Artificial Intelligence and Data Analytics for Engineers

Exercise 4

Introduction

Solving exercises is not mandatory, and exercises will not be graded or corrected by the lecture team. However, we strongly advise you to do as many exercises as possible to prepare for the final exam. Some of the tasks will be discussed with a presentation of the solutions during the exercise session. If you want to do exercises at home, we suggest installing Anaconda (https://www.anaconda.com/distribution/ - you would need to download the Python 3.7 version).

In case you have any questions, feel free to send us an e-Mail to: aidae@ima-ifu.rwth-aachen.de.

Task 0) Setting up your Python environment

Because you created a Python environment the first week, since then, we only need to enable the environment from now on. This needs to be done **every time** you open a new terminal, for instance after a system restart or if you accidentally closed the Terminal. Open a terminal and enter:

```
source activate aidae
```

Now you are ready to start programming in Python.

Task 1) SQL databases.

The goal of this task is to get to know the basics of how to interact with SQL databases. Normally the SQL databases are hosted in dedicated servers, which allows multiple users and applications to access their content. For this task we will use a lightweight disk-based database that doesn't require a separate server process named SQLite.

a) Use the library sqlite3 and create a connector to the database “sensor_readings.db” (this database is given as a separate file). List all tables existing in the database.
b) Using the connection of the section a, get the data of the sensors in the table “sensors”.
c) Using the connection of the section a, explore the data of the table “measurements”. How many measurements does each sensor has? Which sensor has more measurements? Which are the maximum and minimum measurements of the acceleration sensors?

Continue with this task after finishing Task 3

d) In task 3 a simulated sensor was created. Add the description of said sensor to the table “sensors” and 200 data points to the table “measurements”.
e) Using the client connection from task 3 section c, generate data for one minute, send it to the mqtt broker, receive it and save it to the table “measurements” of our database.

Task 2) NoSQL database.

The goal of this task is to get to know the basics of how to interact with NoSQL databases. Each NoSQL database has a querying language adapted to the type of objects that it supports. In this exercise we will use the database tinydb, a disk-based document oriented database.

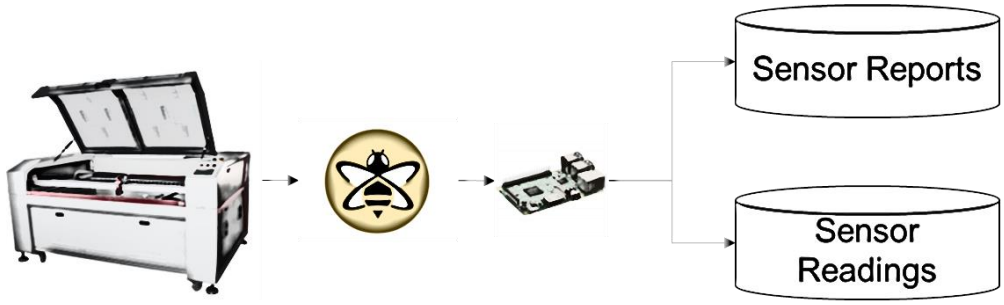
a) Use the library tinydb and create a connector to the database “sensor_reports.json” (this database is given as a separate file). Which type of data is contained here (print 5 documents)? How does this data relate to the SQL database?
b) Using the connection of the section a, select the magnetometer z sensor on the database and explore the data related to it. How many anomalies does it have? What can we infer from the high and anomaly reports?
c) Using the connection of the section a, explore the data of the reports where the state is “anomaly”.

Continue with this task after finishing Task 3

d) Using the client connection from task 3 section c, generate data for two minutes, send it to the MQTT broker, receive it and save an “high-threshold” event when the sensor value is more than 5.

Task 3) MQTT

In this task, we will explore the usage of MQTT brokers for data integration. This example is based on a real world application in which a CNC machine sends the data of its sensors to a private MQTT broker to be analyzed and separated into anomaly states and normal behaviors. See image below.



You will connect to the public MQTT broker HiveMQ. The goal of this task is to get to know the basics of how to interact with MQTT brokers for data transport and integration. This will be required in the future for the setup of data integration pipelines and data acquisition. All the subtasks will require you to load the paho-mqtt library to connect with the MQTT broker.

a) In the first task we will create a simulated sensor. This sensor will generate a sinusoidal signal of amplitude 5 with a noise of variance 0.2. The implementation of this sensor must be done in a class named SinSensor, with the properties “amplitude”, “variance”. To obtain a measurement implement the method getMeasurement as a function of a time “t” passed as a parameter.
b) Once the simulated sensor has been implemented, we must send it to a topic in the public MQTT server (broker.hivemq.com). Create a while loop that each 0.2 seconds sends a datapoint to the topic “aidae/[matriculation number]/SinSensor”. Remember to use the library paho-mqtt.
c) After the data has been sent to the public MQTT broker, use the library paho-mqtt to subscribe to the topic of the SinSensor and acquire the sent data. Remember that the Simulated sensor while loop must be running and sending data in order to be able to acquire it from the mqtt broker. Use this connection to acquire the data and print it once it arrives.
d) Using the client connection from section c, acquire data for one minute, add it to a dataframe and plot it once the data acquisition has finished.
e) (Optional). In the android play store, there is an app named “Sensor Node Free”. This app allows the acquisition of the data from a smartphones sensors, as well as streaming this data to an mqtt broker.

Use this application, and the previously acquired knowledge to gather data from one of the sensors of your smartphone (temperature recommended) and save it to a csv file.