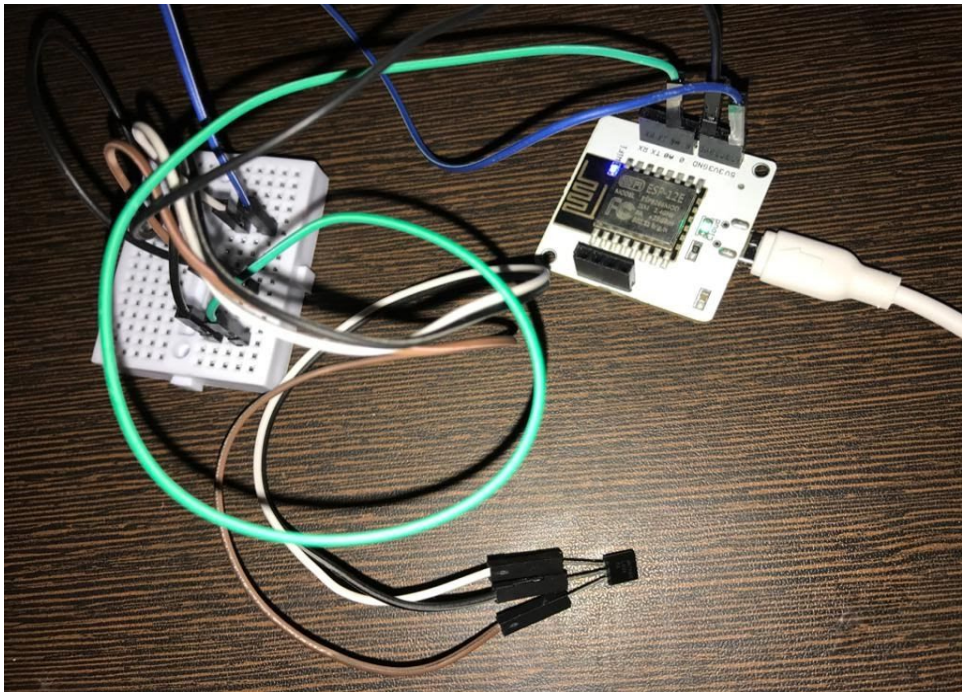
SMART TEMPERATURE MONITORING SYSTEM

Smart Temperature Monitoring System

1. Overview
2. Things used in this project
    - Hardware Components
    - Software apps and online services
3. Story
    - Introduction
    - Steps for building the project
4. Schematics

Overview

A smart temperature monitoring system which identifies the sudden unfavourable change in the temperature of the system using Machine Learning. And alerts the owner directly via messaging.



