

**TinyML**

**Mini Project-1**

Project name: Fall Detection using TinyML

Group Members : Harshit Gupta(2021UCS1621)

Ayush Kumar (2021UCS1636)

Komal Raj (2021UCS1649)

Batch: Computer Science Engineering ,Section-2

Collaboration Email: gear5.akh@gmail.com

Netaji Subhas University of Technology,Batch-2025,Sem-6

*Contents*

**01**

Overview of the Project

Technologies used in this project

**02**

**03**

Procedure to Implement the idea

Results and Conclusion

**04**

**01 Overview of the Project**

Fall Detection using TinyML

**Objective:**

The objective of this project is to develop a real-time fall detection system using TinyML to enhance the safety of individuals, especially the elderly or those living alone or the children of the house.This can also be used for detecting fall in accidents.

**SUse Case:**

**Scenario:**

Imagine an elderly person living alone at home. The fall detection system is integrated into a wearable device, such as a smartwatch, equipped with a variety of sensors including accelerometers and gyroscopes.

1. The user wears the smartwatch, which is powered by a TinyML model specifically trained for fall detection.
2. The TinyML model has been trained on a diverse dataset to accurately recognize patterns associated with falls.

.

**Initialis**

**-ation**

1. The smartwatch continuously monitors the user's movements using its onboard sensors.
2. The TinyML model runs locally on the smartwatch, allowing for real-time analysis without the need for continuous connectivity to a central server.
3. During normal activities like walking or sitting, the TinyML model recognizes the regular patterns of movement and identifies them as non-fall events.

Real time

Monitoring

1.In the event of a fall, the accelerometers and gyroscopes on the smartwatch capture sudden and abnormal changes in motion.

2.The TinyML model analyzes the sensor data and quickly identifies the deviation from normal movement patterns associated with a fall event.

**Fall Event Detection**

**02 Technologies and Hardware used project**

**Machine Learning**

**Arduino Nano**

**Accelerometer**

**Edge Impulse**

**Accelerometer:**

The MPU6050 is a popular Inertial Measurement Unit (IMU) that combines a 3-axis gyroscope and a 3-axis accelerometer in a single chip. It is commonly used in various applications such as robotics, drones, virtual reality, and motion sensing devices. The accelerometer in the MPU6050 measures the acceleration along the x, y, and z axes.

**Arduino Nano:**

The Arduino Nano is a compact and versatile development board designed for easy prototyping and DIY projects. It features the ATmega328 microcontroller running at 16 MHz and offers 32KB of Flash memory for program storage. With 14 digital and 8 analog I/O pins, it provides flexibility for connecting various sensors, actuators, and other components. The Nano is especially suitable for projects with space constraints, as its small form factor makes it breadboard-friendly. It supports popular communication interfaces like I2C, SPI, and UART, making it compatible with a wide range of sensors and modules. The Arduino Nano is a popular choice for hobbyists, students, and professionals engaged in electronics and programming.

Please enter case name or case picture

Please enter case name or case picture

Please enter case name or case picture

Please enter case name or case picture

**03 Procedure to implement the data**

**Data Collection**

**Experience**

**Using Edge Impulse :**

In the first step,we collected real time data using edge impulse and gyrometer sensors present in mobile phones.

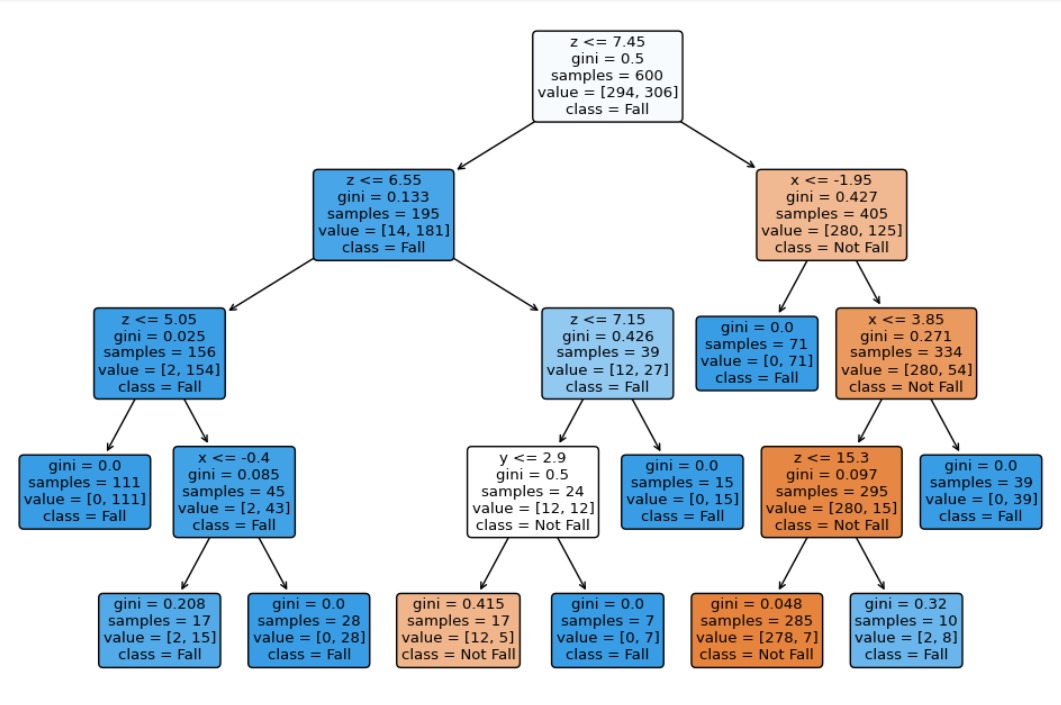
Using that data we trained the model to catogerise our data as fall or not fall.



