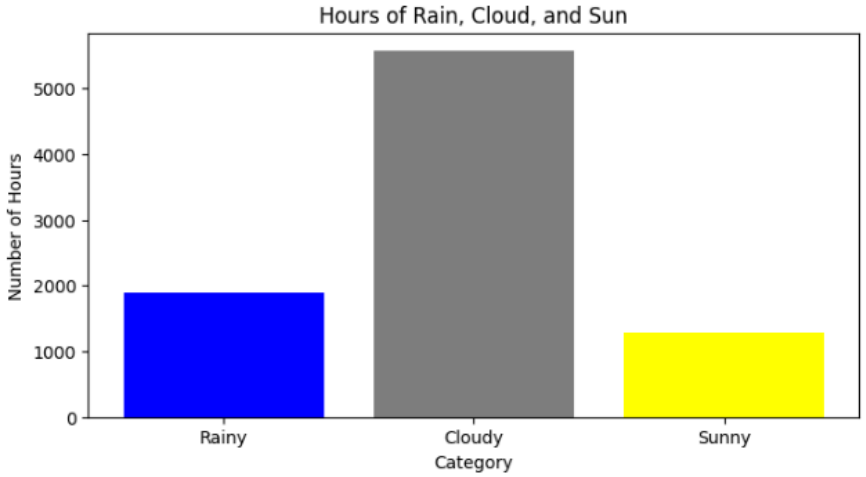
Summary - Data4Good

In our project, we've used the DWD **Weather Data** East Germany API to access accurate and up-to-date meteorological data from the eastern region of Germany. This API offers detailed information on weather conditions such as temperature, precipitation, wind speed, and more. By integrating this API into our project, we were able to obtain reliable data to enhance the accuracy and functionality of our applications or analyses related to weather in this specific area.

Observations:
From the entire dataset, we will only need a few variables from 2021, as the dataset you provided us with was from 2021 and it would make sense to use that.

When analyzing the 2021 data, we observed that we have a few variables with values. Therefore, we will use those that make the most sense for our study. In our case, we will use 'sunshine' and 'precipitation'.
The entire process of obtaining meaningful data is explained in the code (Data4Good_1.ipynb). In the end, we observe the following evaluations.



On the other hand, (Dataprepatation.ipynb) we merged two datasets, removed uninteresting variables, and focused on a subset of the input data. In Particular, we analyzed **accidents** that occurred near the sensors in Berlin.

Once we merge all the CSV files, we can start analyzing them. On one hand, we've developed a **Deep Learning model** where we employ an **MLP** to predict accident probabilities given a date and the weather. We're using the sigmoid activation function since we're interested in values between 0 and 1.

To build this **neural network**, we encountered many dimensionality issues. Typically, inputs have the same dimensions, but in our case, they do not. To solve this we have used zero-padding, as we are working with binary variables.

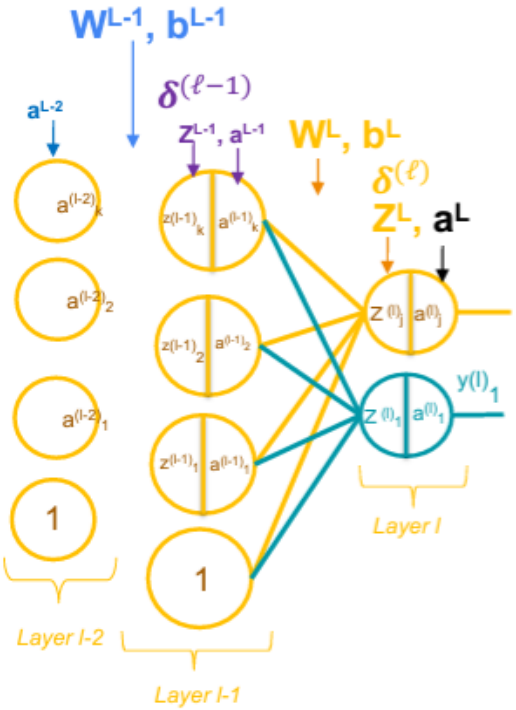
To train the data, we need to perform forward and backpropagation, where we then calculate the loss function. We've used Binary Cross Entropy to classify the labels in a binary manner. However, we've used **PyTorch**, which is a Python library that helps us perform calculations very efficiently and internally computes backpropagation, so we don't have to worry about it. This makes it very useful as neural networks start to increase in complexity.

