: Will be used to perform a prediction on the trained model

4) Matplotlib: Will be used to visualize model results.

All Python processes have been completed through Visual Studio Code.

CHAPTER SIX

TEST/EXPERIMENTS

This section attempts to provide an overview of the testing process. To provide a better experience to users while maintaining a scalable and high-quality application, testing is important.

The model development step was carried out as follows.

First of all, the datasets created for Human Activity Recognition were investigated. Afterwards, comparisons were made over these datasets and it was decided to use the Wisdom dataset. Work has begun on Wisdom 1.0. The following results were obtained in studies on AutoML.

Random Forest

	Normalizer		Hyper Parameters								Metrics			
											Weighted			
id	Type	parameters	bootstrap	class_weight	criterion	max_features	min_samples_leaf	min_samples_split	n_estimators	oob_score	AUC	Precision	Recall	Accuracy
1	Sparse Normalizer	"norm": "l2"	TRUE	balanced	gini	sqrt	0.01	0.01	100	TRUE	0.92011	0.69376	0.72751	0.73090
2	Standart Scale Wrapper	"with_mean": false, "with_std": false	NA	NA	NA	NA	NA	NA	NA	NA	0.98431	0.87702	0.88409	0.88409
3	MinMax Scaler	NA	FALSE	null	gini	0.5	0.061578947	0.01	50	FALSE	0.93867	0.72591	0.72591	0.72591
4	MaxAbs Scaler	NA	TRUE	null	gini	log2	0.01	0.01	10	FALSE	0.95464	0.77217	0.79476	0.79476
										Average	0.94943	0.76722	0.78307	0.78392

Confusion Matrix

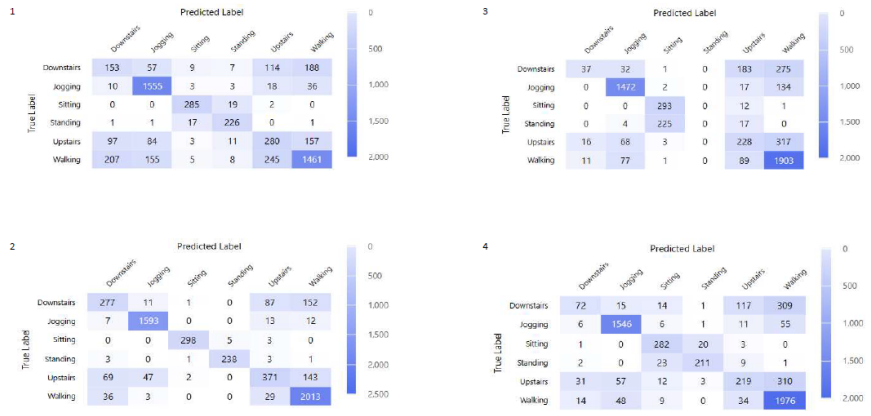

Figure 6. 1 Random forest results

Logistic Regression

	Normalizer		Hyper Parameters					Metrics			
								Weighted			
id	Type	parameters	C	class_weight	multi_class	penalty	solver	AUC	Precision	Recall	Accuracy
1	MinMax Scaler	NA	11.51395399	balanced	multinomial	l2	lbfgs	0.92131	0.74542	0.70210	0.70210
2	Standart Scale Wrapper	"with_mean": true, "with_std": true	2222.996483	null	multinomial	l2	lbfgs	0.92897	0.74688	0.78848	0.78848
3	MaxAbs Scaler	NA	4714.866363	balanced	multinomial	l2	lbfgs	0.91929	0.74637	0.70561	0.70561
							Average	0.92319	0.74622	0.73206	0.73206