Things Used in this project

Hardware Components

1. Bolt WiFi Module
2. USB-A to Mini-USB Cable
3. LM35 Sensor
4. 3 Connecting wires (male-to-female)
5. Power Bank

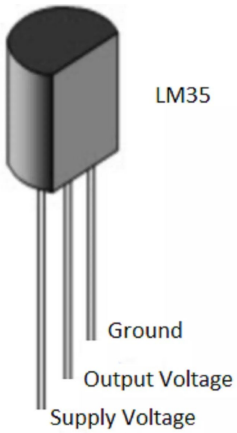
Software apps and Online Services

1. Bolt IoT Android app
2. Ubuntu
3. Twilio (SMS service)

Story

Introduction

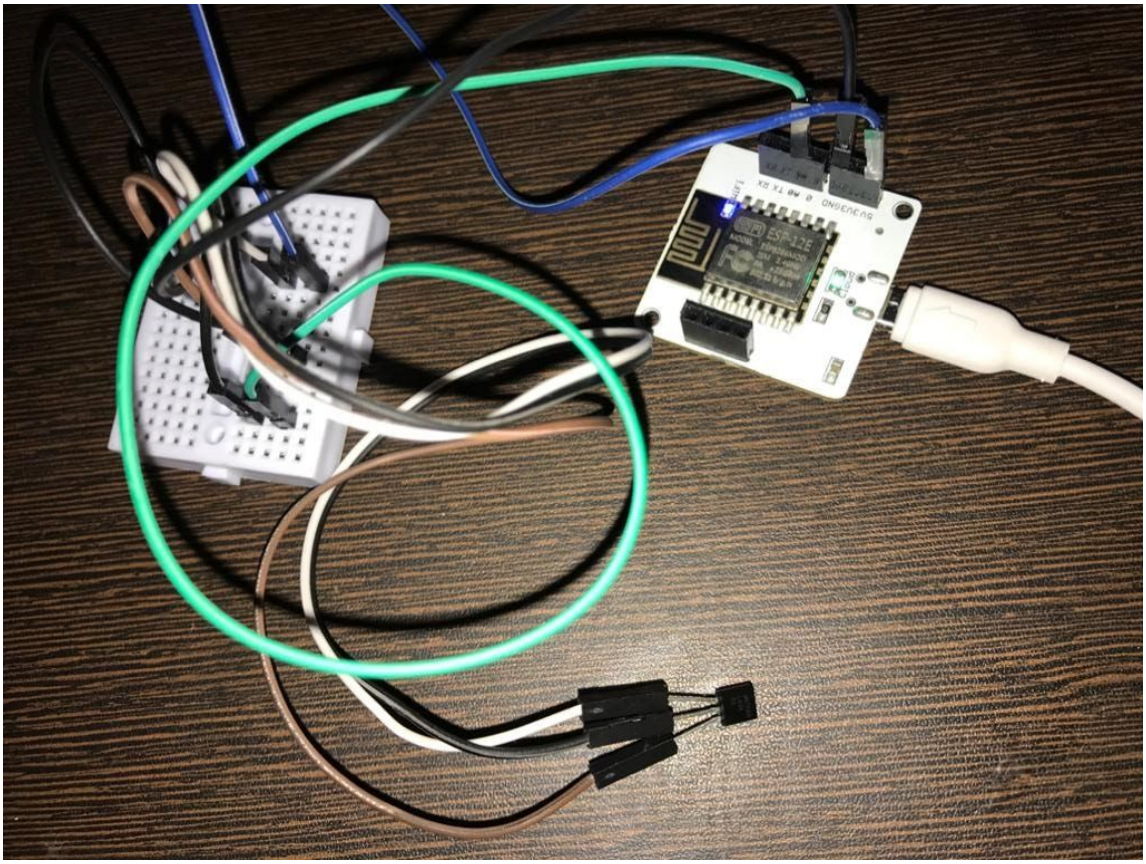
This project monitors the temperature behaviour of the system using the LM35 Sensor and learns the normal behavioural data using Machine Learning. It sends an alert to the owner via SMS the moment the behaviour of the temperature is unexpected by Anomaly Detection (Z - Score Analysis). By this, the temperature of the system while manufacturing or developing a temperature-sensitive product can be controlled without any need of a human 24 x 7 to monitor.



Steps to Build this Project

- Take the LM35 Sensor and hold it with the flat face towards yourself and the pins are downwards. Now from left to right, the pins are VCC, Output, GND (Ground) respectively.

- Connect the VCC to 5V, Output to AO & Ground to GND using male-to-female wires to the Bolt WiFi Module. [you can also use a breadboard for the connection]



- Power the WiFi Module by connecting it with USB Cable. You can either connect directly to the socket or Laptop (or power bank). Blue LED on the module will start blinking slowly.

- Go to https://cloud.boltiot.com and signup, then verify the email by clicking the link you've received on your Gmail Account. (or log in if you already have an account)

- Download the Bolt IoT Mobile App on your android or IOS. Log in using the same credentials as of the Bolt cloud. (previous step)

- On the Mobile App, click on “Add Device” after login. Then click on “Ready”. Now go to wifi settings in your mobile, then connect to the hotspot of the Bolt Module. When connected, the LED will start blinking fastly.

- Now go to the app, click “Continue”. Then Connect with the available WiFi to access the internet. Click on “Connect”. Successful connection to the internet can be confirmed with the Stable Blue LED and Stable Green LED on the WiFi Module.

