Replication Guide: Forecasting Conflict in Africa with Automated Machine Learning Systems*

Vito D'Orazio
Economic, Political, and Policy Sciences
University of Texas at Dallas

Yu Lin
Engineering and Computer Science
University of Texas at Dallas

1 Prerequisite

Required packages and their version are listed below:

- Python 3.6
- Pandas 1.1.0
- Numpy 1.19.1
- Matplotlib 3.2.2
- Sklearn 0.19.0
- H2O 3.32.0.1
- R 4.1.1

2 Data

The `data` directory includes the initial data files (*cm.csv*, *pgm.csv*, *dv_cm_africa.csv*, *dv_pgm_africa.csv*, and *grid_conflict_count.csv*), the October - March true forecasts of the level of violence (*cm_fin.csv*, *pgm.csv*), and the submitted forecasts of the change in violence for each of the competition tasks.

3 Main Experiments

3.1 Prepare Input Files

First, follow the guidelines in the ViEWS Manual to install the ViEWS python package at https://github.com/UppsalaConflictDataProgram/OpenViEWS2. The documentation we used is Release 2.0, dated June 11, 2020. With this package, the initial data is built with `views1.py`, `builddv.R`, and `zerogrids.R`. These produce the following files:

- *cm.csv* and/or *pgm.csv*;

*Replication materials are available at http://dvn.iq.harvard.edu/dvn/dv/internationalinteractions. All questions regarding replication may be directed to Vito D'Orazio.

- *dv_cm_africa.csv* and/or *dv_pgm_africa.csv*;
- *grid_conflict_count.csv*

where *cm.csv* (*pgm.csv*) contains the filtered feature data and *dv_cm_africa.csv* (*dv_pgm_africa.csv*) contains the shifted target variable (*ln_ged_best_sb*). The *grid_conflict_count.csv* counts the historical conflicts of each grid, which is required only for **Hurdle** models.

3.2 Generate D3M Datasets

Before conducting the experiment, we first need to generate a **D3M Dataset** for each task. To do so, open a terminal, go to the path *Views_toolbox/src/auxiliary/* and execute following command.

```
$python3 configuration_builder.py [PARAMs]
```

The list of acceptable PARAMs is:

- (REQUIRED) **--dataset** : *cm* or *pgm*
- (REQUIRED) **--raw_path** : Path to *cm.csv* or *pgm.csv*
- (REQUIRED) **--dv_path** : Path to *dv_cm_africa.csv* or *dv_pgm_africa.csv*
- (OPTIONAL) **--grid_path** : Path to *grid_conflict_count.csv*
- (REQUIRED) **--output_path** : Output folder path
- (REQUIRED) **--name** : Name of the generated D3M folder
- (REQUIRED) **--feature_set** : Feature set to use, select from *views*, *views_30* or *dynamic*
- (OPTIONAL) **--time_mode** : Time split mode, select from 0 & 1, default value is 0.
- (REQUIRED) **--shift** : Time lag, select from 1, 2, 3, 4, 5, 6

The output of this command is a JSON file, named *d3m_config.json*, in the folder *Views_toolbox/src/*. Go to that path and run following command.

```
$python3 vtb_v2.py --config d3m_config.json
```

For example, to generate a D3M dataset for (PGM, Dynamics, Time lag = 1) in Table 4, run the first command with *--dataset pgm --name pgm_s1 --feature_set dynamic --shift 1* to build corresponding configuration file: *d3m_config.json*. Then run the second command to generate the actual D3M dataset. Please adjust PARAMs accordingly to generate different D3M dataset. Estimated running time is $\sim$ 5min for CM dataset and $\sim$ 15min for PGM dataset.

3.3 Train Model & Inference

Assume the generated D3M dataset is in the Desktop, *~/Desktop/pgm_s1/*

3.3.1 AlphaD3M & CMU

In order to train models of **AlphaD3M** & **CMU**, you need first install the *d3m_interface* package. Please refer to https://d3m-interface.readthedocs.io/en/latest/ for the installation guide.

