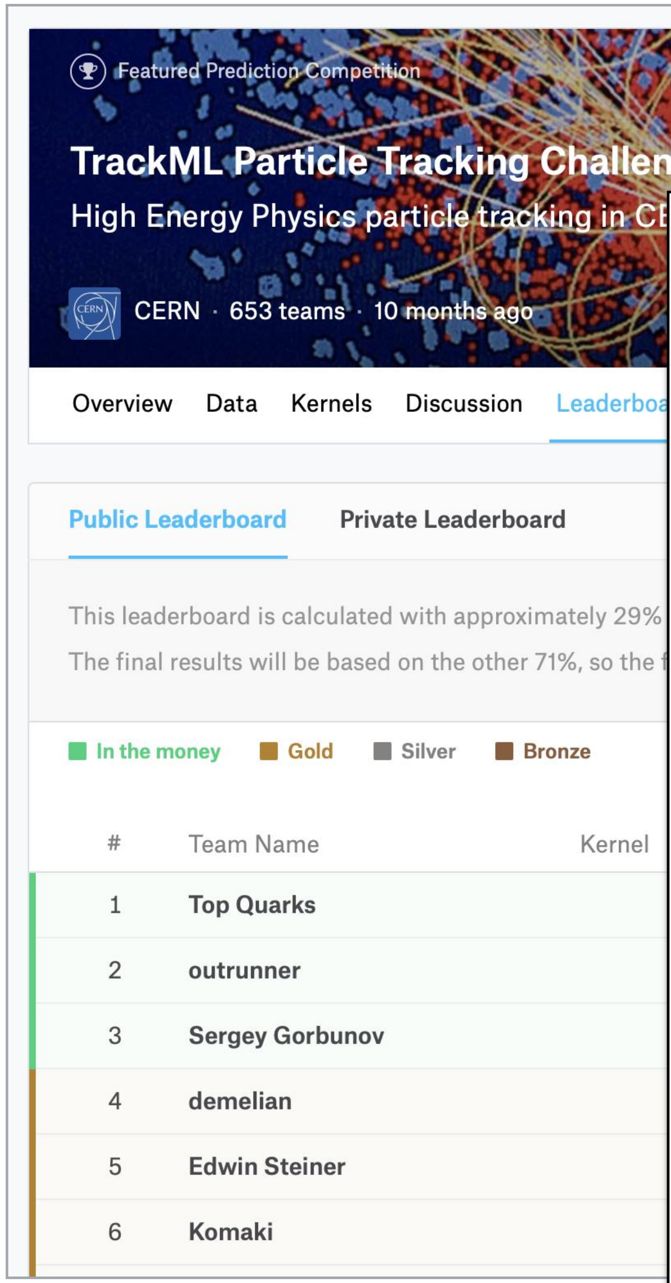
Reproducible Open Benchmarks for Data Analysis Platform

Kyle Cranmer, Irina Espejo,
Sebastian Macaluso, **Heiko Mueller**
New York University