- Now, go to https://cloud.boltiot.com, click on “Devices” on the left panel to check the device ID and status.

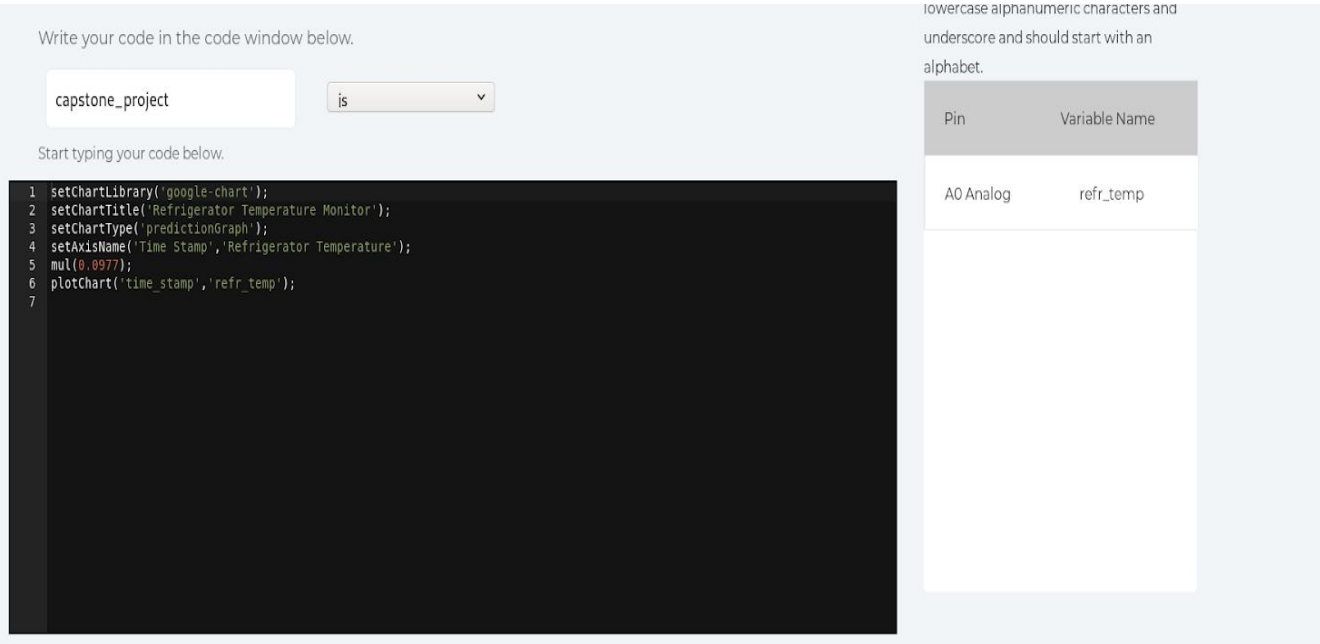
- Click on “Products” on the left panel to add a new product. Then write the name of your product. Select ‘Input Device’ & ‘GPIO’. Uploading an image is optional.

- Click on the product. Link the device to this product by clicking on the ‘chain symbol’ on the right side, then select the device.

- Now click on the ‘configure this product’ symbol. Under the Hardware section, select the data collection rate (e.g. 5 min) & select Analog A0, give a name to this variable in the right input box.

