Netaji Subhas University of Technology

Hardware and Software Tools
TinyML Project

Srishti 2023UCM3581
Vandana 2023UCM2820

Instructor:
Ms. Saurabhi Choudhary

TinyML-Based Multi-Parameter Smart Healthcare Monitoring System

1. Abstract

This project presents a TinyML-based smart healthcare monitoring system implemented on an Arduino microcontroller. The system performs real-time health condition classification using multiple physiological parameters including body temperature, muscle strain (flex sensor), heart intensity simulation (potentiometer), and motion detection (PIR sensor). A lightweight neural network was trained offline using a synthetic dataset consisting of 150 labeled samples across three classes: NORMAL, STRESS, and CRITICAL. The trained model parameters (weights and biases) were deployed directly onto the microcontroller for on-device inference. The system demonstrates a complete TinyML pipeline including dataset generation, preprocessing, model training, parameter extraction, and embedded deployment. Results show successful multi-class classification with real-time feedback through an LCD display and buzzer alert system. This project highlights the feasibility of implementing machine learning models on low-power embedded systems for healthcare monitoring applications.

2. Introduction

TinyML refers to the deployment of machine learning models on low-power microcontrollers with limited computational resources and memory capacity. Unlike traditional cloud-based AI systems, TinyML enables real-time, edge-based inference without internet connectivity. This makes it highly suitable for healthcare monitoring systems where low latency, portability, and power efficiency are critical.

Healthcare monitoring devices often rely on threshold-based logic (e.g., if temperature > 38°C). However, such rule-based systems fail to capture complex multi-parameter relationships. Machine learning models, particularly neural networks, are capable of combining multiple physiological features to make more intelligent decisions.

The objective of this project is to design and implement a multi-parameter health monitoring system using TinyML principles. The system performs embedded neural network inference to classify patient condition into three categories:

- NORMAL
- STRESS
- CRITICAL

This work demonstrates how end-to-end TinyML workflows can be executed using simulation tools and deployed onto microcontroller hardware.

3. System Architecture

The system consists of the following components:

Hardware Components

- Arduino Uno (Microcontroller)
- TMP36 Temperature Sensor
- Flex Sensor (Muscle strain simulation)
- Potentiometer (Heart intensity simulation)
- PIR Sensor (Motion detection)
- I2C 16x2 LCD Display
- Piezo Buzzer

System Flow

Sensors
→ Data Acquisition
→ Feature Normalization
→ Neural Network Inference
→ Classification Output
→ LCD Display & Alert System

The Arduino reads sensor values, applies normalization using precomputed mean and standard deviation values, performs forward propagation through a neural network, and selects the output class using an argmax function.

4. Dataset Generation

A synthetic dataset was generated to simulate physiological health conditions. The dataset consists of 150 samples distributed equally among three classes:

- 50 NORMAL samples
- 50 STRESS samples
- 50 CRITICAL samples

Each sample contains four features:

- Temperature (°C)
- Flex sensor value (0–1023)
- Heart intensity (0–1023)
- Motion state (0 or 1)

Feature ranges were designed to represent realistic physiological conditions:

Class	Temperature	Flex	Heart	Motion
NORMAL	36–37°C	Low	Low	1
STRESS	37.5–38.5°C	Medium	Medium	0
CRITICAL	39–40.5°C	High	High	0

The dataset was normalized using StandardScaler to ensure proper scaling before training.

5. Model Training

The model was trained using Python in Google Colab with Scikit-learn's `MLPClassifier`.

Model Architecture:

- Input Layer: 4 neurons (Temp, Flex, Heart, Motion)
- Hidden Layer: 4 neurons
- Activation Function: ReLU
- Output Layer: 3 neurons (Softmax-style argmax classification)

Training achieved:

Training Accuracy: 100%

After training, the following parameters were extracted:

- Hidden layer weight matrix
- Hidden layer bias vector
- Output layer weight matrix
- Output layer bias vector
- Feature normalization means and standard deviations

These parameters were manually embedded into Arduino code for on-device inference.

6. Embedded TinyML Deployment

The trained neural network was implemented on the Arduino Uno using:

1. Feature scaling using stored mean and standard deviation
2. Matrix multiplication for hidden layer
3. ReLU activation

4. Output layer computation
5. Argmax classification

The inference process runs continuously inside the Arduino loop function.

Based on prediction:

- NORMAL → No alert
- STRESS → Intermittent buzzer alert
- CRITICAL → Continuous buzzer alert

The classification result is displayed on the LCD screen.

7. Results and Observations

The system successfully performs real-time classification based on simulated sensor inputs.

Testing scenarios:

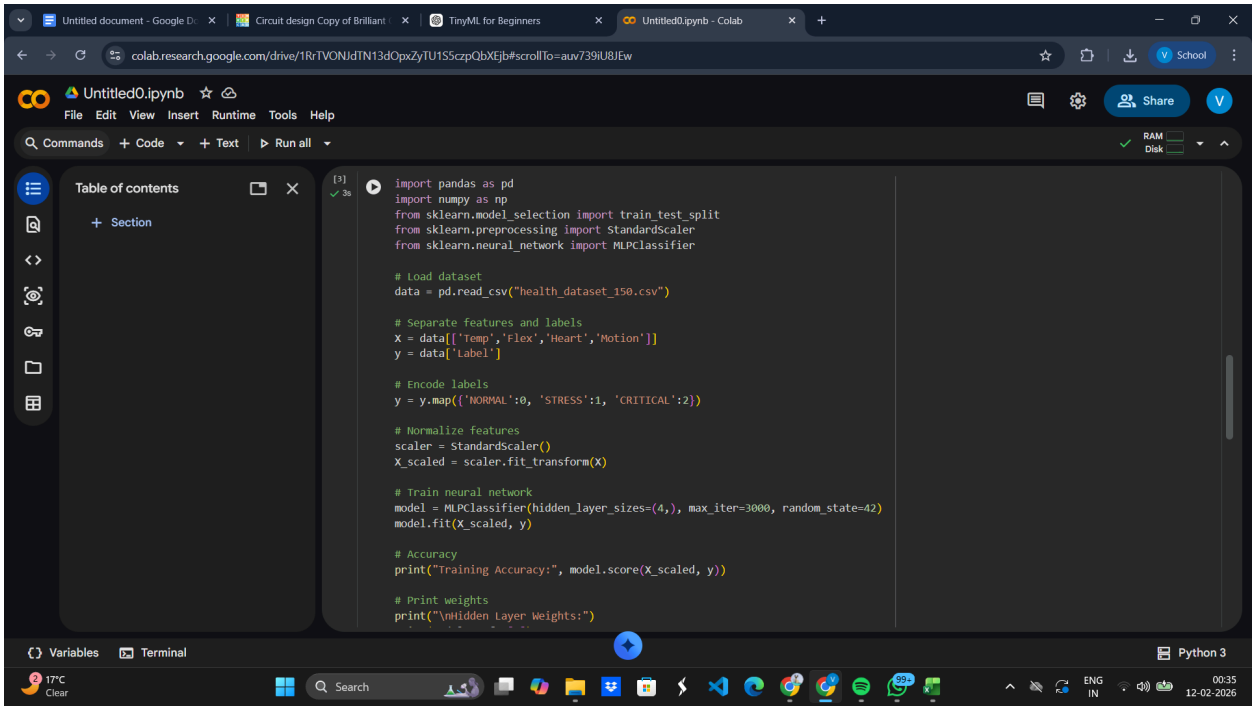
- Normal physiological values → Classified as NORMAL
- Elevated temperature and heart intensity → Classified as STRESS
- High temperature and high strain values → Classified as CRITICAL

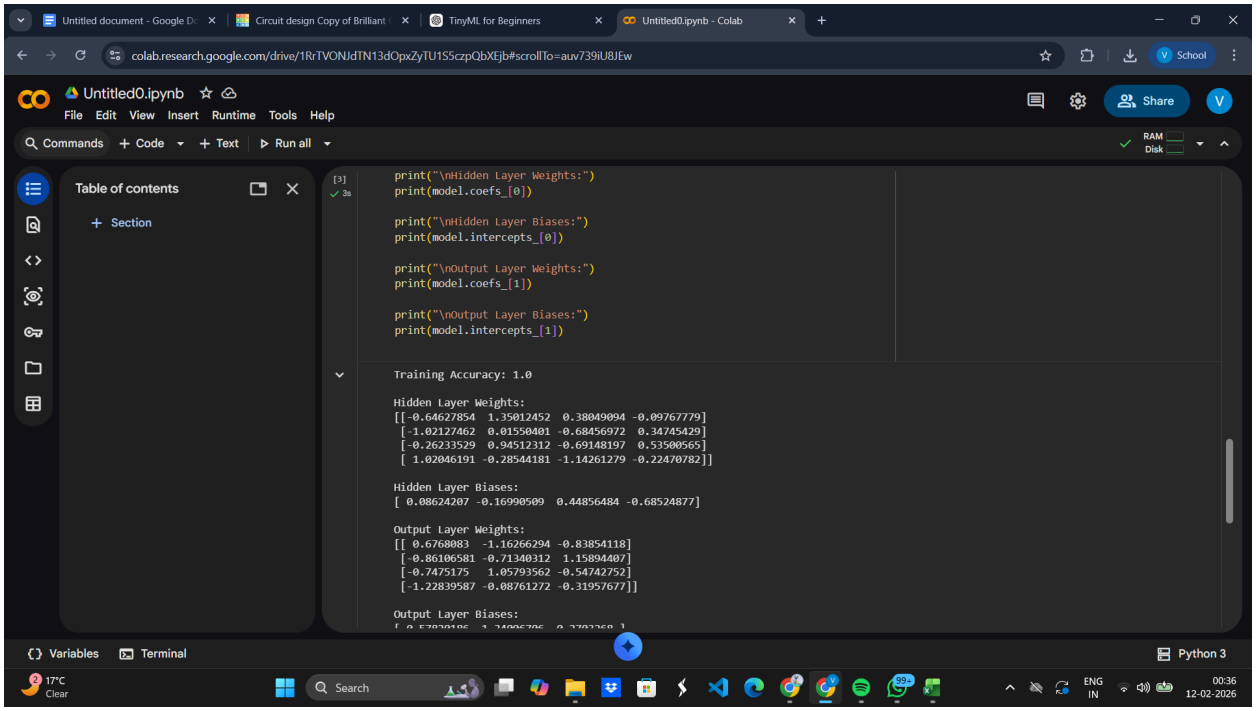
The system demonstrates:

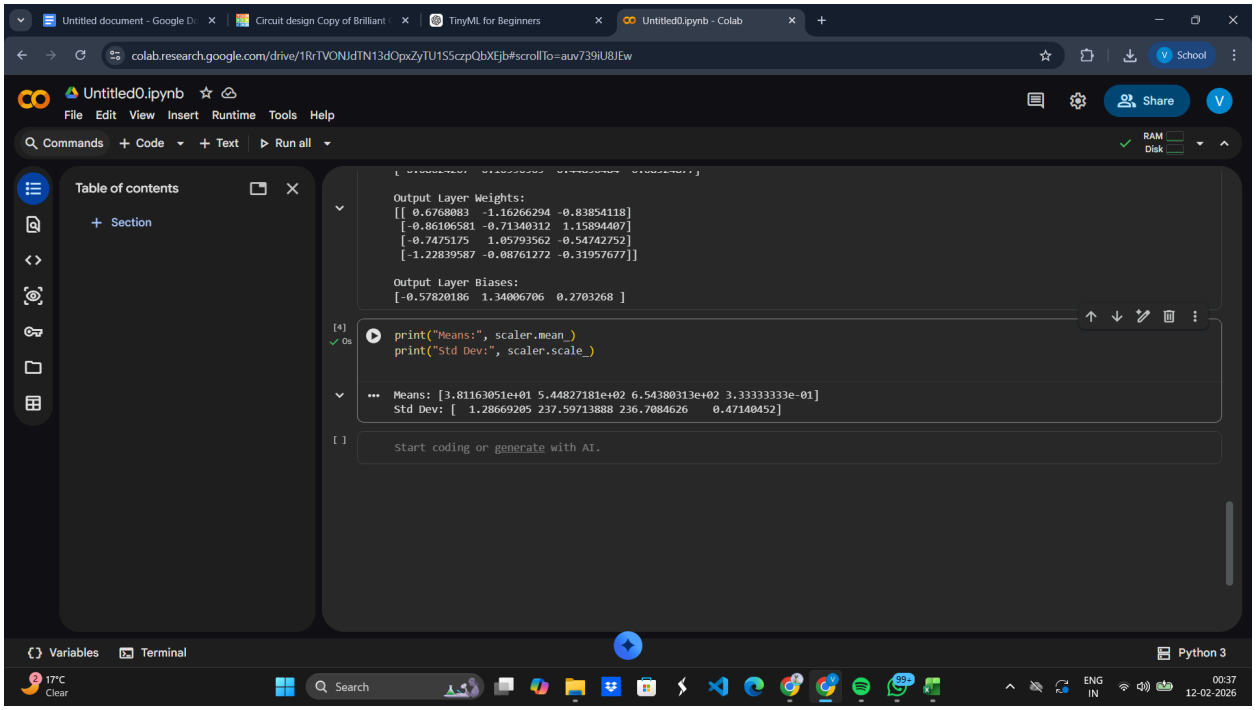
- Accurate multi-class classification
- Stable embedded inference
- Real-time response
- Efficient use of microcontroller resources
- Input validation was added to prevent unrealistic out-of-distribution sensor values from affecting predictions.