**ML model**

**Decision Tree Model used for classification:**

Classification of fall or not fall is done using the following tree and implemented using the given python code.



**ML CODE:**

# Import necessary libraries

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier, plot\_tree

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.metrics import accuracy\_score, classification\_report

data = pd.read\_csv('/content/drive/MyDrive/kaggle/fall\_and\_walk.csv')

X = data.iloc[:,:-1]

y = data.iloc[:, -1] # Assuming 'label' column contains fall (1) or not fall (0)

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# # Create a decision tree classifier

classifier = DecisionTreeClassifier(max\_depth=4)

classifier.fit(X\_train, y\_train)

# Make predictions on the test set

y\_pred = classifier.predict(X\_test)

# Evaluate the model

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy}')

# Print classification report for more detailed evaluation

print(classification\_report(y\_test, y\_pred))

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

plot\_tree(classifier, feature\_names=['x', 'y', 'z'], class\_names=['Not Fall', 'Fall'], filled=True, rounded=True)

plt.show()

**Arduino**

**Nano**

**Using Arduino IDE**

Wrote the following .imo file in arduino ide in order to pass the parameters of DT model used in ML to integrate ML with arduino Nano.Given the .imo file that has to be run in Arduino IDE.

**Arduino Code:**

#include <Adafruit\_MPU6050.h>

#include <Adafruit\_Sensor.h>

#include <Wire.h>

Adafruit\_MPU6050 mpu;

void setup(void) {

Serial.begin(115200);

if (!mpu.begin()) {

Serial.println("Failed to find MPU6050 chip");

while (1) {

delay(10);

}

}

mpu.setAccelerometerRange(MPU6050\_RANGE\_8\_G);

mpu.setGyroRange(MPU6050\_RANGE\_500\_DEG);

mpu.setFilterBandwidth(MPU6050\_BAND\_21\_HZ);

delay(100);

}

void loop() {

/\* Get new sensor events with the readings \*/

sensors\_event\_t a, g, temp;

mpu.getEvent(&a, &g, &temp);

float x=a.acceleration.x,y=a.acceleration.y,z=a.acceleration.z;

bool flag=0;

if(z<=7.45){

if(z<=6.55){

if(z<=5.05){

Serial.println("Fall");

flag=1;

}

else{

Serial.println("Fall");

flag=1;

}

}

else{

if(x<=-0.95){

Serial.println("Fall");

flag=1;

}

else{

Serial.println("Not Fall");

flag=0;

}

}

}

else{

if(x<=-1.95){

Serial.println("Fall");

flag=1;

}

else{

if(x<=3.85){

Serial.println("NOt Fall");

flag=0;

}

else{

Serial.println("Fall");

flag=1;

}

}

}

delay(1000);

}

if(z<=7.45){

if(z<=6.55){

if(z<=5.05){

Serial.println("Fall");

flag=1;

}

else{

Serial.println("Fall");

flag=1;

}

}

else{

if(x<=-0.95){

Serial.println("Fall");

flag=1;

}

else{

Serial.println("Not Fall");

flag=0;

}

}

}

else{

if(x<=-1.95){

Serial.println("Fall");

flag=1;

}

else{

if(x<=3.85){

Serial.println("NOt Fall");

flag=0;

}

else{

Serial.println("Fall");

flag=1;

