Machine Learning for Networking

Lab Project 2

Predicting ‘Noise’ Values from a Sensor Network using ThingSpeak

In this project, we will learn how to use the basic functions of ThingSpeak to store, process and visualize the data generated from a sensor node (we will emulate the sensor network using a Matlab script).

This lab includes the following sessions:

- Session 0: SDGs
- Session 1: First contact with ThingSpeak.
- Session 2: Testing a data prediction solution.
- Session 3: Free session to complete the lab + submit report.

About ThingSpeak:
https://es.mathworks.com/help/thingspeak/getting-started-with-thingspeak.html

Check in detail how these two functions work:
https://es.mathworks.com/help/thingspeak/thingspeakread.html
https://es.mathworks.com/help/thingspeak/thingspeakwrite.html

Requirements:

- If your Matlab version is 2018b or earlier you will need to install the ThingSpeak toolbox. Otherwise it already came with your distribution. https://es.mathworks.com/matlabcentral/fileexchange/52244-thingspeak-support-toolbox

Disclaimer:

- In this subject, we don’t have any kind of relationship or cross interest with ThingSpeak, beyond we are convinced it is a good tool to understand how cloud-computing can help the development of IoT solutions.
- If anybody wants to use a different platform, that would be great and appreciated.

Grading (over 10)

- SDG case: 4 points
- Session 1: 2 points
- Session 2: 4 points

Session 0

Write a report of 1-2-3 pages (A4, 11pt, reasonable page margins) with the following contents:

1) Choose one SDG, explain why you think it is important, and how IoT solutions can help to improve it.
2) List which type of data you will need to gather from sensors. All sensors you need are commercially available or you will need to design / develop your own solutions? Feel free to use your imagination, though science fiction should be avoided.
3) Browse the web to find if there are available datasets satisfying your needs. List what you have found.
4) Explain how Data prediction techniques & ML can be used over those datasets to extract valuable information, or to develop predictive models that can be applied in daily life situations. Can you think of developing an 'app' that could use those models to improve your SDG?

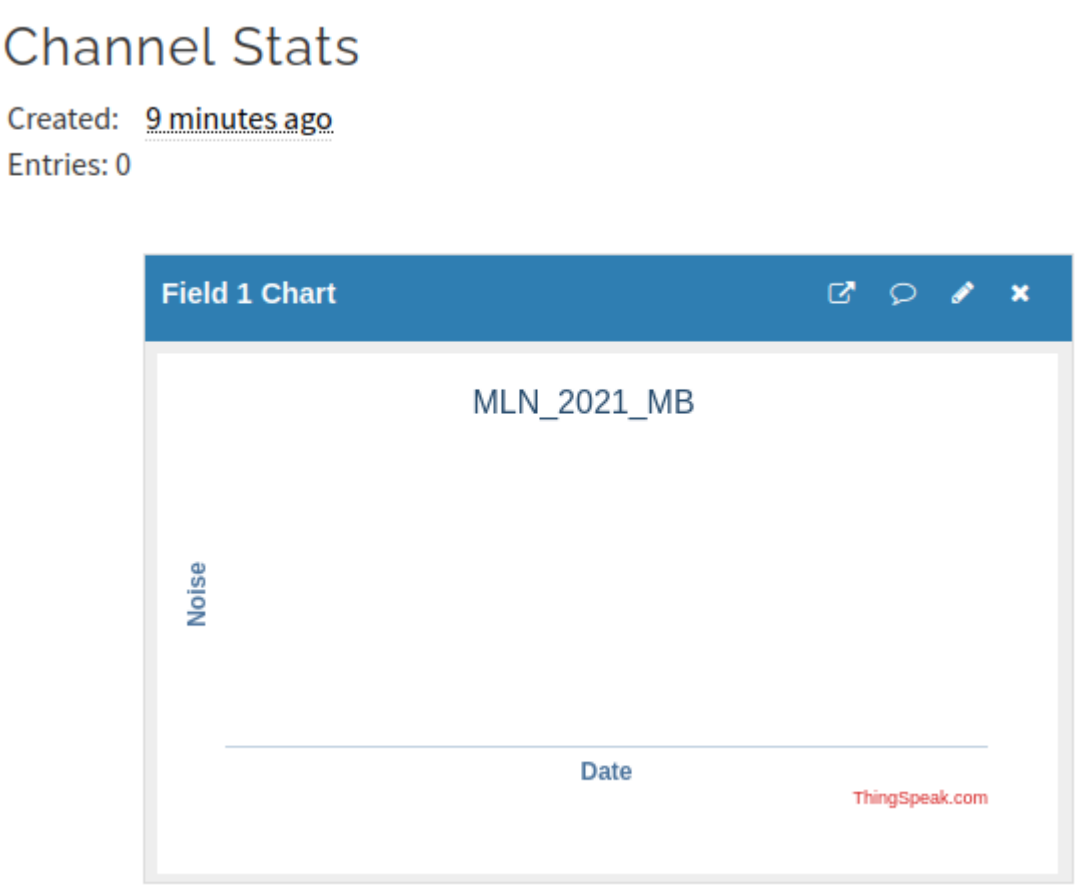
The report must be delivered along with the report of the lab. There is no template, so deliver what you think is appropriate in terms of contents, and quality.