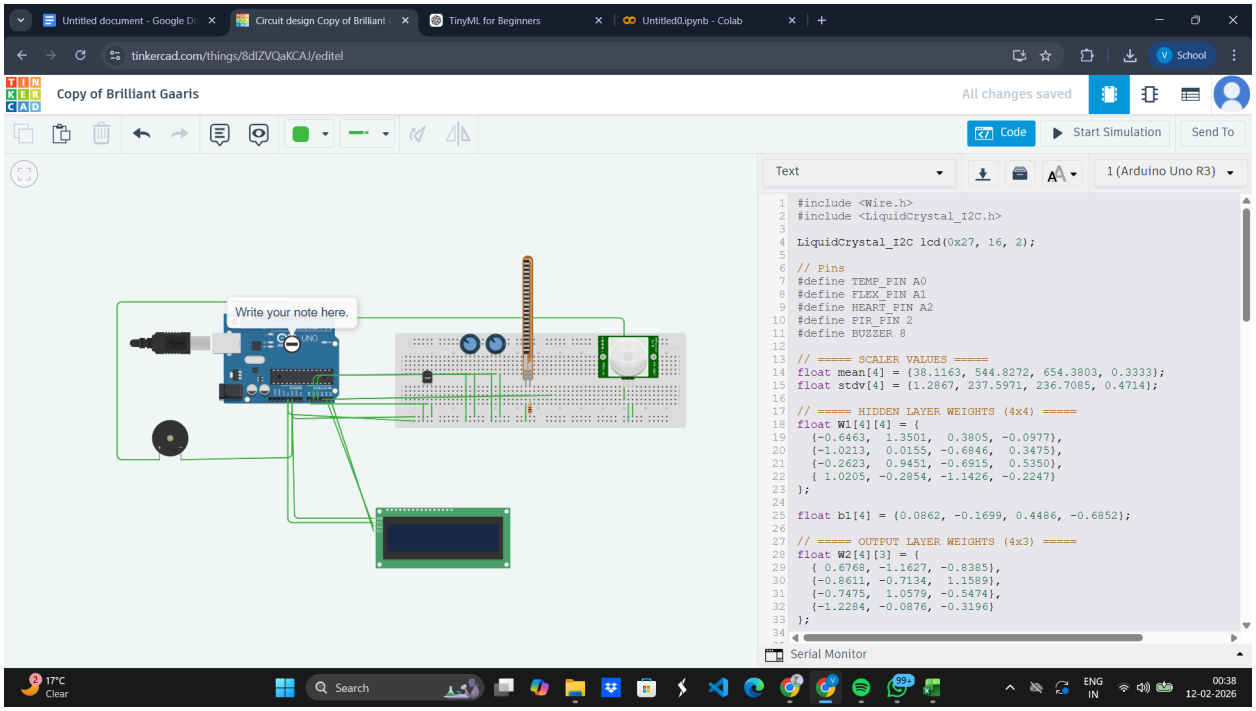
11. Conclusion

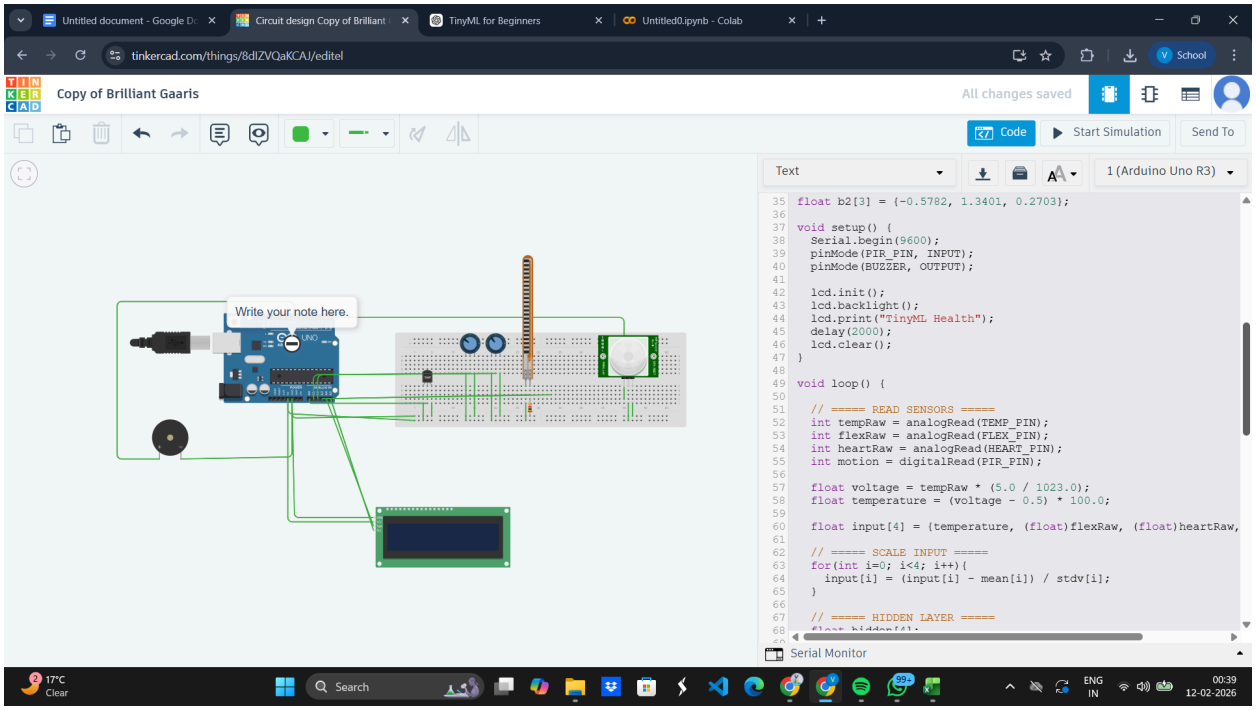
This project successfully demonstrates the implementation of a TinyML-based healthcare monitoring system on an Arduino microcontroller. The system performs real-time classification using a lightweight neural network trained offline and deployed for embedded inference. The work highlights the practical feasibility of integrating machine learning models into low-resource microcontrollers for intelligent edge healthcare applications. The project effectively showcases the complete TinyML workflow from dataset creation to embedded deployment.

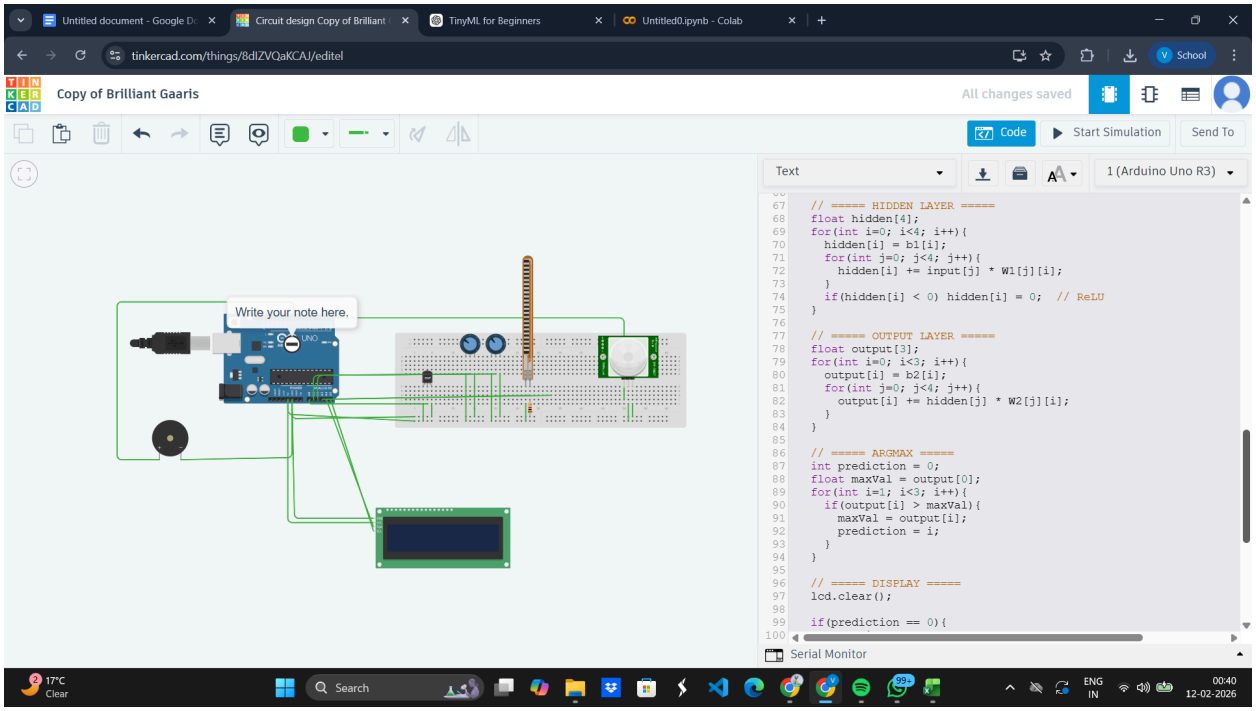

Untitled document - Google D
Circuit design Copy of Brilliant
TinyML for Beginners
Untitled0.ipynb - Colab
colab.research.google.com/drive/1RrTVONJdTN13dOpxZyTU1S5czpQbXEjb#scrollTo=auv739iU8JEw
Untitled0.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
Table of contents
+ Section
[3]
3s
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
# Load dataset
data = pd.read_csv("health_dataset_150.csv")
# Separate features and labels
X = data[['Temp','Flex','Heart','Motion']]
y = data['Label']
# Encode labels
y = y.map({'NORMAL':0, 'STRESS':1, 'CRITICAL':2})
# Normalize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Train neural network
model = MLPClassifier(hidden_layer_sizes=(4,), max_iter=3000, random_state=42)
model.fit(X_scaled, y)
# Accuracy
print("Training Accuracy:", model.score(X_scaled, y))
# Print weights
print("\nHidden Layer Weights:")
Variables
Terminal
Python 3
17°C
Clear
Search
ENG
IN
00:35
12-02-2026