But basically, the feedforward is:

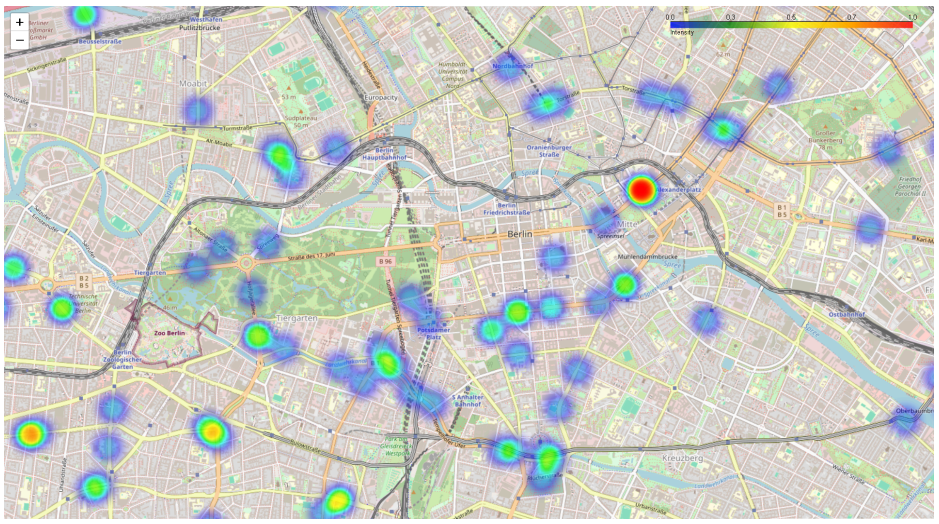
$$h_k(x) = \sigma\left(\sum_{j=1}^{M} w^{(2)}_{kj} \times \left(\sigma\left(\sum_{i=1}^{D} w^{(1)}_{ji} \times x_i + w^{(1)}_{j0} \right) \right) + w^{(2)}_{k0} \right)$$

$z^{(2)}_k$ $z^{(1)}_j$ $a^{(1)}_j$ $a^{(2)}_k$

The case of backpropagation is much more complicated. However, we can save ourselves the calculations thanks to PyTorch, but it's interesting to understand everything that happens behind the scenes. These would be the general derivatives that should be computed



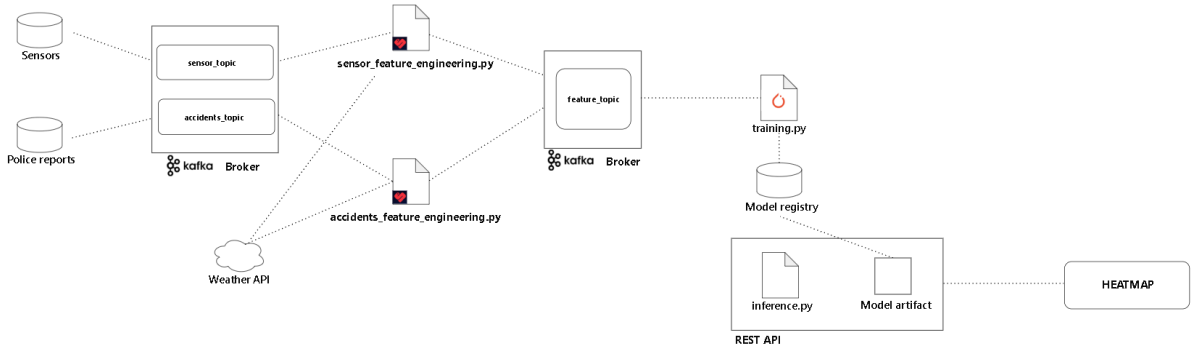
Once we have all the probabilities, we can create various applications. In our case, we have developed a Python script using the library folium that takes the accident probabilities which are linked to the coordinates of Berlin with latitude and longitude, so we can plot the probabilities on the map. This way we establish a relationship between accident intensity and frequency.



Thanks to folium, we've created an interactive graphical interface that allows zooming to view areas with a higher probability of accidents at that moment. With this, we'll be able to redirect people/drivers when there's a certain risk of accidents.

To be usable in real life, we have created an architecture to make it a real-time system, meaning that data is sent periodically to be executed automatically. This ensures that citizens have up-to-date data and allows us to redirect traffic to reduce the likelihood of accidents. To do this, the structure is based on three sources of information: **Police**, **Sensors**, and **Weather**. This is sent to a **Kafka Data Broker** which will send it to a node to execute Python scripts, the output will go to another Data Broker which will send the result that will be the input for the deep learning model, then this would be sent to another Kafka Data Broker, which will send the data to another Python script to create a heatmap of accident probabilities in Berlin which will serve to improve the life of people in Berlin.

Finally, we can see that this system allows us to make it scalable to large scales.



OBS: ALL FILES AND DOCUMENTS ARE IN THE GITHUB:

https://github.com/iv97n/Data4Good