Section III
1. Who are you (mini-bio) and what do you do professionally?

We are a team of Moscow Hares from Russia consisting of two people.
Daniil is directly responsible for the development of models.
Anna is the head of our team, who leads us forward, allocates time and resources.
And together we can outline any task and solve it.

2. What motivated you to compete in this challenge?

We are fans of various competitions and hackathons. Recently, computer vision and working with satellite images have often come across problems. When we saw this competition, we immediately decided to try it. But we could not imagine that we would win until the last moment.

3. High level summary of your approach: what did you do and why?

First we tried to make a single Unet model, as this is a classic approach for segmentation. But it brought outstanding results. After that, we studied the articles in which we solved this problem. Having found out that flooding was often calculated using mathematical formulas, the idea came up to make a pixel-by-pixel classification by translating images into tables. After that, we noticed that both approaches do not fill in the flooding rather than predict the excess. Therefore, we decided not to take the average of the two approaches, but to combine them. It was also important to use the Nasadem band. In the aggregate, this gave the result.

4. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model

First, using Unet with the EfficientNetB0/B4 backbone
```
sm.Unet('efficientnetb0')
```

Second, assembling a dataset for pixel-by-pixel prediction
```
temp = pd.DataFrame()
temp['vh'] = vv.flatten()
temp['vv'] = vh.flatten()
temp['nasadem'] = nasadem.flatten()
temp['jrc_gsw_change'] = jrc_gsw_change.flatten()
temp['jrc_gsw_occurrence'] = jrc_gsw_occurrence.flatten()
temp['jrc_gsw_extent'] = jrc_gsw_extent.flatten()
temp['jrc_gsw_recurrence'] = jrc_gsw_recurrence.flatten()
temp['jrc_gsw_seasonality'] = jrc_gsw_seasonality.flatten()
temp['jrc_gsw_transitions'] = jrc_gsw_transitions.flatten()
```

```

Third, combining predictions to the maximum
```
all_pred = np.round(np.max([unet_pred, cat_pred], axis=0))
```

5. Please provide the machine specs and time you used to run your model.
● CPU (model): Apple M1
● GPU (model or N/A): Apple M1
● Memory (GB): SSD 256Gb
● OS: Mac OS X
● Train duration: training one Unet model take ~ 1-2 hours, one CatBoost model from 5 to 10 minutes
● Inference duration: ~ 20 minutes

6. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

It seems to us that there are no dangerous moments. The models are light enough so that the inference can be reproduced easily.

7. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

No, all we did was look at the data using opencv and pandas.

8. How did you evaluate performance of the model other than the provided metric, if at all?

We tested the models either on validation or on deferred sets.

9. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

We tried: to train other Unet models, to train CatBoost on other fords, use TTA for Unet models.

10. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

It would be interesting for us to continue developing our own developments using a mix from a neural network and pixel-by-pixel gradient boosting. We think this approach is promising for this task. It makes sense to train the models more efficiently, use a large amount of data, it is possible to facilitate the models and reduce the inference time.
```