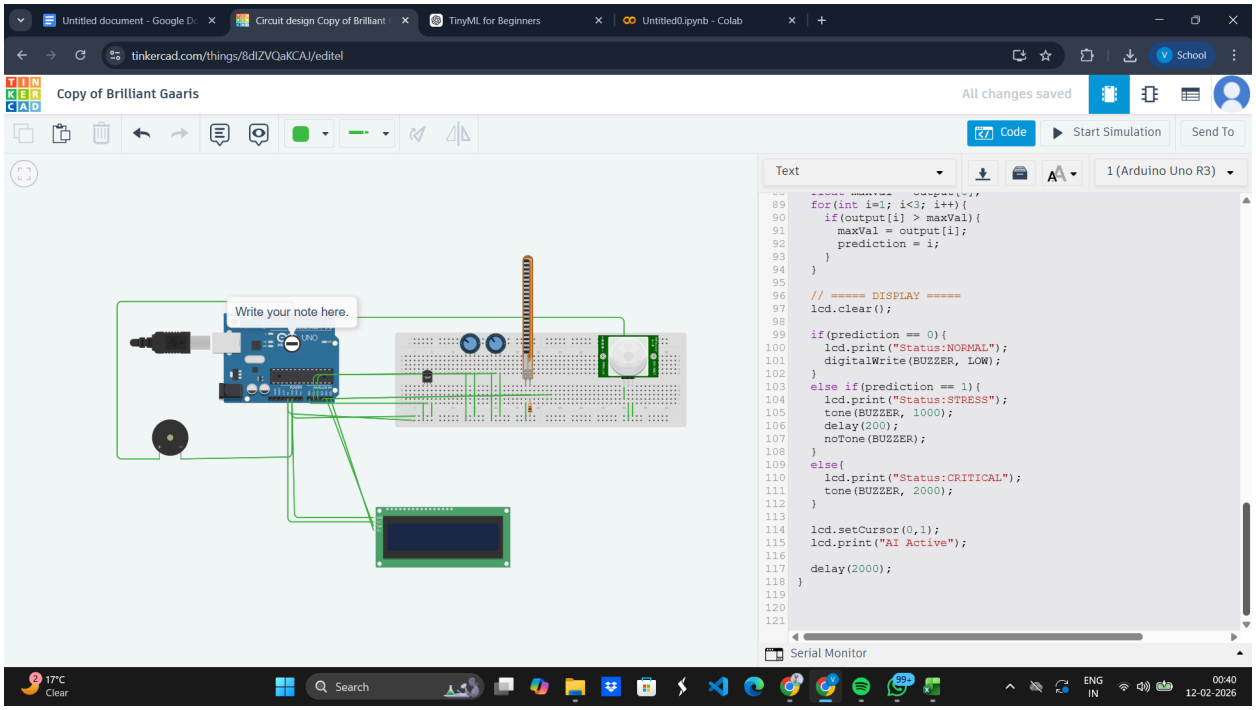

Untitled document - Google D
Circuit design Copy of Brilliant
TinyML for Beginners
Untitled0.ipynb - Colab
colab.research.google.com/drive/1RrTVONJdTN13dOpxZyTU1S5czpQbXEjb#scrollTo=auv739iU8JEw
Untitled0.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM
Disk
Table of contents
+ Section
[3]
3s
print("\nHidden Layer Weights:")
print(model.coefs_[0])
print("\nHidden Layer Biases:")
print(model.intercepts_[0])
print("\nOutput Layer Weights:")
print(model.coefs_[1])
print("\nOutput Layer Biases:")
print(model.intercepts_[1])
Training Accuracy: 1.0
Hidden Layer Weights:
[[-0.64627854 1.35012452 0.38049094 -0.09767779]
[-1.02127462 0.01550401 -0.68456972 0.34745429]
[-0.26233529 0.94512312 -0.69148197 0.53500565]
[1.02046191 -0.28544181 -1.14261279 -0.22470782]]
Hidden Layer Biases:
[0.08624207 -0.16990509 0.44856484 -0.68524877]
Output Layer Weights:
[[0.6768083 -1.16266294 -0.83854118]
[-0.86106581 -0.71340312 1.15894407]
[-0.7475175 1.05793562 -0.54742752]
[-1.22839587 -0.08761272 -0.31957677]]
Output Layer Biases:
Variables
Terminal
Python 3
17°C
Clear
Search
ENG
IN
00:36
12-02-2026


Untitled0.ipynb - Colab
colab.research.google.com/drive/1RrTVONJdTN13dOpxZyTU1S5czpQbXEjb#scrollTo=auv739iU8JEw
Untitled0.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
Table of contents
+ Section
Output Layer Weights:
[[0.6768083 -1.16266294 -0.83854118]
[-0.86106581 -0.71340312 1.15894407]
[-0.7475175 1.05793562 -0.54742752]
[-1.22839587 -0.08761272 -0.31957677]]
Output Layer Biases:
[-0.57820186 1.34006706 0.2703268]
print("Means:", scaler.mean_)
print("Std Dev:", scaler.scale_)
Means: [3.81163051e+01 5.44827181e+02 6.54380313e+02 3.33333333e-01]
Std Dev: [1.28669205 237.59713888 236.7084626 0.47140452]
Start coding or generate with AI.
Variables Terminal
Python 3
17°C Clear
Search
ENG IN
00:37 12-02-2026