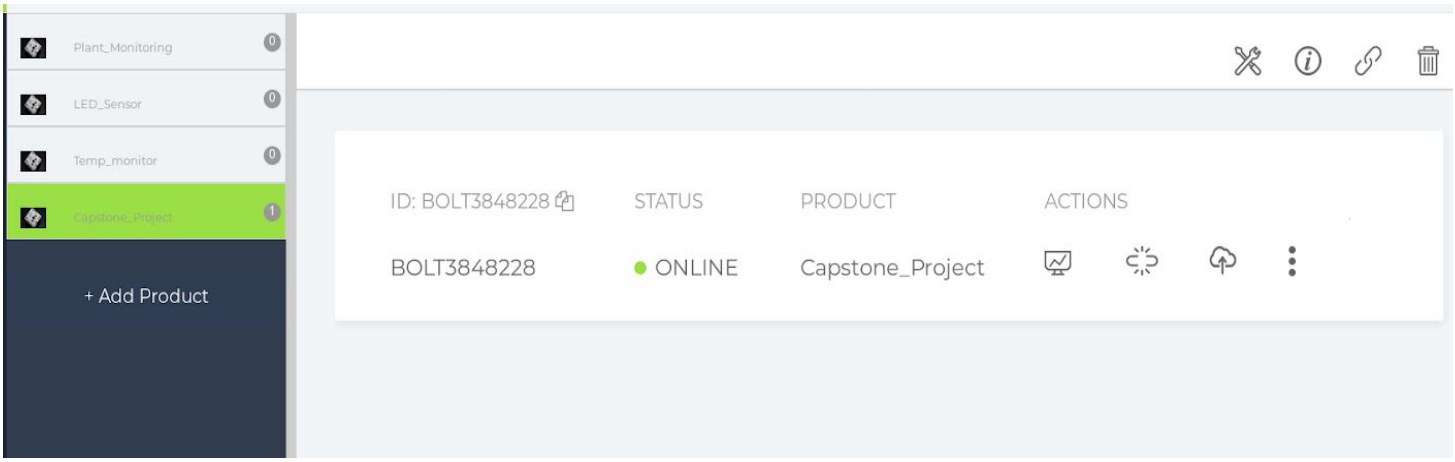
- Under the code section, write the file name, choose the file type “JS”.

- Write the following code in it:



- Save the configuration. Close this panel.

- Now, Deploy the configuration by clicking on the ‘cloud’ symbol.



- Click on ‘view this device’ symbol under ACTIONS column.

- Here, you can see the temperature behaviour of the system by the graph.

- To make this product to detect Anomalies & alert the owner, we need a Linux Distro (as it’s easy to run on this platform).

- Download Ubuntu, start using with VMware or Virtual Box [you can also Dual Boot your system with Ubuntu]

- Open the Terminal,
  - ➢sudo apt-get -y update {to update ubuntu packages}
  - ➢sudo apt install python3-pip {to install python3}

➢sudo pip3 install boltiot {to install boltiot library}

- Open https://www.twilio.com in the browser.

- Sign-up by entering the new credentials, choose product -> SMS, Language -> Python.

- Enter your phone number and verify it.

- Click on New Project, give it a name, then follow as directed.

- Open 'Learn & Build' on the left panel and click on 'get a number', then click on 'choose this number'. You will be assigned a new number with any random country code.

- On the Dashboard, you can see the 'SID' and 'AUTH TOKEN'.
- On the terminal,
  - ➢Command: >> nano configure.py
  - ➢Paste the following code in that file and replace it with actual parameters within the quotes. Then save this file and exit.

```
SID = 'You can find SID in your Twilio Dashboard'
AUTH_TOKEN = 'You can find  on your Twilio Dashboard'
FROM_NUMBER = 'This is the no. generated by Twilio. You can find this on your Twilio Dashboard'
TO_NUMBER = 'This is your number. Make sure you are adding +91 in beginning'
API_KEY = 'This is your Bolt Cloud account API key'
DEVICE_ID = 'This is the ID of your Bolt device'
```

➢Make a python file ‘product.py’ by command: >> touch product.py

➢Nano product.py

➢Write the following code in this file with an objective of getting an alert when the temperature goes beyond the threshold value or below the minimum decided value. Save it & Exit.

```
import configure
from boltiot import Sms, Bolt
import json, time

minimum_limit = 3
maximum_limit = 6


mybolt = Bolt(configure.API_KEY, configure.DEVICE_ID)
sms = Sms(configure.SID, configure.AUTH_TOKEN, configure.TO_NUMBER, configure.FROM_NUMBER)


while True:
        print ("Reading sensor value")
        response = mybolt.analogRead('A0')
        data = json.loads(response)
        print("Sensor value is: " + str(data['value']))
        try:
        sensor_value = int(data['value'])
        sensor_value = sensor_value/10.24
        if sensor_value > maximum_limit or sensor_value < minimum_limit:
        print("Making request to Twilio to send a SMS")
        response = sms.send_sms("The Current temperature sensor value is " +str(sensor_value))
        print("Response received from Twilio is: " + str(response))
        print("Status of SMS at Twilio is :" + str(response.status))
        except Exception as e:
```

```
        print ("Error occured: Below are the details")
        print (e)
        time.sleep(10)
```