After the installation is complete, go to the provided *baseline_models.ipynb* (Jupyter Notebook) and run the code from beginning (Table 4). It is worth noting that $\sim$ 2 hrs is required for each configuration in the *pending_list*.

Moreover, notice that **AlphaD3M** & **CMU** support built-in time series forcasting model. To generate such results (Table 5), go to *vtb_v2.py* L75, replace *['regression']* with *['forecasting']*, re-generate the D3M dataset and re-run *baseline_models.ipynb*.

The Pipeline Profiler Plot (Figure 2) can also be generated by running the last optional block in the *baseline_models.ipynb*.

3.3.2 H2O

To train a H2O model (Table 4), you could follow the instruction here. Alternatively, a text-version guidance is also provided. We used 32 cores for each run.

- Step 1. Download the H2O AutoML JAR file and run its server (*java -jar h2o.jar*)
- Step 2. Go to the Flow Web UI. The default site is *http://localhost:54321*
- Step 3. In the homepage, click **importFiles** button and upload the training file. For example, select
  $\sim$ */Desktop/pgm_s1/TRAIN/dataset_TRAIN/tables/learningData.csv*
  from the generated D3M dataset
- Step 4. Parse the uploaded file by clicking the **Parse these files** button. Remember to make attributes *country_name* and *month* as *Enum*
- Step 5. You will get a *learningData.hex* object after the parse operation. Then click **run AutoML**, set *training_frame* as *learningData.hex* and set *response_column* as *dv*. In the *ignored_columns* block, check *d3m_index* and start **Build Models**.
- Step 6. Wait for 1 hour until the training is done.
- Step 7. Follow step 3 & 4 to upload the test split. Specifically,
  $\sim$ */Desktop/pgm_s1/TEST/dataset_TEST/tables/learningData.csv*
  By default, you will get a new data object *leanringData1.hex*
- Step 8. Select the best model in the **Learderboard** (generated by Step 6), click the **Predict** button and select *leanringData1.hex*. The performance metrics will be presented.

3.4 Visualizations

3.4.1 Africa Conflict Plots

To generate the Africa forecast figures (Figure 3, 4, 5, 6, 7, 8) presented in the paper, three items are required: 1) download the prediction file generated by H2O AutoML Server; 2) Clone VIEWS git repository and install required packages, and 3) Replace the *example_maps.ipynb* with the provided *example_maps.ipynb*

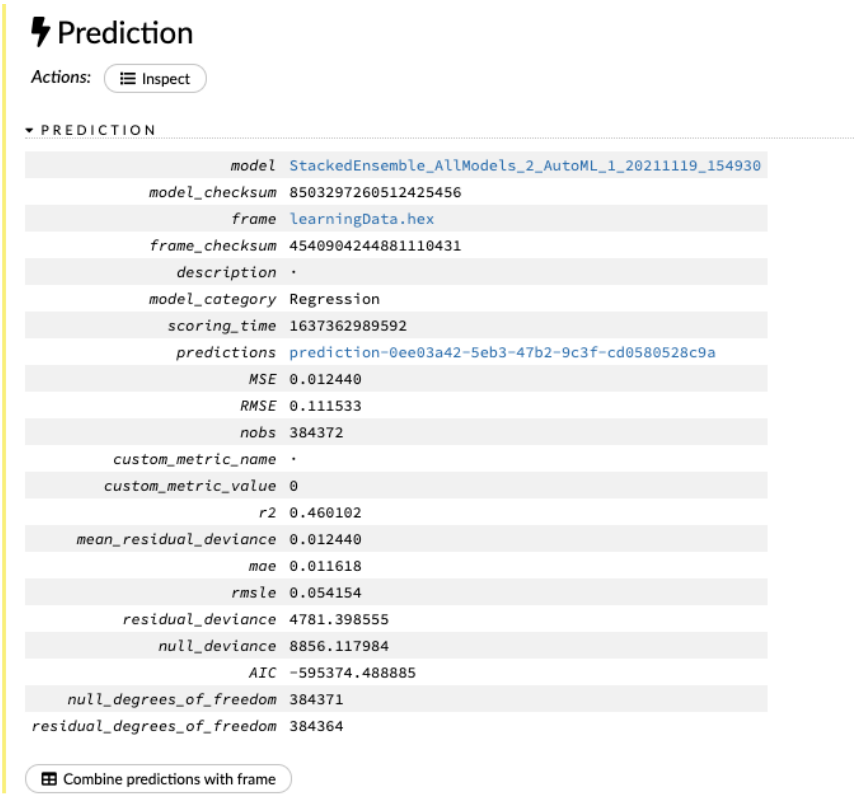


Figure 1: After prediction, click **Combine predictions with frame**.