Confusion Matrix

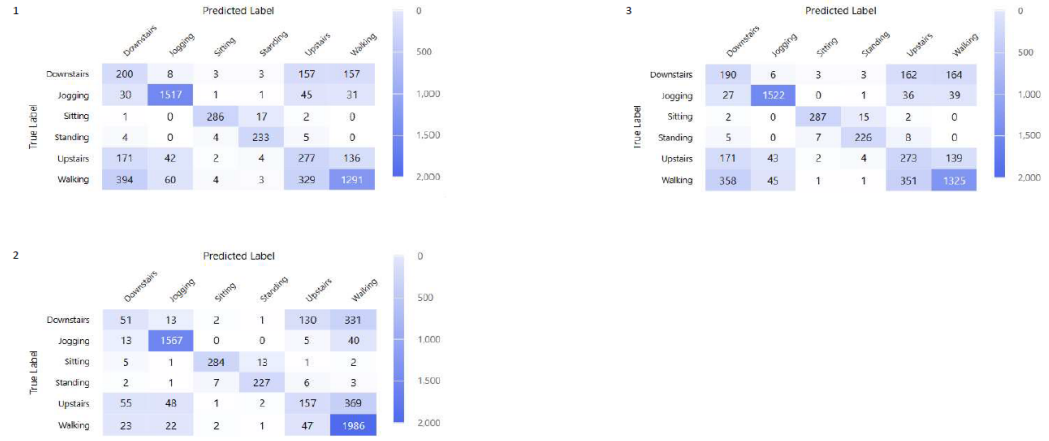

Figure 6. 2 Logistic regression results

Gradient Boosting

	Normalizer		Hyper Parameters							Metrics			
										Weighted			
id	Type	parameters	criterion	Learning Rate	max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	subsamples	AUC	Precision	Recall	Accuracy
1	Standard Scaler	"with_mean": true,"with_std": false	friedman	0.1	9	null	0.01	0.056842105	25	0.95263158	0.97892	0.86007	0.86951	0.86951

Confusion Matrix

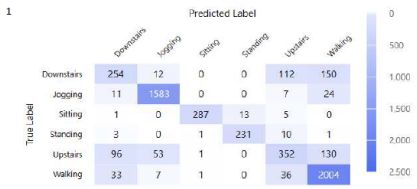

Figure 6. 3 Gradient boosting results

XGBoost Classifier

	Normalizer		Hyper Parameters														Metrics			
																		Weighted		
id	Type	parameters	booster	colsample_bytree	eta	grow_policy	max_bin	gamma	max_depth	max_leaves	n_estimators	objective	reg_alpha	reg_lambda	subsample	tree_method	AUC	Precision	Recall	Accuracy
1	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.9	0.1	NA	NA	0	6	3	25	reg:logistic	0	0.72916667	0.5	auto	0.9760583	0.8524124	0.8624954	0.8624954
2	SparseNormalizer	"norm": "max"	gbtree	0.8	0.3	NA	NA	0.1	10	63	10	reg:logistic	1.25	0.72916667	1	auto	0.9517163	0.7889358	0.8141381	0.8141381
3	SparseNormalizer	"norm": "max"	gbtree	0.8	0.3	NA	NA	0	6	0	10	reg:logistic	0	0.625	0.8	auto	0.9455776	0.775825	0.804725	0.804725
4	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.6	0.2	lossguide	1023	NA	3	0	100	reg:logistic	2.0833333	1.45833333	0.5	hist	0.9788832	0.8638257	0.8726467	0.8726467
5	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	1	0.3	NA	NA	0	8	255	10	reg:logistic	0.8333333	1.97916667	0.8	auto	0.9795273	0.8672948	0.8750461	0.8750461
6	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.6	0.1	lossguide	255	0	6	3	800	reg:logistic	1.6666667	0.20833333	0.6	hist	0.9790832	0.8668156	0.874677	0.874677
7	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.9	0.3	NA	NA	5	6	0	100	reg:logistic	0.1041667	1.45833333	0.6	auto	0.978707	0.8605648	0.8687708	0.8687708
8	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.6	0.3	lossguide	255	NA	4	0	100	reg:logistic	1.5625	1.04166667	1	hist	0.9855618	0.8850164	0.8909192	0.8909192
9	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.5	0.2	NA	NA	0	5	0	100	reg:logistic	0.5208333	0.52083333	0.8	auto	0.9858324	0.8857345	0.8920266	0.8920266
10	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.5	0.3	NA	NA	0	10	255	10	reg:logistic	0	0.10416667	0.7	auto	0.980097	0.8660638	0.8750461	0.8750461
11	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.5	0.5	NA	NA	1	6	0	25	reg:logistic	0	1.875	0.9	auto	0.9832893	0.8764831	0.8827981	0.8827981
12	StandardScalerWrapper	"with_mean": false,"with_std": false	gbtree	0.6	0.4	NA	NA	0	8	31	25	reg:logistic	0.7291667	1.35416667	1	auto	0.9863753	0.8845988	0.89055	0.89055
13	MaxAbsScaler	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	auto	0.9879029	0.8920817	0.8973791	0.8973791
																Average	0.9768163	0.8588963	0.8693245	0.8693245