Untitled document - Google D
Circuit design Copy of Brilliant
TinyML for Beginners
Untitled0.ipynb - Colab
tinkercad.com/things/8dIZVQaKCAJ/editel
Copy of Brilliant Gaaris
All changes saved
Code
Start Simulation
Send To
Text
1 (Arduino Uno R3)
Write your note here.
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
// Pins
#define TEMP_PIN A0
#define FLEX_PIN A1
#define HEART_PIN A2
#define PIR_PIN 2
#define BUZZER 8
// ===== SCALER VALUES =====
float mean[4] = {38.1163, 544.8272, 654.3803, 0.3333};
float stdv[4] = {1.2867, 237.5971, 236.7085, 0.4714};
// ===== HIDDEN LAYER WEIGHTS (4x4) =====
float W1[4][4] = {
{-0.6463, 1.3501, 0.3805, -0.0977},
{-1.0213, 0.0155, -0.6846, 0.3475},
{-0.2623, 0.9451, -0.6915, 0.5350},
{ 1.0205, -0.2854, -1.1426, -0.2247}
};
float b1[4] = {0.0862, -0.1699, 0.4486, -0.6852};
// ===== OUTPUT LAYER WEIGHTS (4x3) =====
float W2[4][3] = {
{ 0.6768, -1.1627, -0.8385},
{-0.8611, -0.7134, 1.1589},
{-0.7475, 1.0579, -0.5474},
{-1.2284, -0.0876, -0.3196}
};
Serial Monitor
17°C Clear
Search
ENG IN
00:38 12-02-2026


Untitled document - Google D
Circuit design Copy of Brilliant
TinyML for Beginners
Untitled0.ipynb - Colab
tinkercad.com/things/8dlZVQaKCAJ/editel
School
Copy of Brilliant Gaaris
All changes saved
Code
Start Simulation
Send To
Text
1 (Arduino Uno R3)
Write your note here.
float b2[3] = {-0.5782, 1.3401, 0.2703};
void setup() {
Serial.begin(9600);
pinMode(PIR_PIN, INPUT);
pinMode(BUZZER, OUTPUT);
lcd.init();
lcd.backlight();
lcd.print("TinyML Health");
delay(2000);
lcd.clear();
}
void loop() {
// ===== READ SENSORS =====
int tempRaw = analogRead(TEMP_PIN);
int flexRaw = analogRead(FLEX_PIN);
int heartRaw = analogRead(HEART_PIN);
int motion = digitalRead(PIR_PIN);
float voltage = tempRaw * (5.0 / 1023.0);
float temperature = (voltage - 0.5) * 100.0;
float input[4] = {temperature, (float)flexRaw, (float)heartRaw,
// ===== SCALE INPUT =====
for(int i=0; i<4; i++){
input[i] = (input[i] - mean[i]) / stdv[i];
}
// ===== HIDDEN LAYER =====
Serial Monitor
17°C
Clear
Search
ENG
IN
00:39
12-02-2026


Untitled document - Google D
Circuit design Copy of Brilliant
TinyML for Beginners
Untitled0.ipynb - Colab
tinkercad.com/things/8dlZVQaKCAJ/editel
School
Copy of Brilliant Gaaris
All changes saved
Code
Start Simulation
Send To
Text
1 (Arduino Uno R3)
Write your note here.
// ===== HIDDEN LAYER =====
float hidden[4];
for(int i=0; i<4; i++){
hidden[i] = b1[i];
for(int j=0; j<4; j++){
hidden[i] += input[j] * W1[j][i];
}
if(hidden[i] < 0) hidden[i] = 0; // ReLU
}
// ===== OUTPUT LAYER =====
float output[3];
for(int i=0; i<3; i++){
output[i] = b2[i];
for(int j=0; j<4; j++){
output[i] += hidden[j] * W2[j][i];
}
}
// ===== ARGMAX =====
int prediction = 0;
float maxVal = output[0];
for(int i=1; i<3; i++){
if(output[i] > maxVal){
maxVal = output[i];
prediction = i;
}
}
// ===== DISPLAY =====
lcd.clear();
if(prediction == 0){
Serial Monitor
17°C
Clear
Search
ENG
IN
00:40
12-02-2026