Session 1

1) Create a new 'free' account in ThingSpeak. https://thingspeak.com/ Take a look at the conditions and limitations (e.g. num of messages you can upload per second).

2) Create a new private channel. Call it MLN_2021_xyz, where xyz are the initial letters of the names of the members of the group (if less than 3, then just xy). Configure it with a single field (it comes that way by default), and call it 'Noise'. No need to add anything else.

   After saving, take note of the channel id. For instance: **14xxxxx.** By default, you will get Field 1 chart, with no data.



3) Download the Matlab code from the ML web page: MLN_NoiseSensorEmulator.m. Examine what it does and execute it. Note that the command thingSpeakWrite is commented.

4) From your channel, go to API Keys, and obtain the Write API key, i.e., something like **HA0FXXXXXXXXXXX**

5) Update the SensorEmulator.m file with that information, i.e.:
   thingSpeakWrite(**14xxxxx**, Measure(i), 'WriteKey', '**HA0FXXXXXXXXXXX**');

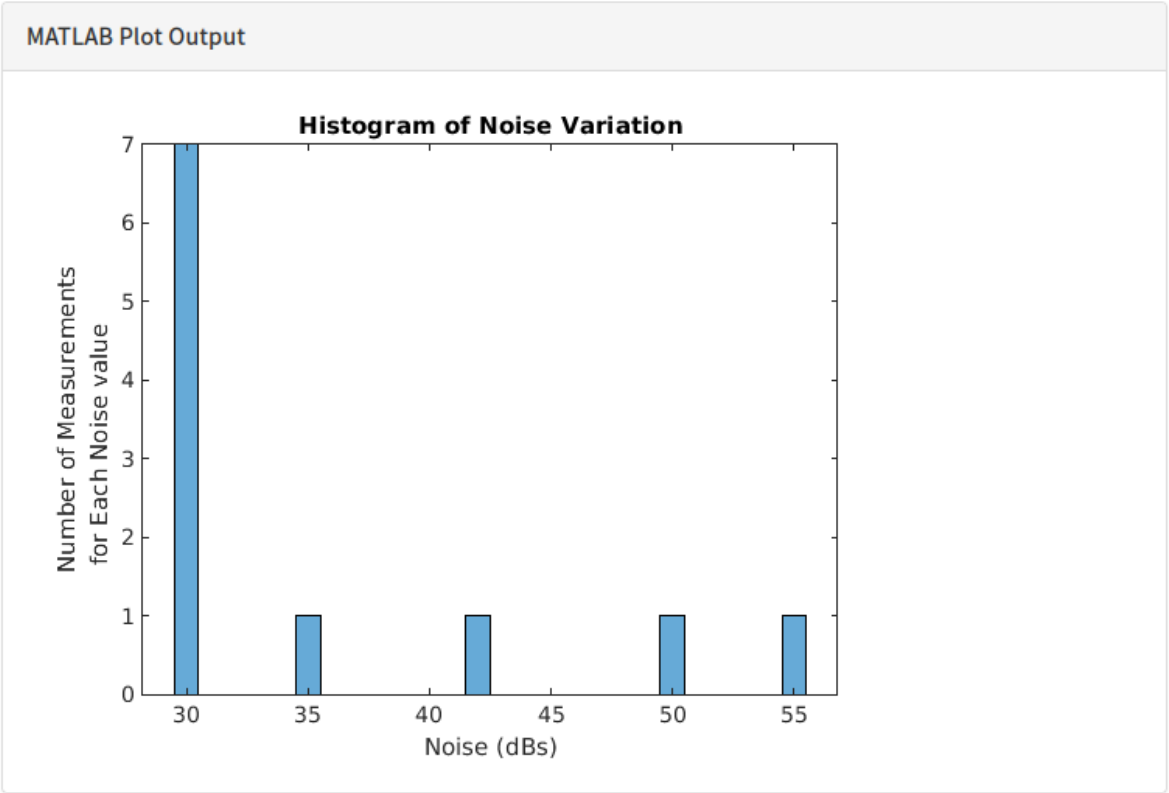
6) Uncomment that line, and execute again the MLN_NoiseSensorEmulator.m function. At the same time, go to the main page of your channel in ThingSpeak (Private View), and explain what you observe in Field 1 Chart.

7) Go to Data Import / Export, and export your data in csv format. Does it correspond to the uploaded data?

8) Go to Channel Settings, and find how to 'clear' all data. Don't do it, but keep it in mind in case you need to do it.

9) Now, we are going to add a new plot to visualize the data: a histogram. Go to APPs → Matlab Visualization. Click ‘new’ and select in examples: Use a histogram to understand variation in data.
    a) Change the title to: Noise Histogram
    b) Update the code to suit your own data! All the info you need is on the right side of the page. You have to update the readChannelID, the name ‘TemperatureFieldID’ to ‘NoiseFieldID’, set up the readAPIKey = “"; and so on… Update also the comments and labels in the histogram.