➢Make a python file ‘product1.py’ by command: >> touch product1.py

➢Write the following code in this file with an objective of getting an alert when there’s an Anomaly or unexpected behaviour of the temperature suddenly using Z-Score Analysis. Save it & Exit.

```
import configure, json, time, math, statistics
from boltiot import Sms, Bolt
def compute_bounds(history_data,frame_size,factor):
        if len(history_data)<frame_size :
        return None

        if len(history_data)>frame_size :
        del history_data[0:len(history_data)-frame_size]
        Mn=statistics.mean(history_data)
        Variance=0
        for data in history_data :
        Variance += math.pow((data-Mn),2)
        Zn = factor * math.sqrt(Variance / frame_size)
        High_bound = history_data[frame_size-1]+Zn
        Low_bound = history_data[frame_size-1]-Zn
        return [High_bound,Low_bound]

mybolt = Bolt(configure.API_KEY, configure.DEVICE_ID)
sms = Sms(configure.SID, configure.AUTH_TOKEN, configure.TO_NUMBER, configure.FROM_NUMBER)
history_data=[]

while True:
        response = mybolt.analogRead('A0')
        data = json.loads(response)
        if data['success'] != 1:
        print("There was an error while retriving the data.")
        print("This is the error:"+data['value'])
        time.sleep(10)
        continue

        print ("This is the value "+data['value'])
```

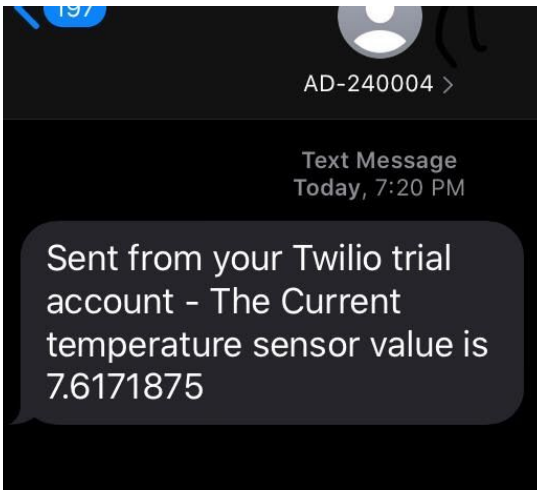
```
        sensor_value=0
        try:
        sensor_value = int(data['value'])
        except e:
        print("There was an error while parsing the response: ",e)
        continue

        bound = compute_bounds(history_data,configure.FRAME_SIZE,configure.MUL_FACTOR)
        if not bound:
        required_data_count=configure.FRAME_SIZE-len(history_data)
        print("Not enough data to compute Z-score. Need ",required_data_count," more data points")
        history_data.append(int(data['value']))
        time.sleep(10)
        continue

        try:
        if sensor_value > bound[0] :
        print ("The fridge temperature increased suddenly. Sending an SMS.")
        response = sms.send_sms("Alert: High Temperature; Someone opened the refrigerator.
Temperature Value: " +str(sensor_value/10.24) +" deg C")
        print("This is the response ",response)
        elif sensor_value < bound[1]:
        print ("The fridge temperature decreased suddenly. Sending an SMS.")
        response = sms.send_sms("Alert: Low Temperature; The temperature has decreased suddenly.
Please check your refrigerator. Temperature Value: " +str(sensor_value/10.24) +" deg C")
        print("This is the response ",response)
        history_data.append(sensor_value);
        except Exception as e:
        print ("Error",e)
        time.sleep(10)
```