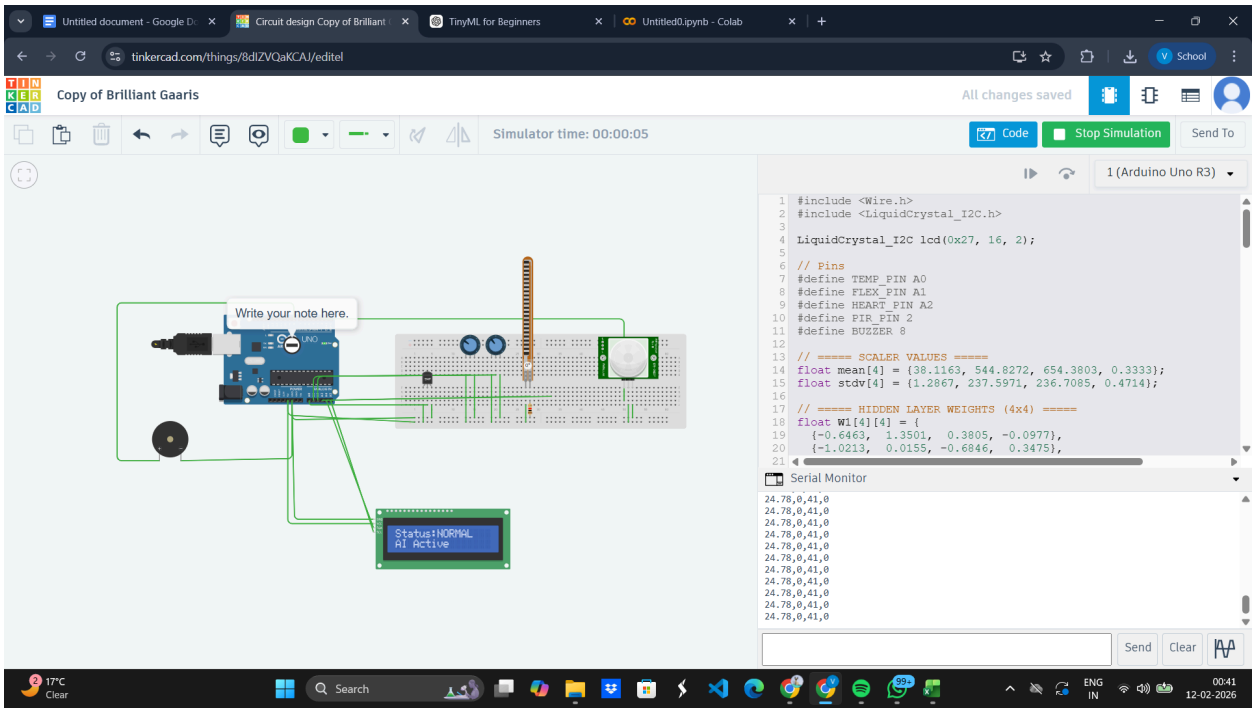


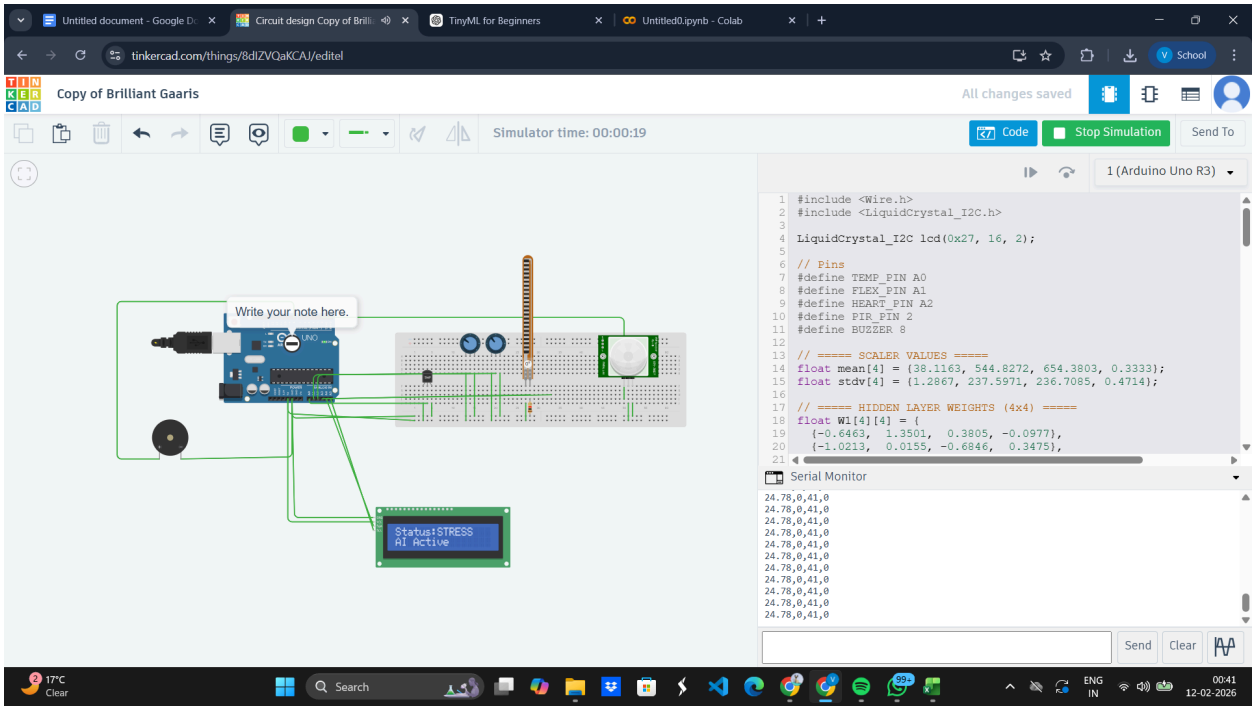

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

// Pins
#define TEMP_PIN A0
#define FLEX_PIN A1
#define HEART_PIN A2
#define PIR_PIN 2
#define BUZZER 8

// ===== SCALER VALUES =====
float mean[4] = {38.1163, 544.8272, 654.3803, 0.3333};
float stdv[4] = {1.2867, 237.5971, 236.7085, 0.4714};

// ===== HIDDEN LAYER WEIGHTS (4x4) =====
float W1[4][4] = {
  {-0.6463,  1.3501,  0.3805, -0.0977},
  {-1.0213,  0.0155, -0.6846,  0.3475},
  {-0.2623,  0.9451, -0.6915,  0.5350},
  { 1.0205, -0.2854, -1.1426, -0.2247}
};

float b1[4] = {0.0862, -0.1699, 0.4486, -0.6852};
```

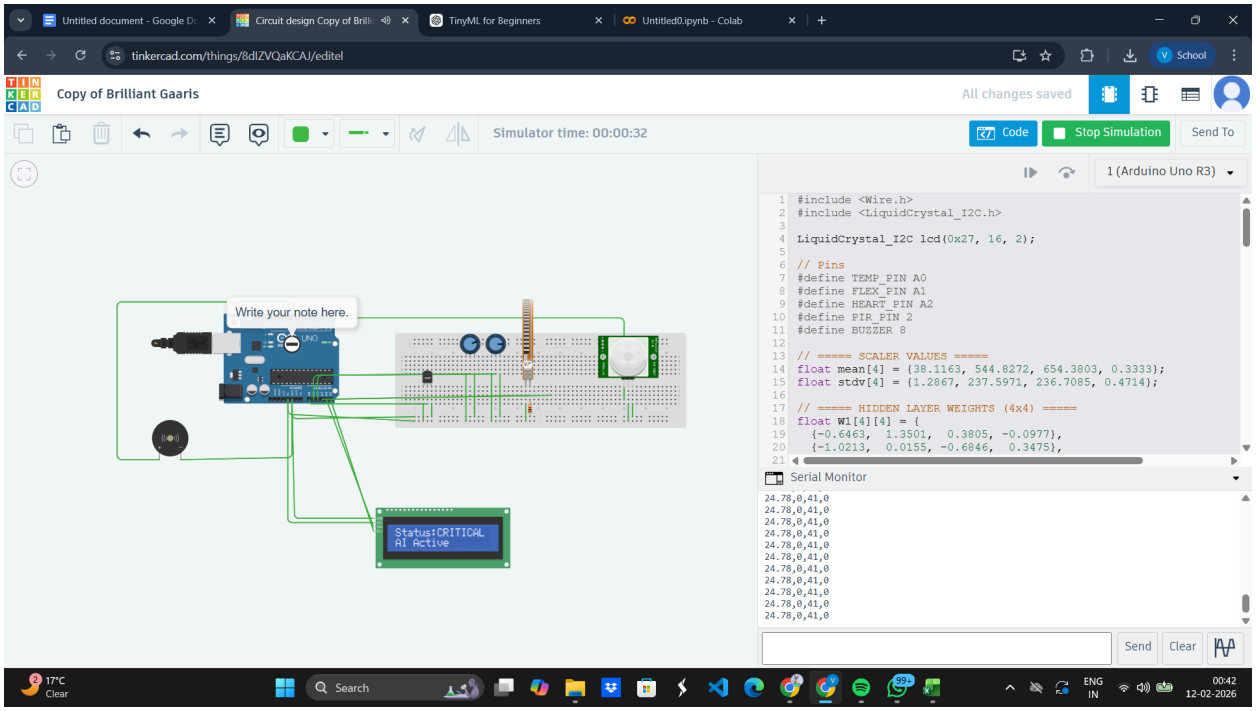
```
// ===== OUTPUT LAYER WEIGHTS (4x3) =====
float W2[4][3] = {
  { 0.6768, -1.1627, -0.8385},
  {-0.8611, -0.7134,  1.1589},
  {-0.7475,  1.0579, -0.5474},
  {-1.2284, -0.0876, -0.3196}
};

