Artefact* : Automated Data Error Cleaning Impact on Federated Learning Utility and Fairness

James Sudlow
james.sudlow@insa-lyon.fr
INSA Lyon - CNRS - LIRIS
Lyon France
Baudouin Naline
INSA Lyon - CNRS - LIRIS
Lyon France
baudouin.naline@insa-lyon.fr
Sara Bouchenak
INSA Lyon - CNRS - LIRIS
Lyon France
sara.bouchenak@insa-lyon.fr

james.sudlow@insa-lyon.fr baudouin.naline@insa-lyon.fr sara.bouchenak@insa-lyon.fr

---

Résumé
The paper is an experimental report on the impact of data cleaning in a Federated Learning setting. This implies running a large number of experiments, so for reproducibility testing a few examples have been selected to allow a person to test it within reasonable compute times. It already includes a requirement.txt and one should use Python 3.13.5

Mots-clés : fluke, python

---

1. General overview

1.1. Contributions

These large scale set of experiments are run for different data cleaning methods. With four utility and five demographic fairness metrics measured.
For command line prompt, linebreaks are to allow the code to fit on the page, but should be removed before running.

- C_1 Main experiments for all cleaning methods in an evenly distributed scenario.
- C_2 An experiement where the distribution among clients varies for one dataset.
- C_3 Testing the effectiveness of bias mitigation methods in this context.
- C_4 Testing the impact of error injection.

1.2. Artefacts

For all contributions the code is available on
https ://github.com/sara-bouchenak/TITANIA
With the relevant datasets and traces included.

*. Format inspiré de SC : `https://github.com/sara-bouchenak/TITANIA`

2. Details of the artefacts

2.1. Run a simple examples (2 minutes human time, 7 minutes CPU time)
Link to contribution

This section presents a simple example, to test the general functioning of the code : the OL-std-mean-G cleaning method applied on the Adult Dataset, an IID setting, with a logistic regression model.

Expected results

This should get the same results as one run for one of the experiments, as found in :
traces/overall_impact/dataset=Adult/model=LogRegression/data_cleaning=OL-std-mean-G/exp_seed=101,data_seed=59/results.json

Artefacts setup

Install the correct version of python, install requirement.txt and use the adult dataset which should be included.

Running the artefacts

For command line prompt, linebreaks are to allow the code to fit on the page, but should be removed before running.
Run this short command for training the model :

```
python main.py -m +experiment=overall_impact/Adult/LogRegression/simple_example
```

After the script is completed, the output files are saved in the subfolder of :
'outputs/overall_impact/Adult/EXP_TIMESTAMP/model=LogRegression/data_cleaning=OL-std-mean-G
/exp_seed=101,data_seed=5'
(with 'EXP_TIMESTAMP' the timestamp of the experiment).

Comments on the results analysis

It should contain a file 'results.json' with the same values as the result experiment in
traces/overall_impact/dataset=Adult/model=LogRegression/data_cleaning=OL-std-mean-G/exp_seed=101,data_seed=59/results.json
Metrics are explained in the ReadMe section producing-traces-and-statistics.
This script 'src/TITANIA/result_statistics/compare_results_json.py' helps compare the global performance of the two JSON files :

```
python src/TITANIA/result_statistics/compare_results_json.py
--path_to_json_1 PATH_TO_THE_JSON --path_to_json_2 traces/
overall_impact/dataset=Adult/model=LogRegression/
data_cleaning=OL-std-mean-G/exp_seed=101,data_seed=59/results.json
```

2.2. Run a complete experiment (5 minutes human time, 1h30 CPU time)
Link to contribution

This section reproduces some results of the Figure 3 of the paper (Data cleaning under various FL non-IID settings). The tests use multiple cleaning methods and vary distribution giving depth to the results.

Expected results

It focuses on 8 experiments, which considers 2 combinations of cleaning methods (*w/o cleaning* being the baseline without cleaning, and *w/ cleaning+* being with cleaning by two Federated learning solutions, i.e., FedCorr for label errors and Cafe for missing values, as well as a mathematical outlier cleaning method, i.e., OL-std-mode-L) and 4 data distributions (*IID*, *non-IID 0.01*, *non-IID 0.05*, and *non-IID 0.1*).

