AccuMop

System for monitoring the accumulation of insurance risks. The entire process consists of data preparation (SAS), data manipulation (Python), and subsequent visualization (Streamlit).

Idea

This approach to monitoring risk accumulation involves dividing the monitored area into a grid of equally sized territories. In our case, we monitor risk accumulation in the Czech Republic, which we divide into a grid of squares with an approximate area of 1 km^2. In each square of the resulting matrix, we obtain a value between 0 and 1, representing the degree of risk accumulation in that area.

Data Preparation - SAS

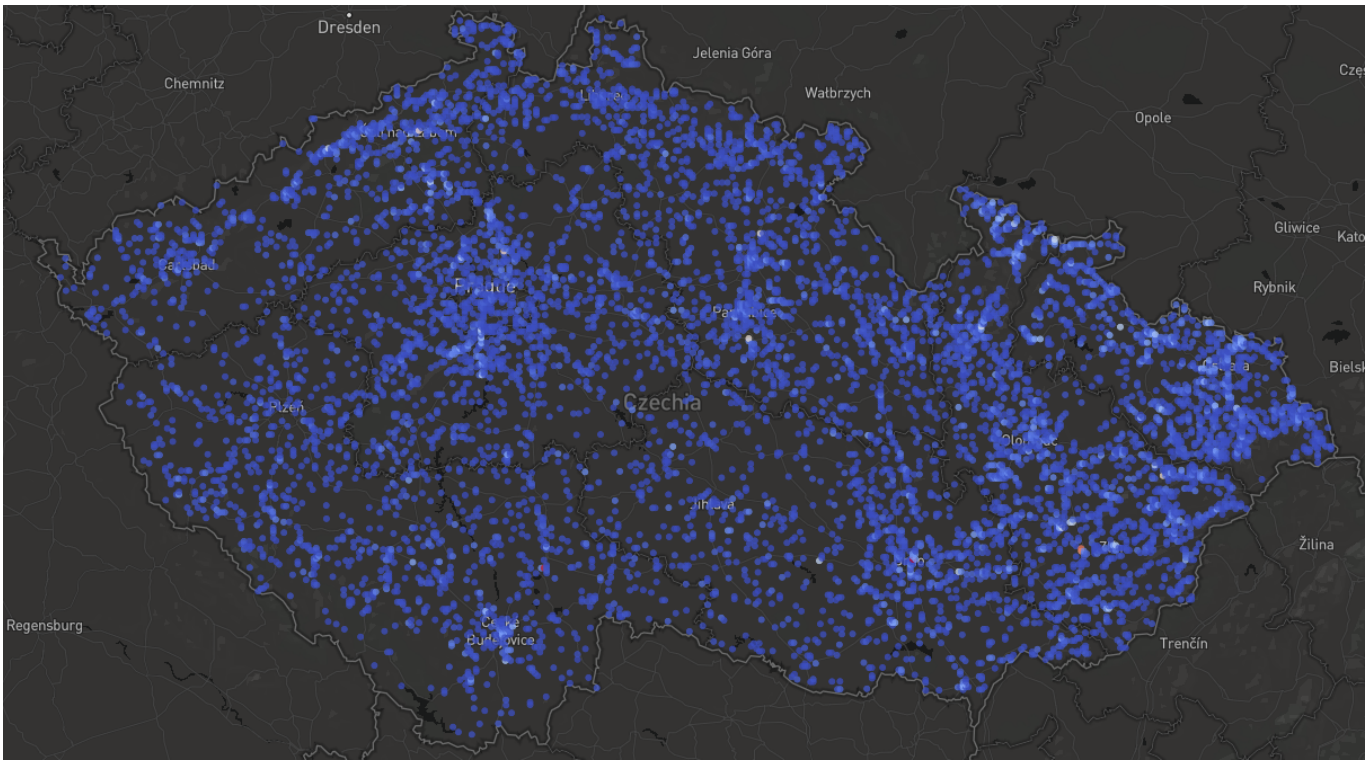
The key value for the resulting measure of risk accumulation is the expected damage for each risk on individual objects. Therefore, we start with a table containing the `longitude` and `latitude` values for each object on the map, along with the `expected damage` value and the `risk zone`.