}

}

}

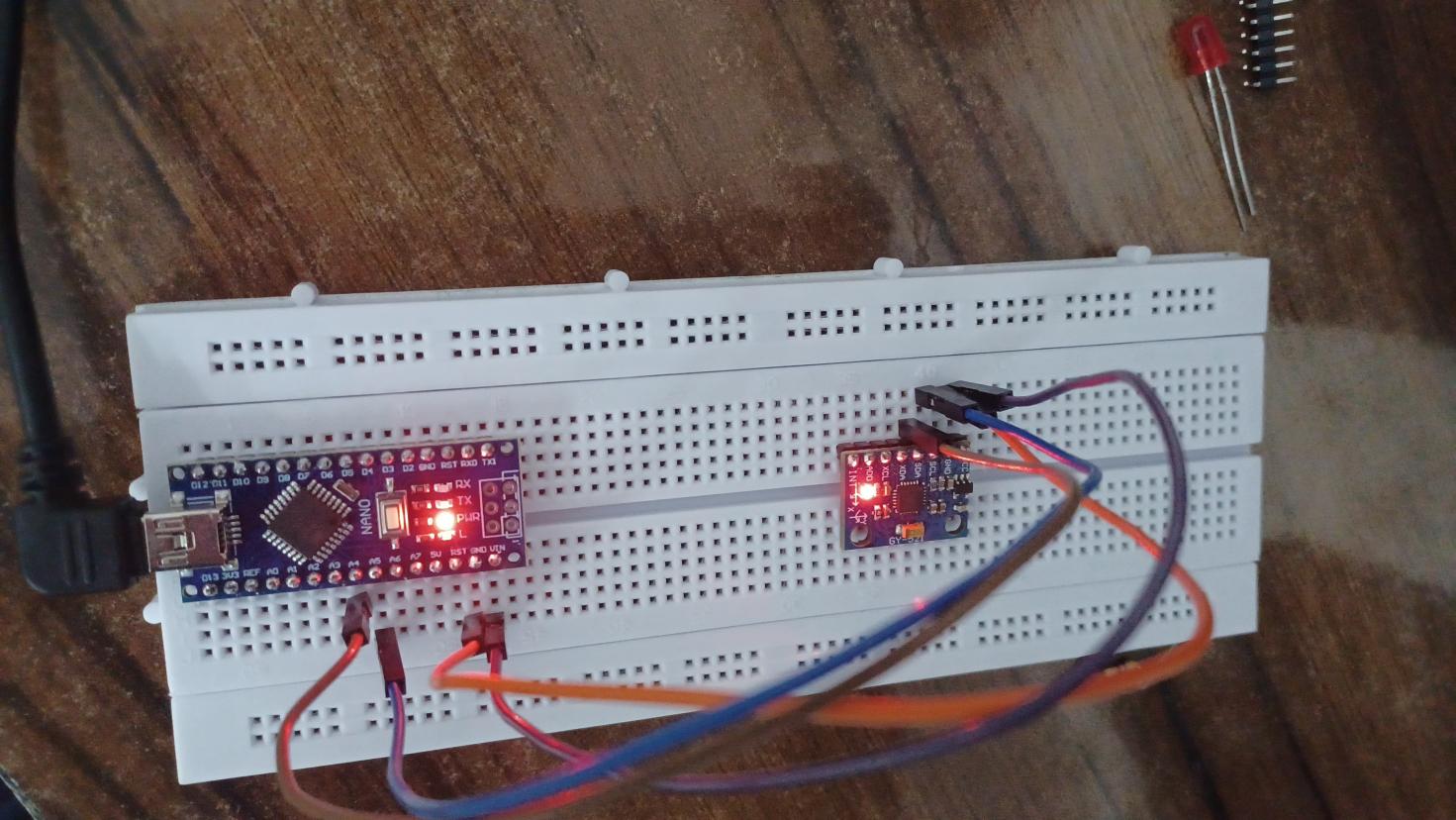
delay(1000);

}

**Arduino Nano**

**and Accerelometer Circuit**

A



**Enter important indicator tasks**

**Conclusion & Benifits**

**This TinyML-based fall detection system provides an unobtrusive and efficient solution to enhance the safety and well-being of individuals, allowing them to maintain an independent lifestyle while having a reliable safety net in case of emergencies.**

**Benifits**

* Increased safety for individuals prone to falls, especially the elderly.
* Rapid response to potential emergencies, minimizing the time it takes for assistance to reach the user.
* Low-power consumption and real-time processing, thanks to the use of TinyML.

**Enter the main items or achievements or tasks to be completed here**



**20XX COME ON！**

I hope the company will be better next year!