Artefacts setup

Install the correct version of python, install requirement.txt and use the adult dataset which should be included.

Running the artefacts

For command line prompt, linebreaks are to allow the code to fit on the page, but should be removed before running.
Two experiment config files are provided to reproduce some results for the varied dirichlet figure of the paper.

```
python main.py -m +experiment=FL_non_iid_settings/Adult/dirichlet_example
```

```
python main.py -m +experiment=FL_non_iid_settings/Adult/iid_example
```

This creates subfolders with the outputs in 'outputs/FL_non_iid_settings/Adult/EXP_TIMESTAMP' (with 'EXP_TIMESTAMP' the timestamp of the experiment).
Either run the following script

```
python src/TITANIA/result\_statistics/copy\_traces.py --date FOLDER\_DATE
```

With 'FOLDER_DATE' the date which should be your folder's name (for example 2026-04-14_11-44-41).
Or create a folder 'example' and within it 'dataset=Adult'. Then drag and drop the relevant folders so that the structure is :
traces
– example/
—- dataset=Adult/
—— data_distribution_name=iid/
—— data_distribution_name=label_dirichlet_skew,dirichlet_alpha=0.1/
—— data_distribution_name=label_dirichlet_skew,dirichlet_alpha=0.01/
—— data_distribution_name=label_dirichlet_skew,dirichlet_alpha=0.05/
Then run these two lines to create graphs, the first one creates a CSV in the 'traces/example/dataset=Adult' folder, the second one creates graphs in the 'plots/example/Adult' folder :

```
python ./src/TITANIA/result_statistics/create_dataset.py
--dataset Adult --experiment=example
```

```
python ./src/TITANIA/result_statistics/graphs.py
--dataset Adult --experiment=example
```

Comments on the results analysis

The 'plots/example/Adult/non_iid_Accuracy_EOD_race.pdf' plot should be the same as the plot in the paper (with only squares and circles) with some differences in dimensions.

2.3. All results of the paper from traces
Link to contribution

This section explains how to use the traces to reproduce all tables and figures of the paper.

Running the artefacts

For command line prompt, linebreaks are to allow the code to fit on the page, but should be removed before running.

2.3.1. Produce all result tables of the paper (2 minutes human time, 2 minutes CPU time)

Run the following command to compute all result tables (i.e., Tables 3 to 7) based on the traces :

```
python ./src/TITANIA/result_statistics/print_tables.py --exp_name EXP_NAME
```

With 'EXP_NAME', the name of the selected experiment folder in the 'traces' folder.
Running all experiments would be too slow, so traces folders is provided 'traces/overall_impact'.

2.3.2. Produce all result graphs of the paper (2 minutes human time, 25 minutes CPU time)

To create all the graphs from the paper using the traces previously produced, run these commands (to get just the graphs in the paper replace 'Heart,ARS,KDD,MEPS,Adult' by 'Adult' makes running the code faster).
The first one creates CSVs based one the traces :

```
python ./src/TITANIA/result_statistics/create_dataset.py
--dataset Heart,ARS,KDD,MEPS,Adult
--experiment=FL_non_iid_settings,error_rates,bias_mitigation
```

The second creates a plot structure :

```
python ./src/TITANIA/result_statistics/graphs.py
--dataset Heart,ARS,KDD,MEPS,Adult
--experiment=FL_non_iid_settings,error_rates,bias_mitigation
```

The plots folder includes all the potential combinations. The relevant ones in the paper are :
- 'plots/FL_non_iid_settings/Adult/non_iid_Accuracy_EOD_race.pdf'
- 'plots/bias_mitigation/Adult/bias_mitigation_Accuracy.pdf'
- 'plots/bias_mitigation/Adult/bias_mitigation_SPD_race.pdf'
- 'plots/error_rates/Adult/error_rate_Precision.pdf'
- 'plots/error_rates/Adult/error_rate_AOD_race.pdf'