To download the CSV file with predictions from H2O AutoML Server, you need first train a model by following Section 2.2 (Set *--time_mode* as 1) and Section 2.3.2. After that, select the top-1 model in the leaderboard and follow Figure 1 & 2. You will obtain a CSV file and assume its name is *prediction.csv*.

Follow the instruction here to clone the VIEWS git repository and install required packages. Once the installation is finished, replace the Jupyter Notebook */projects/plots/example_maps.ipynb* with the provided one and follow the guide in the notebook.

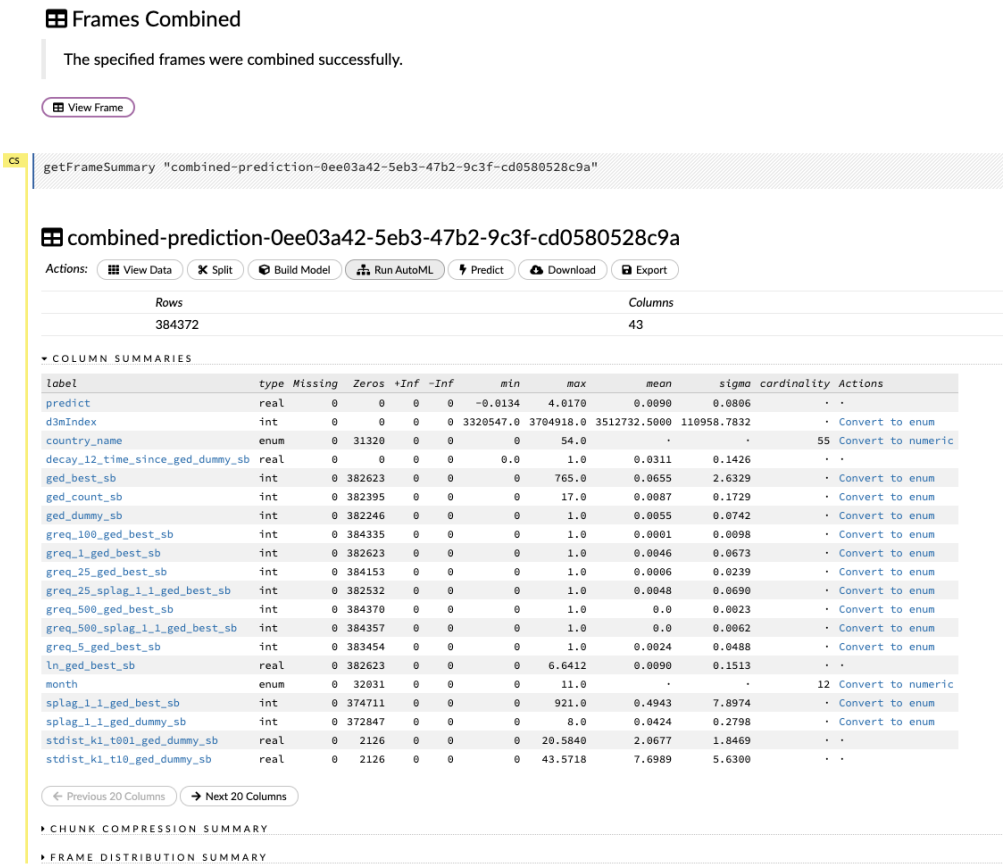
3.4.2 CM Delta plots

To generate the CM Delta plots (Figure 9 in the paper), you need first train 6 H2O models and download their CSV predictions. Go to *Views_toolbox/src/* and run following command.

```
$python3 process.py [PARAMs]
```

The list of acceptable PARAMs is:

- (REQUIRED) **--predN** : Path to the prediction files, $N = 1, 2, 3, 4, 5, 6$
- (REQUIRED) **--matchN** : Path to the matching files, $N = 1, 2, 3, 4, 5, 6$
- (REQUIRED) **--source** : Path to the original data, *cm.csv* or *pgm.csv*
- (REQUIRED) **--output** : Path of the output, default is *./cm_fin.csv*
- (REQUIRED) **--delta** : Flag to output delta results



Frames Combined

The specified frames were combined successfully.