Next, we define a square grid over the area we are monitoring, and we assign each location from the original table to the corresponding square.

For each risk, we calculate the sum of expected damages for that risk in each square across all insured objects (in our case, buildings) falling within the risk zone allowing for significant damage from that risk. This way, we obtain a matrix of values that, when multiplied by the maximum value of the matrix, fall within the range of 0 to 1.

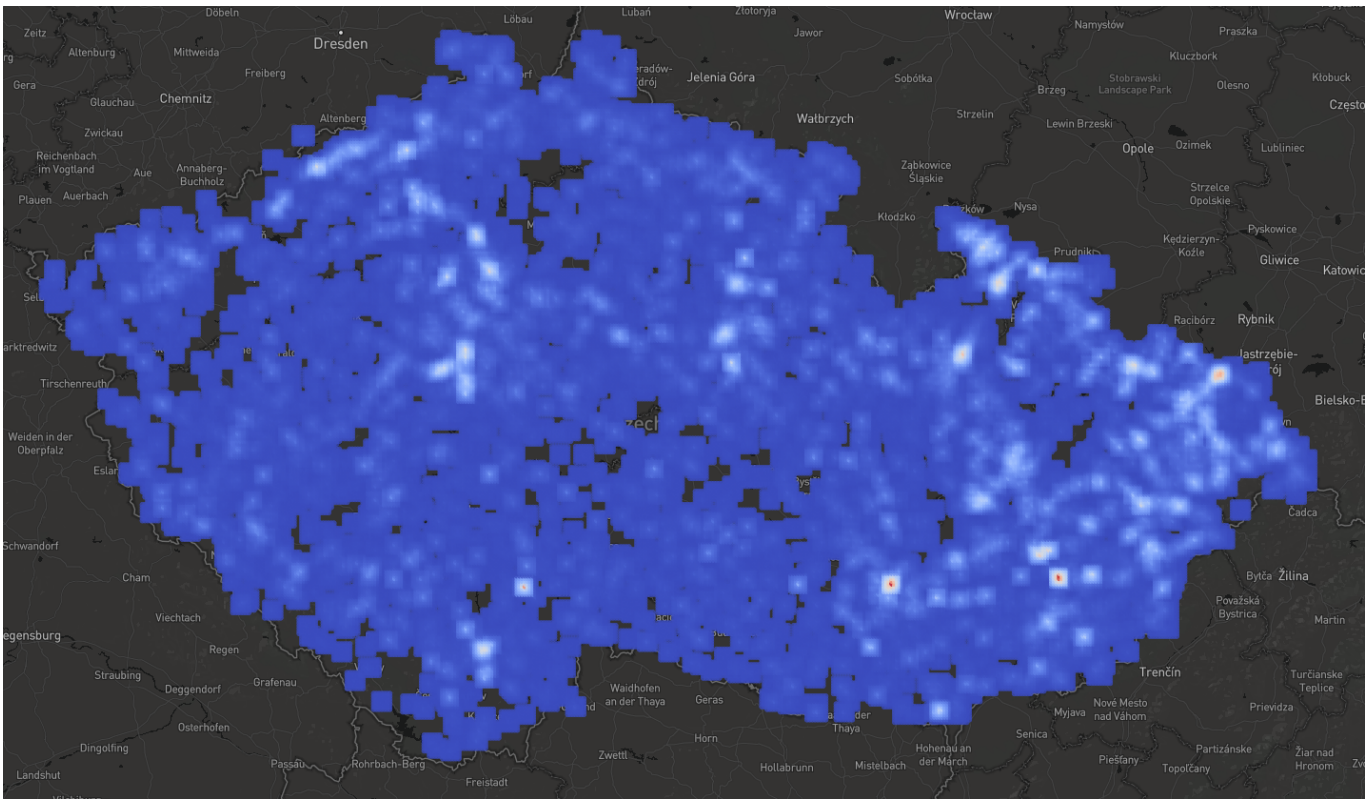
Data Manipulation and Visualization - Python, Streamlit

The acquired data can now be displayed directly on the map, with redder points in the displayed matrix corresponding to values closer to 1, indicating areas with greater risk accumulation.



