Technical requirements

1. Install Python (I use 3.9.18)
2. Install pip if is not present.
    1. Download https://bootstrap.pypa.io/get-pip.py
    2. Execute:

    ```
    python get-pip.py
    ```
3. Install requirements.

```
python -m pip install -r requirements.txt
```

4. Execute command call api.

```
python call_api.py [-h] --token_file TOKEN_FILE [--x_column X_COLUMN] [--y_column Y_COLUMN]
```

5. Execute command create graphics

```
python create_graphics.py [-h] [--x_column X_COLUMN] [--y_column Y_COLUMN] [--window WINDOW]
```

6. Execute command for training Deep Learning model for classifying images.

```
python analyze_images.py [-h] [--model_save_path MODEL_SAVE_PATH] [--batch_size BATCH_SIZE]
```

where:

- MODEL_SAVE_PATH is the path where trained parameters are stored (default: model.pth)
- BATCH_SIZE is the size of the batch of data loaded at the same time. (default: 4)
- LR is the learning rate of the algorithm. If it is big, learning is unstable (default: 0.001)
- EPOCHS is the number of times the algorithm iterates over the data. (default: 30).
- images_folder is the folder with the images. It has to have the following structure:
    - images_folder
        * correct/*.png
        * incorrect/*.png

Example:

```
python analyze_images.py images/ --epochs 10 --model_save_path model.pth
```

7. Execute command for using the trained model.

```
test_model.py [-h] [--model_path MODEL_PATH] [--result_csv RESULT_CSV] image_folder
```

where:

- MODEL_PATH is the path of the trained model. (default: model.pth)
- RESULT_CSV is the path where results will be stored as csv file. (default: result.csv)

- image_folder is the folder containing the images that will be tested. This folder just must contain the png files to be tested.

Example:

```
python test_model.py --model_path model.pth test_image_folder/
```

The result.csv should look as:

```
image_name,incorrect probability
sample.png,0.988
```

8. Execute command for training Machine Learning system for csv.

```
analyze_csv.py [-h] [--model_save_path MODEL_SAVE_PATH] csv_folder
```

where:

- MODEL_SAVE_PATH is the path where trained parameters are stored. (default: model.pkl)
- csv_folder is the folder with the csv. It has to have the following structure:
  - csv_folder
    - correct/*.csv
    - incorrect/*.csv

Example:

```
python analyze_csv.py ml_data/
```

9. Execute command for using the trained model.

```
test_model_csv.py [-h] [--model_path MODEL_PATH] [--result_csv RESULT_CSV] csv_folder
```

where:

- MODEL_PATH is the path where trained model is stored. (default: model.pkl)
- RESULT_CSV is the path where result is stored. (default: result.csv)
- csv_folder is the folder with the csv files for apply the model.

Example

```
python test_model_csv.py test_csv/
```

10. Execute pipeline.

```
pipeline.py [-h]
[--incorrectness_threshold INCORRECTNESS_THRESHOLD]
[--overlap_threshold OVERLAP_THRESHOLD]
[--min_length MIN_LENGTH]
[--log_folder LOG_FOLDER]
csv_folder
graphs_folder
model_path
combinations_csv_path
```

```
combinations_png_path
best_combinations_path
```

where: - INCORRECTNESS_THRESHOLD is the threshold for considering a sample as “best sample” - OVERLAP_THRESHOLD is the overlap threshold for combining 2 csv. - MIN_LENGTH is the minimum length for considering csv. - LOG_FOLDER is the folder for creating the logs. - csv_folder is the folder with the csv files for apply the model. - graphs_folder is the folder to store the graphs. - model_path is the path to the model for computing incorrectness. - combinations_csv_path is the path to save combinations csv. - combinations_png_path is the path to save combinations png. - best_combinations_path is the path to save the best combinations png.

Example:

```
python pipeline.py
new_csv/csv_files/
new_csv/figs2/
model.pth
new_csv/combinations2/
new_csv/combinations2_img/
new_csv/best_combinations_png/
--overlap_threshold 0.1
--min_length 50
--incorrectness_threshold 0.005
```