Machine Learning for IoT
Lab 1 – Getting Started

The following instructions describe how to setup the development environment that will be used for the labs and homework. Part of the activities will take place on a ***cloud environment***; the rest will take place on a ***local environment*** onboard of a Raspberry Pi 4 (RPI) device.

HowTo 1: Cloud Environment Setup

Computing: Deepnote
The cloud development environment is hosted on Deepnote (deepnote.com), a collaborative platform where developers can create, run, and share Jupyter notebooks.
Join the *ML4IoT 25 LABs* workspace by clicking on the invitation link received on your student mailbox (sXXXXXX@studenti.polito.it). Sign in at Deepnote using the same mail address where you receive the invitation (other addresses will not work).

The workspace is organized in *projects*. Each project is a collection of Jupyter notebooks and files, that all run in the same environment.
The *ML4IoT 25 LABs* workspace contains the following shared projects:

- Material: (read-only) Notebooks, code, and files with all the reference material.
- Live Coding: (read-only) Notebooks created during the lectures/labs.
- GroupXX: projects associated to a student team, where students belonging to the team can create, edit, and run the code needed to solve the labs' exercises.

Navigate to the *Material* project and follow the instructions in the notebook *0. Deepnote HowTo* to setup your personal project for individual development.

Storage: Redis Cloud
The cloud storage is hosted on Redis Cloud. Redis is a key-value database, i.e., each entry of the database consists of a simple string (the key) that is unique and a data field (the value).
Redis integrates a time-series data structure, where the timestamp is the key and the data is the value. Moreover, Redis offers many features to query and post-process time-series data efficiently.

Sign up for a FREE account on https://redis.io/try-free/
Then, **create a database** following these instructions:

1. Sign in at https://redis.io/try-free/
2. From the left sidebar, click *Daabases*, then click the *New database* button in the center of the page
3. In the *Database details* section, click on "30 MB free". Keep other settings at their default values.
4. Click the *Create Database* button at the bottom of the page.
5. In the *General* section, find the *Public endpoint.*
6. In the *Security* section, find the database *password*.
7. The endpoint and password will be used later to connect to the database.

HowTo 2: Local Environment Setup

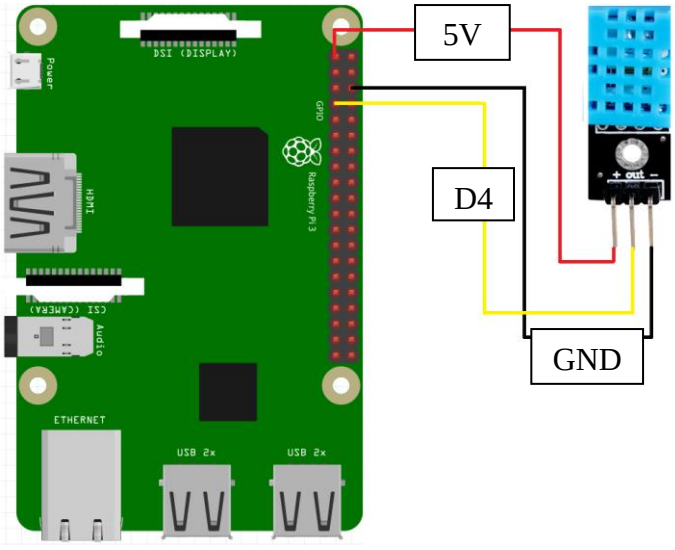
VS Code Setup

Download & Install Visual Studio Code (VS Code) on your laptop. VS Code will be used for connecting to the RPI via SSH.

1. Download the installer from https://code.visualstudio.com/Download
2. Run the installer.

RPI Kit Setup

1. Remove the SD card from the SD adapter and insert the SD card into the RPI. The SD card mounts a development environment running on Raspbian OS, including Python 3.11 and all necessary Python packages for the lab exercises (*tensorflow*, *scipy*, *sounddevice*, *adafruit_dht*, *redis*, *paho-mqtt*, *cherrypy*).
2. Connect the RPI to your laptop using the Ethernet cable.
3. Connect the USB microphone to the RPI
4. Connect the DHT-11 sensor to the RPI (refer to the schematic below):
    a. Connect the **+** pin of DHT-11 to the **5V** pin of RPI (1st pin from the top-left)
    b. Connect the **OUT** pin of DHT-11 to the **D4** pin of RPI (4th pin from the top-left)
    c. Connect the **–** pin of DHT-11 to the **GND** pin of RPI (3rd pin from the top-right)
5. Connect the RPI to the power supply (wait approx. 2 minutes for the startup)



SSH Connection Laptop <-> RPI

1. Open VS Code from your laptop.
2. Go to *View, Command Palette…*
3. Enter: *Remote-SSH: Connect to Host*
4. Enter: `ml4iot@raspberrypi.local`
5. Enter the default password: `pi4iot`

RPI Wi-Fi Connection

To connect the RPI to the internet, you can configure Wi-Fi connectivity to the eduroam network:

1. In VS Code, open a terminal (*Terminal, New Terminal)*.
2. In the terminal, run the following command to open the network manager:

```
sudo nmtui
```