Figure 6. 4 XGBoost classifier results

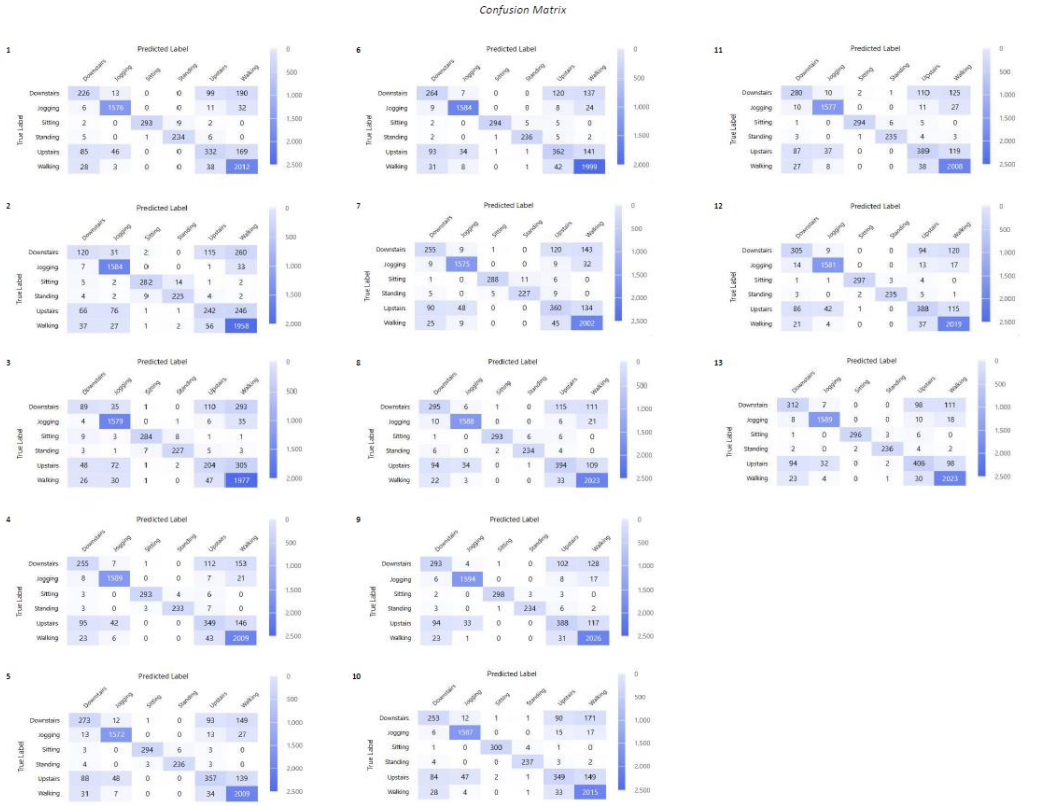

Figure 6. 5 Confusion matrix of xgboost classifier

Extreme Random Trees

	Normalizer		Hyper Parameters								Metrics			
											Weighted			
id	Type	parameters	bootstrap	class_weight	criterion	max_features	min_samples_leaf	min_samples_split	n_estimators	oob_score	AUC	Precision	Recall	Accuracy
1	Min Max Scaler	NA	FALSE	balanced	gini	0.9	0.01	0.01	25	FALSE	0.96350	0.96350	0.96350	0.96350
2	Robust Scaler	"quantile_range": [10,90]	FALSE	balanced	gini	0.8	0.01	0.337894737	25	FALSE	0.91988	0.73912	0.74289	0.74289
3	Standard Scaler Wrapper	param_kwargs: { "with_mean": false, "with_std": true	FALSE	balanced	gini	null	0.01	0.056842105	200	FALSE	0.95636	0.80380	0.79753	0.79753
4	MaxAbsScaler	NA	FALSE	balanced	entropy	log2	0.01	0.01	400	FALSE	0.93176	0.76347	0.76264	0.76264
5	MaxAbsScaler	NA	FALSE	null	gini	0.7	0.035789474	0.01	10	FALSE	0.93630	0.73208	0.76135	0.76135
6	MinMaxScaler	NA	TRUE	null	gini	null	0.01	0.056842105	50	FALSE	0.05684	0.73385	0.77815	0.77815
										Average	0.79411	0.78930	0.80101	0.80101