float b2[3] = {-0.5782, 1.3401, 0.2703};

void setup() {
  Serial.begin(9600);
  pinMode(PIR_PIN, INPUT);
  pinMode(BUZZER, OUTPUT);

  lcd.init();
  lcd.backlight();
  lcd.print("TinyML Health");
  delay(2000);
  lcd.clear();
}

void loop() {

  // ===== READ SENSORS =====
  int tempRaw = analogRead(TEMP_PIN);
  int flexRaw = analogRead(FLEX_PIN);
  int heartRaw = analogRead(HEART_PIN);
  int motion = digitalRead(PIR_PIN);

  float voltage = tempRaw * (5.0 / 1023.0);
  float temperature = (voltage - 0.5) * 100.0;

  float input[4] = {temperature, (float)flexRaw, (float)heartRaw, (float)motion};

  // ===== SCALE INPUT =====
  for(int i=0; i<4; i++){
    input[i] = (input[i] - mean[i]) / stdv[i];
  }

  // ===== HIDDEN LAYER =====
  float hidden[4];
  for(int i=0; i<4; i++){
```

```
  hidden[i] = b1[i];
  for(int j=0; j<4; j++){
    hidden[i] += input[j] * W1[j][i];
  }
  if(hidden[i] < 0) hidden[i] = 0;  // ReLU
}

// ===== OUTPUT LAYER =====
float output[3];
for(int i=0; i<3; i++){
  output[i] = b2[i];
  for(int j=0; j<4; j++){
    output[i] += hidden[j] * W2[j][i];
  }
}

// ===== ARGMAX =====
int prediction = 0;
float maxVal = output[0];
for(int i=1; i<3; i++){
  if(output[i] > maxVal){
    maxVal = output[i];
    prediction = i;
  }
}

// ===== DISPLAY =====
lcd.clear();

if(prediction == 0){
  lcd.print("Status:NORMAL");
  digitalWrite(BUZZER, LOW);
}
else if(prediction == 1){
  lcd.print("Status:STRESS");
  tone(BUZZER, 1000);
  delay(200);
  noTone(BUZZER);
}
else{
  lcd.print("Status:CRITICAL");
  tone(BUZZER, 2000);
}
```

```
  lcd.setCursor(0,1);
  lcd.print("AI Active");

  delay(2000);
}
```




Untitled document - Google D
Circuit design Copy of Brilli
TinyML for Beginners
Untitled0.ipynb - Colab
tinkercad.com/things/8dlZVQaKCAJ/editel
School
Copy of Brilliant Gaaris
All changes saved
Simulator time: 00:00:19
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
// Pins
#define TEMP_PIN A0
#define FLEX_PIN A1
#define HEART_PIN A2
#define PIR_PIN 2
#define BUZZER 8
// ===== SCALER VALUES =====
float mean[4] = {38.1163, 544.8272, 654.3803, 0.3333};
float stdv[4] = {1.2867, 237.5971, 236.7085, 0.4714};
// ===== HIDDEN LAYER WEIGHTS (4x4) =====
float W1[4][4] = {
{-0.6463, 1.3501, 0.3805, -0.0977},
{-1.0213, 0.0155, -0.6846, 0.3475},
Write your note here.
Status:STRESS
AI Active
Serial Monitor
24.78,0,41,0
Send
Clear
17°C
Clear
Search
ENG
IN
00:41
12-02-2026


Untitled document - Google D
Circuit design Copy of Brilli
TinyML for Beginners
Untitled0.ipynb - Colab
tinkercad.com/things/8dlZVQaKCAJ/editel
School
Copy of Brilliant Gaaris
All changes saved
Simulator time: 00:00:32
Code
Stop Simulation
Send To
1 (Arduino Uno R3)
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
// Pins
#define TEMP_PIN A0
#define FLEX_PIN A1
#define HEART_PIN A2
#define PIR_PIN 2
#define BUZZER 8
// ===== SCALER VALUES =====
float mean[4] = {38.1163, 544.8272, 654.3803, 0.3333};
float stdv[4] = {1.2867, 237.5971, 236.7085, 0.4714};
// ===== HIDDEN LAYER WEIGHTS (4x4) =====
float W1[4][4] = {
{-0.6463, 1.3501, 0.3805, -0.0977},
{-1.0213, 0.0155, -0.6846, 0.3475},
Write your note here.
Status:CRITICAL
AI Active
Serial Monitor
24.78,0,41,0
Send
Clear
17°C
Clear
Search
ENG
IN
00:42
12-02-2026