3. Go to *Edit a connection*
4. Go to *Add*
5. Go to *Wi-Fi*
6. Set *Profile name* to *eduroam*
7. Set *SSID* to *eduroam*
8. Set *Security* to *WPA & WPA2 Enterprise*
9. Set *Authentication* to *PEAP*
10. Set *Anonymous identity* to your student email address (sXXXXXX@studenti.polito.it)
11. Set *PEAP version* to *Automatic*
12. Set *Inner authentication* to *MSCHAPv2*
13. Enter your username (sXXXXXX@studenti.polito.it) and password
14. Go to *OK*
15. Press the ESC button from your keyboard
16. Go to *Activate a connection* and ensure that the *eduroam* connection is active (an * next to the name indicates that the connection is active)
17. Press ESC multiple times to close the network manager
18. Check internet connectivity by running the following command in the terminal:

```
ping google.com
```

19. Press **CTRL+C** to stop the ping command.

You can connect to your personal Wi-Fi (while working at home) or mobile hotspot (needed for some exercises while working in class) following these instructions:

1. In VS Code, open a terminal (*Terminal, New Terminal)*.
2. In the terminal, run the following command to open the network manager:

```
sudo nmtui
```

3. Go to *Edit a connection*
4. Go to *Add*
5. Go to *Wi-Fi*
6. Set *Profile name* to the name of your personal connection
7. Set *SSID* to the name of your personal connection
8. Set *Security* to *WPA & WPA2 Personal*
9. Enter the password in the *Password* field
10. Go to *OK.* Activate the connection and exit from the network manager

Create a Working Directory

1. In VS Code, open a terminal (*Terminal, New Terminal)*.
2. Create a new directory by running the following command in the terminale:

```
mkdir workdir
```

3. Open the newly created directory in VS Code by running the following commands in the terminal:

```
cd workdir
code .
```

(If needed) HowTo 3: Restore the RPI

If your development environment on the RPI becomes corrupted, you can restore the SD card.
Note: This process will cause irreversible data loss, so ensure you manually back up your code before proceeding.

1. Download & install Balena Etcher from https://etcher.balena.io/#download-etcher
2. Download the *rpi.img* file from the *Labs/lab-1-getting-started* folder on the *Portale della Didattica*
3. Shutdown the RPI by disconnecting the power supply
4. Remove the SD card from the RPI
5. Insert the SD card into the SD adapter
6. Insert the SD adapter (with the SD card) into your laptop
7. Open Balena Etcher
8. Click *Flash from file*
9. Select the *rpi.img* file
10. Click *Select target* and select *SDHC SCSI Disk Device*
11. Click *Flash!*

Exercise 1: Record Audio with the RPI & USB Microphone

In VS Code, write a Python script to record audio data with the RPI and the USB microphone.

a. In the *workdir* folder, create a new Python file (e.g., *lab1_ex1.py*) and write a script that uses the `sounddevice` package to record audio data. Stop the recording when the Q key is pressed.

   Suggestion: check the documentation at:
   https://python-sounddevice.readthedocs.io/en/0.4.5/api/streams.html

b. Modify the script to store the audio data every second. Use the stream *callback* function to store data in parallel while recording. Use the *scipy.io.wavfile.write* function to store the audio data. Use the timestamp of the recording as the filename.

   Suggestion: check the documentation at
   https://docs.scipy.org/doc/scipy/reference/generated/scipy.io.wavfile.write.html

c. Modify the script to allow the user to disable/enable the audio storage by pressing the P key.

d. Modify the script to introduce the following parameters as command-line arguments using the `argparse` package:
   - Bit depth (str): int16 or int32
   - Sampling Rate in Hertz (int)
   - Recording duration in seconds (int)

e. Measure the output .wav size (in KB) using the *os.path.getsize* method from the *os* package

f. Run the script with different parameter values. Try different combinations of bit depths (int16 and int32), sampling rates (44.1 kHz and 48 kHz), and duration (1 s and 2 s).

g. Listen to the recorded audio and evaluate the audio quality in the different cases. Then, fill out the following table and comment the results.

h. Finally, define an equation that computes the .wav file size as a function of bit depth, sampling rate, and recording duration. Validate the equation using the data collected in the previous step.

Bit Depth	**Sampling Rate (kHz)**	**Duration (s)**	**Size (KB)**
int16	44.1	1	
int16	44.1	2	
int16	48	1	
int16	48	2	
int32	44.1	1	
int32	44.1	2	
int32	48	1	
int32	48	2	

Exercise 2: Monitor T/H with Python

In VS Code, write a Python script to monitor temperature and humidity (T/H) with the DHT-11 sensor.

a. Create a new Python file (e.g., *lab1_ex2.py*) and write a script that uses the `adafruit_dht` module to read temperature and humidity from the DHT-11 sensor.

b. Every 2 seconds, print the T/H values using the following format:
year-month-day hour:minute:second.microseconds - *mac_address:*temperature = *T*
year-month-day hour:minute:second.microseconds - *mac_address:*humidity = *H*

where *mac_address* is the MAC address of your network card, *T* is the temperature value in degree Celsius, and *H* is the humidity value in %RH.

Example:

```
2022-10-01 19:21:51.699254 - 0xf0b61e0bfe09:temperature = 21
2022-10-01 19:21:51.699254 - 0xf0b61e0bfe09:humidity = 60
2022-10-01 19:21:53.701326 - 0xf0b61e0bfe09:temperature = 22
2022-10-01 19:21:53.701326 - 0xf0b61e0bfe09:humidity = 59
```

c. Modify the script to store T/H values to Redis Cloud. For storage, define two Redis TimeSeries, called *mac_address:*temperature and *mac_address:*humidity, respectively (e.g., *0xf0b61e0bfe09:temperature* and *0xf0b61e0bfe09:humidity*)

d. On Deepnote, create a new notebook to read the T/H values from Redis and visualize them.