Figure 6. 6 Extreme random trees results

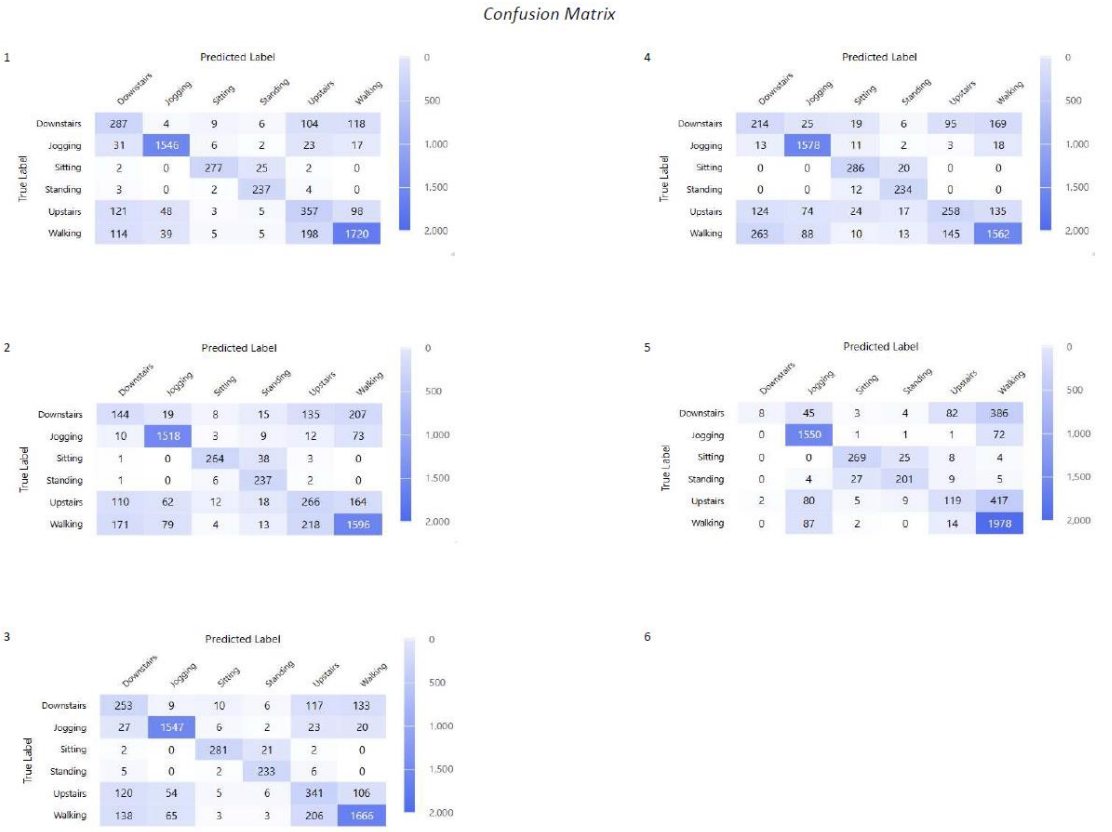

Figure 6. 7 Confusion matrix of extreme random trees

Light GBM

id	Type	Normalizer parameters	Hyper Parameters boosting_type	colsample_bytree	learning_rate	max_bin	max_depth	min_child_weight	min_data_in_leaf	min_split_gain	n_estimators	num_leaves	reg_alpha	reg_lambda	subsample	Metrics AUC	Weighted Precision	Weighted Recall	Weighted Accuracy
1	MaxAbs Scaler	NA	NA	NA	NA	NA	NA	NA	20	NA	NA	NA	NA	NA	NA	0.9887598	0.8992009	0.9034699	0.9034699
2	SparseNormalizer	"norm": "l1"	goss	0.792222222	0.094737368	80	6	0	0.05178966	0.894736842	25	140	0.263157895	0	1	0.926302	0.7367113	0.7644887	0.7644887
3	MinMax Scaler	NA	gbdt	0.495555556	0.094737368	140	4	8	0.00345731	0.421052632	10	113	0.421052632	0.526315789	0.198421053	0.958406	0.7957742	0.8097084	0.8097084
4	StandardScalerWrapper	with_mean: false, "with_std": true	gbdt	0.198888889	0.068474211	150	7	9	0.010353793	0.315789474	25	200	0	0.210526316	0.346842105	0.970423	0.811392	0.8277962	0.8277962
5	MaxAbs Scaler	NA	gbdt	0.297777778	0.089474737	60	3	4	0.072416552	0.789473684	800	224	0.263157895	0.157894737	0.544736842	0.9750374	0.9750374	0.8619417	0.8619417