    You should be able to visualize something (more complete) like this:



10) At this point, we are going to create a function able to calculate some ‘parameters’ of the data received. For instance the mean and variance. Go to APPS → Matlab Analysis, and click ‘New’. Select from examples: “Calculate and display average humidity”, and rename it to “NoiseAnalysis”. Then, update the code accordingly. Note that the last part (after the fprintf) is, at the moment, informative.

    Note that the code from the example uses the data from the last 60 minutes. Let’s use the last 3 values instead. How to do it:
    https://es.mathworks.com/help/thingspeak/thingspeakread.html

11) To conclude today, we are going to send the results from the analysis to another channel. Create a second channel and call it ‘MLN_xyz_AnalyzedData’. Annotate its id and Write API key.

12) Go back to APPS→ Matlab Analysis, and update the NoiseAnalysis function:

```
% Replace the [] with channel ID to write data to:
writeChannelID = [];
% Enter the Write API Key between the '' below:
writeAPIKey = '';
% We are going to write only the mean
thingSpeakWrite(writeChannelID,avgNoise,'WriteKey',writeAPIKey);
```

13) Go to APPs → Time Control, and click 'New Time Control'. Update its name to "NoiseTimecontrol", set it to 'recurring', and plan its execution every 5 mins. In Code to execute, select Noise Analysis.

14) Validate that it works, by checking if the mean is correctly saved in the MLN_xyz_AnalyzedData channel.

15) That's all for today. Feel free to play with the platform, of course.

Report the steps followed, including screenshots from the plots obtained.

Session 2

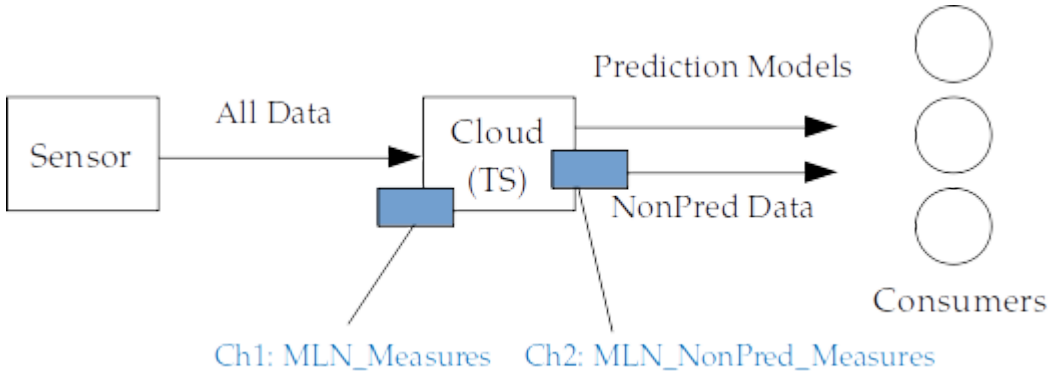


Figure. Architecture of the proposed system.

In this session we will consider the use of data prediction techniques to reduce the number of data transmitted by a sensor node.

1) Download the provided Matlab code, **MLN_NoisePredictionDataSet_TBC.m** and **MLN_NoisePredictionTest_TBC.m**.
2) Complete the function MLN_NoisePredictionDataSet.m to train a simple linear regression model (1 coefficient) using the DataSet (200 samples).
3) Implement the prediction model in MLN_NoisePredictionTest.m, and execute it. Observe how the prediction model works. Why at some point does the prediction model fail? How can that situation be solved?

Now, we are going to implement that prediction model in ThingSpeak. We will first create two new channels

- Ch1: MLN_Measures: that we will feed only with the received data from Matlab, and
- Ch2: MLN__NonPred_Measures: where we will store the measures that cannot be correctly predicted (error > 2 dBs).

The idea of the proposed system to emulate is:

1) A sensor sends always data to the cloud (ThingSpeak)
2) The cloud generates prediction models, and makes them available to the consumers (final APPs).
3) The cloud only makes available to the consumers the data that cannot be predicted in channel MLN_NonPred_Measures), from where we assume that data is sent to the consumers.

Imagine the case that the server in the cloud has to send the same data to thousands of consumers. Even if it is a single 'packet', saving such data exchange will reduce energy consumption, as well as the Internet congestion.

Investigate how the '**React**' ThingSpeak function can be used to implement such a feature. Basically, everytime we receive a new measurement in MLN_Measures, we will execute a new Matlab Analysis function (i.e., we can call it NoisePrediction) that using the two last values stored in MLN_Measures verifies that the last received value can be correctly predicted using the model and the previous value. If not, the last value stored in MLN_Measures is written in MLN__NonPred_Measures.

To feed the MLN_Measures channel, you can use the function MLN_NoiseSensorEmulator.m, from session 1.

Report the steps followed, how you have configured the React function in ThingSpeak, and the code of the NoisePrediction function you developed.

Session 3

Free session to complete the proposed work in previous sessions, and finalize the report to deliver that includes the SDG report, the session 1 report, and the session 2 report.