The issue is that square areas may arise for which we do not have sufficient data, i.e., uninsured objects, yet the map shows significant risk accumulation in surrounding areas. This problem is addressed using a modified algorithm used for image blurring.

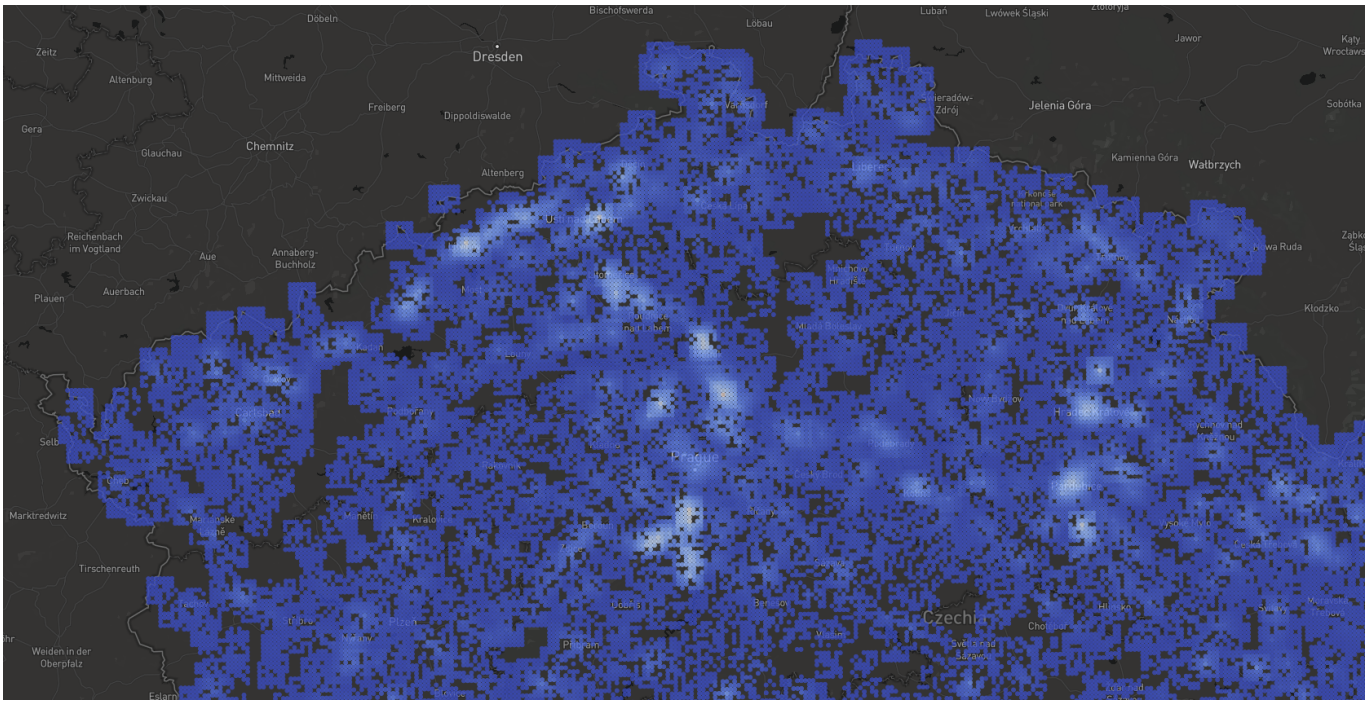
Therefore, we want to obtain a new matrix of values based on the original one, but considering the values of surrounding squares for each square in the matrix. However, the influence of distant areas should diminish; beyond a certain distance, the mutual influence of individual areas is not considered (the default value is 3 km).



If s_n represents the value of a square in the new matrix, s_o is the value of a square in the old matrix, d is the Euclidean distance between squares, and p is an optional value (default is 1), the formula for calculating values in the new matrix can be expressed as follows:

$$s_{n} = \sum\frac{s_{o}}{\left(1+d \right)^{p}}$$

After this process, there may still be an issue where some squares contain nonzero or even high values, even though all buildings fall within a risk-free zone (for example, near a river but on a hill). To address this, we convert all such values in the new matrix to 0 and then divide all matrix values by the maximum value of the matrix again, ensuring that the maximum value is 1.



The result can be displayed on the map again and utilized in further processes.