6	StandardScalerWrapper	with_mean: false, "with_std": true	gbdt	0.495555556	0.094737368	140	6	0	0.082760345	0.105263158	25	164	0.315789474	0.315789474	0.594210526	0.9581698	0.785307	0.8062016	0.8062016
7	Robust Scaler	quantile_range: [10, 90], "with_centering": true, "with_scaling": false	goss	0.99	0.047373684	260	10	8	0.048281034	0.789473684	200	59	0.736842105	0.736842105	1	0.736842111	0.837774	0.8493909	0.8493909
8	StandardScalerWrapper	with_mean: true, "with_std": false	goss	0.396666667	0.00001	70	7	2	0.093104138	0.947368421	50	173	0.894736842	0.368421053	1	0.9286313	0.1475714	0.3840901	0.3840901
9	MaxAbs Scaler	NA	goss	0.594444444	0.026323158	310	-1	3	0.00001	0.789473684	50	131	0.368421053	1	1	0.36842105	0.8567357	0.8652639	0.8652639
10	MaxAbsScaler	NA	gbdt	0.396666667	0.089474737	200	8	4	0.031041379	0.157894737	200	101	0.631578947	0.210526316	0.792105263	0.9838738	0.8802104	0.8868586	0.8868586
11	StandardScaler	with_mean: true, "with_std": true	goss	0.396666667	0.073684842	90	-1	3	0.013801724	0.631578947	200	224	0.789473684	0.473684211	1	0.9825665	0.8755005	0.8820598	0.8820598
12	MinMax Scaler	NA	gbdt	0.792222222	0.078949474	340	6	7	0.017249655	0.578947368	200	254	0.631578947	0.105263158	0.693157895	0.9822605	0.8716772	0.8794758	0.8794758
																Average	0.8966411	0.7894767	0.8100621	0.8100621

