```
#include <EloquentTinyML.h>
#include <eloquent_tinyml/tensorflow.h>
#include "iris_model_data.h"

#define NUMBER_OF_INPUTS 4
#define NUMBER_OF_OUTPUTS 3
#define TENSOR_ARENA_SIZE 2 * 1024  // Increased to 4KB for stability

Eloquent::TinyML::TensorFlow::TensorFlow<NUMBER_OF_INPUTS, NUMBER_OF_OUTPUTS,
TENSOR_ARENA_SIZE> ml;

// Test each input if one input is under test keep another input inside the comments
//float input[NUMBER_OF_INPUTS] = {-0.8977, 1.0156, -1.3049, -1.2559};
float input[NUMBER_OF_INPUTS] = {7.0, 3.2, 4.7, 1.4};

void setup() {
  Serial.begin(115200);
  delay(4000);  // Increased delay for serial monitor connection

  // Initialize model with error checking
  if (!ml.begin(iris_model)) {
    Serial.println("❗ Failed to initialize model!");
    Serial.print("Error: ");
    Serial.println(ml.getErrorMessage());
    while (true);  // Halt on failure
  }

  Serial.println("🚀 Model initialized successfully");

  // Run inference with error checking
  float output[NUMBER_OF_OUTPUTS];
  int st_time = micros();
  ml.predict(input, output);
  int en_time = micros();

  Serial.println("📊 Predictions:");
  for (int i = 0; i < NUMBER_OF_OUTPUTS; i++) {
    Serial.print("Class ");
    Serial.print(i);
    Serial.print(": ");
    Serial.println(output[i], 5);
  }

  float max_prob = output[0];
  int predicted_class = 0;
  for (int i = 1; i < NUMBER_OF_OUTPUTS; i++) {
    if (output[i] > max_prob) {
      max_prob = output[i];
      predicted_class = i;
    }
  }
    Serial.println();
    Serial.print("☑ Predicted Class:");
    Serial.println(predicted_class);
    Serial.print("Inference Time:");
    Serial.print(en_time-st_time);
    Serial.println("micro sec.");
}
void loop() {
  // Empty
}
```