View Frame

CS `getFrameSummary "combined-prediction-0ee03a42-5eb3-47b2-9c3f-cd0580528c9a"`

combined-prediction-0ee03a42-5eb3-47b2-9c3f-cd0580528c9a

Actions: View Data | Split | Build Model | Run AutoML | Predict | Download | Export

Rows	Columns
384372	43

COLUMN SUMMARIES

label	type	Missing	Zeros	+Inf	-Inf	min	max	mean	sigma	cardinality	Actions
predict	real	0	0	0	0	-0.0134	4.0170	0.0090	0.0806	·	·
d3mIndex	int	0	0	0	0	3320547.0	3704918.0	3512732.5000	110958.7832	·	Convert to enum
country_name	enum	0	31320	0	0	0	54.0	·	·	55	Convert to numeric
decay_12_time_since_ged_dummy_sb	real	0	0	0	0	0.0	1.0	0.0311	0.1426	·	·
ged_best_sb	int	0	382623	0	0	0	765.0	0.0655	2.6329	·	Convert to enum
ged_count_sb	int	0	382395	0	0	0	17.0	0.0087	0.1729	·	Convert to enum
ged_dummy_sb	int	0	382246	0	0	0	1.0	0.0055	0.0742	·	Convert to enum
greq_100_ged_best_sb	int	0	384335	0	0	0	1.0	0.0001	0.0098	·	Convert to enum
greq_1_ged_best_sb	int	0	382623	0	0	0	1.0	0.0046	0.0673	·	Convert to enum
greq_25_ged_best_sb	int	0	384153	0	0	0	1.0	0.0006	0.0239	·	Convert to enum
greq_25_splag_1_1_ged_best_sb	int	0	382532	0	0	0	1.0	0.0048	0.0690	·	Convert to enum
greq_500_ged_best_sb	int	0	384370	0	0	0	1.0	0.0	0.0023	·	Convert to enum
greq_500_splag_1_1_ged_best_sb	int	0	384357	0	0	0	1.0	0.0	0.0062	·	Convert to enum
greq_5_ged_best_sb	int	0	383454	0	0	0	1.0	0.0024	0.0488	·	Convert to enum
ln_ged_best_sb	real	0	382623	0	0	0	6.6412	0.0090	0.1513	·	·
month	enum	0	32031	0	0	0	11.0	·	·	12	Convert to numeric
splag_1_1_ged_best_sb	int	0	374711	0	0	0	921.0	0.4943	7.8974	·	Convert to enum
splag_1_1_ged_dummy_sb	int	0	372847	0	0	0	8.0	0.0424	0.2798	·	Convert to enum
stdist_k1_t001_ged_dummy_sb	real	0	2126	0	0	0	20.5840	2.0677	1.8469	·	·
stdist_k1_t10_ged_dummy_sb	real	0	2126	0	0	0	43.5718	7.6989	5.6300	·	·

Previous 20 Columns | Next 20 Columns

CHUNK COMPRESSION SUMMARY

FRAME DISTRIBUTION SUMMARY

Figure 2: After the frame is combined, click **Download**.

The output of this script is a calibrated CSV file named *cm_fin.csv*. The R script, `violins.R`, uses this to create the violin plots (Figure 9).

3.4.3 Feature Importance

To generate the feature importance plots for PGM and CM models (Figure 10 & 11), train a CM or PGM model by following Section 2.2 and Section 2.3.2. After that, click the best GBM model in the leaderboard and the variable importance plot will be presented in the pop-up window.