Figure 6. 8 Light gbm results

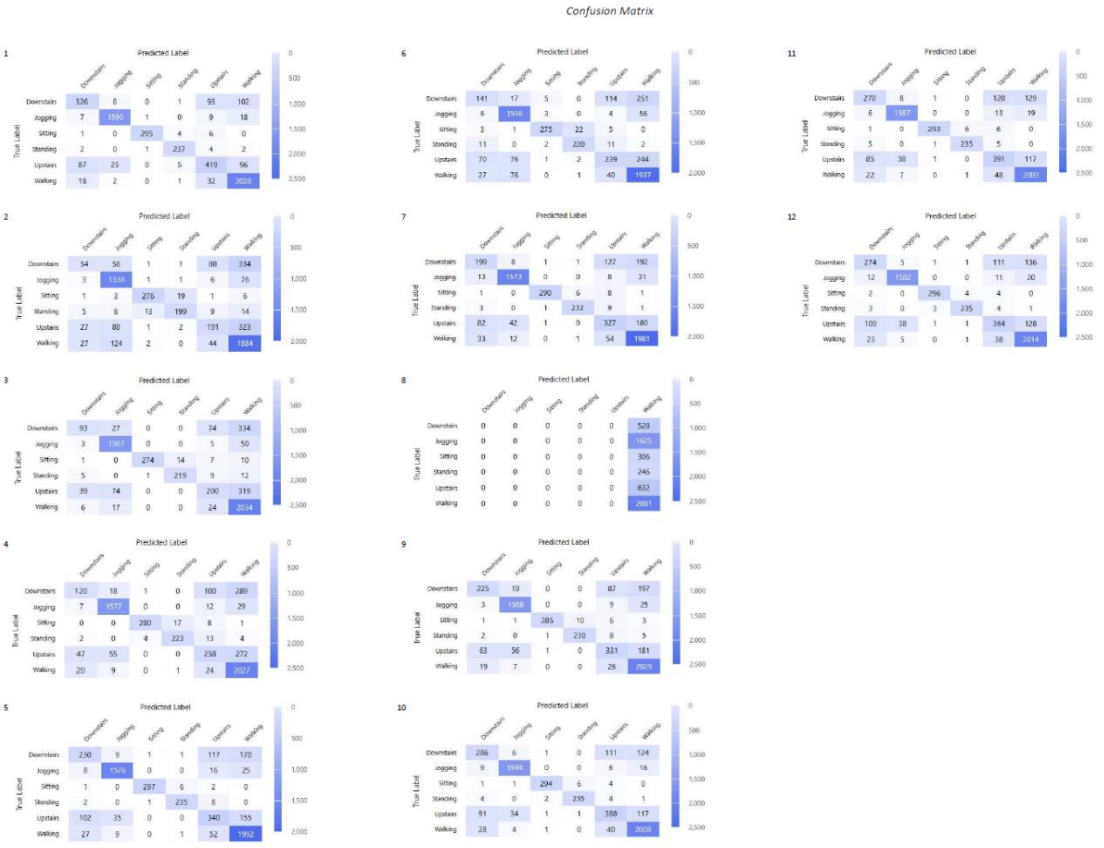

Figure 6. 9 Confusion matrix of Light gbm

When the error metrics above were examined, it was determined that the models with the highest accuracy on average were the XGBoost Classifier. The underlying reason for looking at the average at this stage was that it created any illusions due to overfitting of the models. The average accuracy values and Confusion Matrixes of the models made using different hyperparameters were examined and among the Random Forest, Logistic Regression, Gradient Boosting, XGBoost Classifier, Extreme Random Trees and Light GBM models, the XGBoost Classifier model was chosen in terms of both high accuracy and minimal time requirement. Hyperparameter tunings were examined and decided as follows:

'booster': 'gbtree',
'colsample_bytree': 0.5,
'eta': 0.2,
'grow_policy': 'depthwise',
'max_bin': 2,
'gamma': 0,
'max_depth': 5,
'max_leaves': 0,
'n_estimators': 100,
'objective': 'reg:logistic',
'reg_alpha': 0.5208333,
'reg_lambda': 0.52083333,
'subsample': 0.8,
'tree_method': 'auto'

After the model was trained and tested on Wisdom 1.0 data, the accuracy rate was 86%, but this rate dropped drastically during the validation phase. When the Wisdom data were examined, it was found that the time intervals were different in the sliding window stage. This caused us to obtain very bad results in the validation part, although the test rate was high in our real-life trials. After the raw data was taken, the data times were taken into account and the windowing process was applied again, the same problem was encountered in the data. Thereupon, it was decided to work on the Wisdom 2.0 dataset, and all the processes were applied again using raw data and a similar result was met again. It was then decided to examine the differences in human activities between people to examine what might be causing this. After researching, articles were found that even people's gender had an impact on their steps (Ji.et al. 2005). In addition, it was emphasized that some gait disorders had serious effects on the accuracy of the model. When these results are brought together, with the new system developed, users will now keep their phones in their pockets for 10-15 minutes, perform certain daily activities, and accordingly, more personal models will be created. These models will be used in the later stages of determining the activities of that user. Considering the purpose of the project, it will be ensured that the target audience is a more inclusive

model, not just a segment that walks in the correct posture, climbs the stairs, runs, rests, and walks. Some of the studies on this are as follows:

- A person performs certain activities in 15 minutes
- The model is trained using these data.

- The KFold algorithm is used to determine the accuracy of the model (Hyperparameters: "n_splits=3, shuffle=True")
- Cross Validation Mean detected as 0.97845

Among the 234 attempts made on the validation set, 22 errors were encountered. The majority of these errors were found to be from similar activities. 91% accuracy was achieved on validation. It was agreed that this validation set, which was created completely independently, was a very satisfactory result.

In general, this application, which provides great convenience in terms of ease of use, was encountered with very low error metrics after person-based model development, and the project gained a much more inclusive and egalitarian position.

CHAPTER SEVEN

CONCLUSION

In this thesis, 6 models were studied by using the accelerometer in a mobile phone and a comprehensive study on human activity recognition was presented. Our primary goal was to accurately identify and categorize various human activities based on sensor data, and in doing so, it was our primary goal for every human being to benefit from this service. Therefore, a system was developed in order to collect the data to be used in the training of the models in a very short time from the person who will use it. The installation phase is a system that will only take the user 10 minutes.

Among the models examined, it was decided to use the XGBoost Classifier and very high accuracy rates of 97% in the testing phase and 91% in the validation phase were obtained.

We also conducted a comprehensive evaluation of our model using various performance metrics such as accuracy, weighted precision, weighted recall and weighted AUC score. The results obtained demonstrate the effectiveness and reliability of our approach in accurately predicting human activities. When the results were examined, it was found that very productive results were determined.

The contributions of this thesis are important in the field of human activity recognition. By providing accurate and real-time activity recognition, our work can be applied to many fields, including health monitoring, sports analytics, military, aged care, and smart environments, and at this stage it is the most powerful of artificial intelligence and machine learning as it is fast, usable, and open to use by anyone. It offers a "just" environment, which is one of its important rules.

It is planned to be implemented in the future and used in various industries and sectors where the current situation of people is critical. Some examples of these are the elderly, disabled, babies, and soldiers. In addition, various improvements will be made in order to further increase the accuracy rate by taking into account the prolongation of the time by using Artificial Neural Networks by going further to the development part.

REFERENCES

Abdulkareem, N. M., & Abdulazeez, A. M. (2021). Machine learning classification based on Radom Forest Algorithm: A review. International Journal of Science andBusiness, 5(2), 128-142.

