On this page, we explain how to install a local environment to run and modify the content of the online course QuaXP.

- Official link to QuaXP: https://www3.tuhh.de/sts/hoou/data-quality-explored/
- Link to the Gitlab repository: https://collaborating.tuhh.de/hoou-tuhh/projekte/quaxp-gruppe/quaxp-data-quality-explored

The course is implemented with Jupyter Book. We first need to install Jupyter Book locally to be able to modify the content.

On this page:

Run the book locally

Install Jupyter if you don’t have it already installed: https://jupyter.org/install.html

Install Jupyter Book

Official page: https://jupyterbook.org/intro.html

```
pip install -U jupyter-book
```

Clone the repository

Link to the repository:
https://collaborating.tuhh.de/hoou-tuhh/projekte/quaxp-gruppe/quaxp-data-quality-explored

```
git clone https://collaborating.tuhh.de/hoou-tuhh/projekte/quaxp-gruppe/quaxp-data-quality-explored.git
```

Run the book

Open an Anaconda prompt and navigate to the folder you previously cloned.
Build the book with this command:

```
jupyter-book build .
```

If there are errors, they will be printed in the prompt. If there are no errors, the prompt indicates that the book is visible locally, using a browser, by opening this page: `_build\html\index.html`. It also gives the link to directly type in the browser bar.

Modify the content of the book

Each page of the book is coded into a single file. The pages can be of two types:

- simple Markdown page, if the page contains only text and media (extension .md),
- Jupyter Notebook (runs with Python), if the page contains some interactive code cells (extension .ipynb).

To start modifying or creating a page, start a Jupyter server:

```
jupyter notebook
```

A page will open in a browser with a tree. Navigate to the folder of the course, and open the file you want to modify by clicking on it, or create a new file.

After the changes are done, save the file and build the book again to visualize your changes locally.

Once the changes are satisfying, push them to a new branch on Gitlab.

There is a pipeline that builds and tests the book after a push, so make sure that your push was successful.

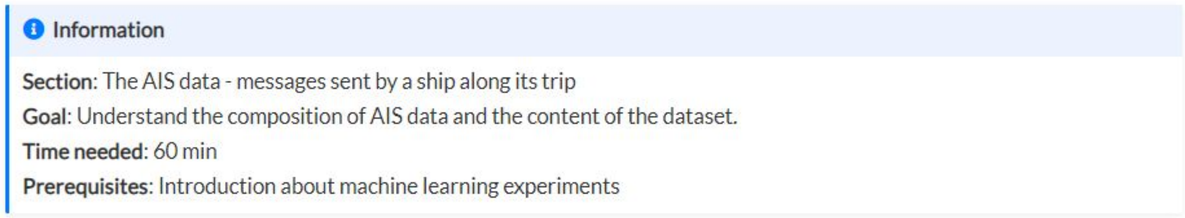
About the online version of the book

The Gitlab repository is mirrored to a Github repository, necessary to have the book online. A mirroring means that all the changes that we do on the Gitlab repository will be transferred to the Github repository, so no access to the Github repository is needed.

Some features of the book

The information panel

Each page starts with an information panel, grouping information about the page itself:



Information

Section: The AIS data - messages sent by a ship along its trip
Goal: Understand the composition of AIS data and the content of the dataset.
Time needed: 60 min
Prerequisites: Introduction about machine learning experiments

To create this panel, create a Markdown cell in the Jupyter Notebook file of the page, and fill it with this:

~~~
```{admonition} Information
__Section__: Title of the page
__Goal__: Goal of the page
__Time needed__: time needed to study the page
__Prerequisites__: prerequisite needed
```
~~~

Hint: in MD, end each line with two spaces to create a line break. Otherwise, each line will appear on the same line on the final visual.

Hidden code cells

To create the separation between the beginner and the advanced content, we use hidden cells. The advanced content is originally hidden and can be displayed by clicking on a “+” button, which appears on the book, in the text.

To hide code cells with Jupyter Notebook, we need access to the metadata of the cells. In the Jupyter Notebook file, click on “View > Cell toolbar > Edit metadata”.
The style of the page slightly changed. Now, it is possible to click on a button “Edit metadata” for each cell in the file.

In a code cell, we can choose to hide the input (the code), the output, or both. This is done by changing the following tags: `"hide-input"` and `"hide-output"`. Put both tags to hide both. Usually, every code cell's input should be hidden, as we do not want to force the beginner learner to see a code cell. The output is not hidden if it is explained for the beginner level, but it is hidden if we have created a different cell for the beginner level (for example, an interactive cell).

Hidden text cells

To hide text cells (Markdown cells), we cannot use the metadata. We have to hard code it by using this frame:

~~~
```{toggle} Advanced level
Write here the content of the text cell to hide.
```
~~~

This is used for explanations about the code (libraries, functions, etc…) and more generally explanations that are only useful for the advanced level.

Custom functions

Sometimes, we want to use some functions that we created, but to keep it simple for the learner, we do not want to display those functions in the main pages of the book. Usually, for transparency, we explain those functions in a page of the appendix.

Those functions are simply put in the file called "`x-functions.ipynb`", with `x` being the number of the chapter in which the functions are used.
Because of that, we need to run the file "`x-functions.ipynb`" at the beginning of each page where the functions are used. Usually, this is done by adding this line of code:

```
%run 1-functions.ipynb
```

in the code cell where we import the library Pandas and the data used on the page (usually, the first code cell of the page).

Quizzes

We try to have a quiz for the learner to answer at the end of each page of the book. Those quizzes are developed using H5P. They are hosted on the blog of the HOOU (https://blog.hoou.de/wp-admin/edit.php), so to create a new one, we need access to a HOOU account. Once of the wordpress interface of the HOOU, click on "H5P-Inhalt" on the left menu, create an H5P element and integrate it to the book in a code cell like this:

```
from IPython.display import IFrame
IFrame("https://blog.hoou.de/wp-admin/admin-ajax.php?action=h5p_embed&id=i
d", "760", "376")
```

Change the id of the element and possibly the size. to fit to the content. Do not forget to hide the input of this code cell, as we are only interested in the IFrame created in the output.

Note

Using Jupyter Book takes away some freedom in the design of the book. Jupyter Book is still currently under development and some bugs can be present, that we cannot do much about. One example is the interactive widgets, which are based on the library ipywidgets.
It is however possible to think about other extended ways to have the behaviour we would like to have.