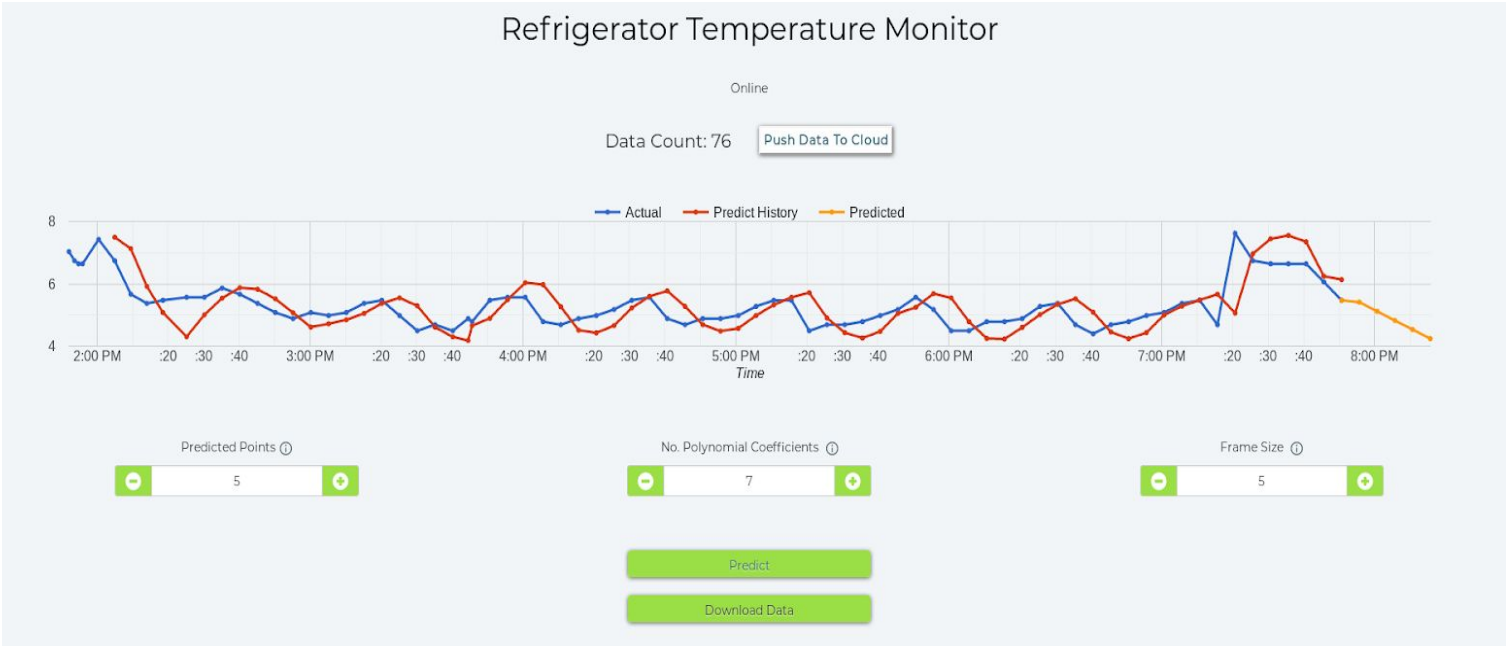
➢Run the file: >> python3 product.py [or product1.py] for the above mentioned objectives.

➢SMS Alerts:


AD-240004
Text Message
Today, 7:20 PM
Sent from your Twilio trial account - The Current temperature sensor value is 7.6171875


195
AD-340004
Text Message
Today, 1:18 PM
Sent from your Twilio trial account - Alert: High Temperature; Someone opened the refrigerator. Temperature Value: 5.95703125 deg C

❖ Graph:


Refrigerator Temperature Monitor
Online
Data Count: 76
Push Data To Cloud
Actual
Predict History
Predicted
8
6
4
2:00 PM
:20
:30
:40
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
Time
Predicted Points
5
No. Polynomial Coefficients
7
Frame Size
5
Predict
Download Data

- Up to 10 points can be predicted by clicking “Predict” button. They will be represented by ‘orange colour’ by default. Actual readings will be represented by ‘blue colour’.

- Done.

Schematics