Andrews, J., Gkountouna, O., & Blaisten-Barojas, E. (2022). Forecasting moleculardynamics energetics of polymers in solution from supervised machine learning. Chemical science, 13(23), 7021.

Copeland, J. L., & Esliger, D. W. (2009). Accelerometer assessment of physical activityin active, healthy older adults. Journal of Aging & Physical Activity, 17(1).

De-La-Hoz-Franco, E., Ariza-Colpas, P., Quero, J. M., Espinilla, M., (2018) "Sensor Based Datasets for Human Activity Recognition – A Systematic Review of Literature," in IEEE Access, 6, 59192-59210, 2018, doi: 10.1109/ACCESS.2018.2873502.

Gupta, N., Gupta, S.K., Pathak, R.K. et al (2022). Human activity recognition in artificial intelligence framework: a narrative review. Artif Intell Rev 55, 4755–4808(2022). https://doi.org/10.1007/s10462-021-10116-x

Hassan, M. M., Uddin, M. Z., Mohamed, A., & Almogren, A. (2018). A robust human activity recognition system using smartphone sensors and deep learning. Future Generation Computer Systems, 81, 307-313.

Jayalath, S., Abhayasinghe, N., & Murray, I. (2013, October). A gyroscope based accurate pedometer algorithm. In International Conference on Indoor Positioning andIndoor Navigation (Vol. 28, p. 31st).

Ji, T., & Pachi, A. (2005). Frequency and velocity of people walking. Struct. Eng, 84(3), 36-40.

Lara, O. D., & Labrador, M. A. (2012). A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials, 15(3), 1192-1209.

Nweke, H. F., Wah, T. Y., Mujtaba, G., Al-garadi, M. A. (2019), Data fusion and multiple classifier systems for human activity detection and health monitoring: Review and open research directions, Information Fusion, 46, 2019, 147-170, ISSN 1566-2535, https://doi.org/10.1016/j.inffus.2018.06.002

Ogundur, G. (2021, December 13). Model Seçimi-K Fold Cross Validation - Gulcan Ogundur - Medium. Medium. https://medium.com/@gulcanogundur/model- se%C3%A7imi-k-fold-cross-validation-4635b61f143c

Tran, D. N., & Phan, D. D. (2016, January). Human activities recognition in android smartphone using support vector machine. In 2016 7th international conference on intelligent systems, modelling and simulation (isms) (pp. 64-68). IEEE.

Wahid. A. (Apr. 36,2017). Machine Learning Types. Slideshare. https://www.slideshare.net/awahid/big-data-and-machine-learning-for-businesses .

Wan, S., Qi, L., Xu, X., Tong, C., & Gu, Z. (2020). Deep learning models for real-time human activity recognition with smartphones. Mobile Networks and Applications, 25(2), 743-755.

Wisesa, I. W. W., & Mahardika, G. (2019, April). Fall detection algorithm based on accelerometer and gyroscope sensor data using Recurrent Neural Networks. In IOP Conference Series: Earth and Environmental Science (Vol. 258, No. 1, p. 012035).IOP Publishing.

Yadav, S. K., Tiwari, K., Pandey, H. M., Akbar, S. A. (2021), A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions, Knowledge-Based Systems,223, 2021, 106970, ISSN 0950-7051, https://doi.org/10.1016/j.knosys.2021.106970

Yin, J., Yang, Q., & Pan, J. J. (2008). Sensor-based abnormal human-activity detection.

IEEE transactions on knowledge and data engineering, 20(8), 1082-1090.

Zhang, S., Li, Y., Zhang, S., Shahabi, F., Xia, S., Deng, Y., Alshurafa (2022), N. DeepLearning in Human Activity Recognition with Wearable Sensors: A Review on Advances. *Sensors 2022*, 22, 1476. https://doi.org/10.3390/s22041476