Shih-Chieh Hsu, Aaron Maritz,
Ajay Rawat, Cha Suaysom
University of Washington

Benchmarks & Challenges



TrackML Particle Tracking Challenge

High Energy Physics particle tracking in CE

$25,000

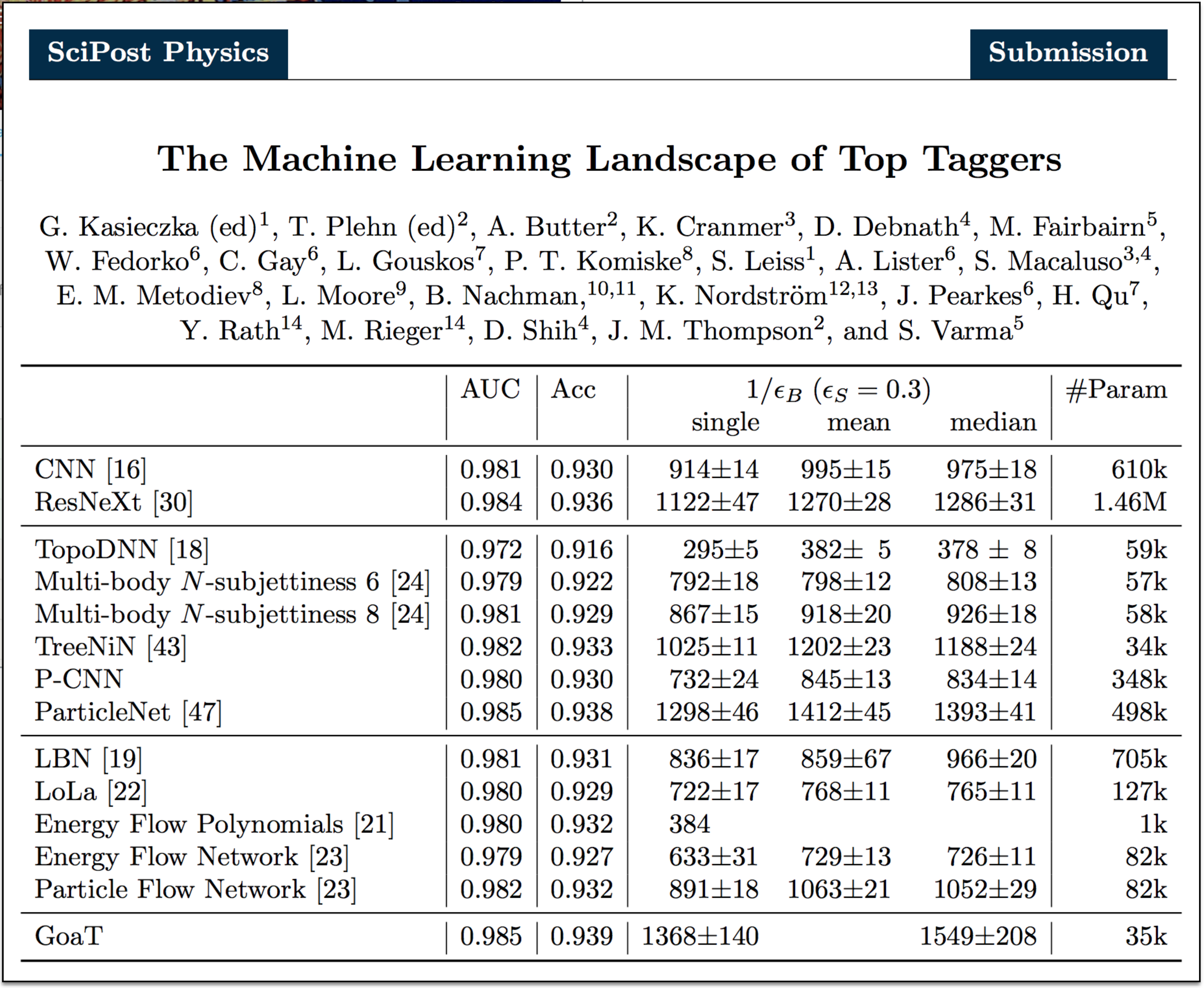
CERN · 653 teams · 10 months ago

This leaderboard is calculated with approximately 29%

The final results will be based on the other 71%, so the f

🟩 In the money ■ Gold ■ Silver ■ Bronze

#	Team Name	Kernel
1	Top Quarks	
2	outrunner	
3	Sergey Gorbunov	
4	demelian	
5	Edwin Steiner	
6	Komaki	



SciPost Physics | Submission

The Machine Learning Landscape of Top Taggers

G. Kasieczka (ed)[1], T. Plehn (ed)[2], A. Butter[2], K. Cranmer[3], D. Debnath[4], M. Fairbairn[5], W. Fedorko[6], C. Gay[6], L. Gouskos[7], P. T. Komiske[8], S. Leiss[1], A. Lister[6], S. Macaluso[3,4], E. M. Metodiev[8], L. Moore[9], B. Nachman,[10,11], K. Nordström[12,13], J. Pearkes[6], H. Qu[7], Y. Rath[14], M. Rieger[14], D. Shih[4], J. M. Thompson[2], and S. Varma[5]

	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$)			#Param
			single	mean	median	
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	867±15	918±20	926±18	58k
TreeNiN [43]	0.982	0.933	1025±11	1202±23	1188±24	34k
P-CNN	0.980	0.930	732±24	845±13	834±14	348k
ParticleNet [47]	0.985	0.938	1298±46	1412±45	1393±41	498k
LBN [19]	0.981	0.931	836±17	859±67	966±20	705k
LoLa [22]	0.980	0.929	722±17	768±11	765±11	127k
Energy Flow Polynomials [21]	0.980	0.932	384			1k
Energy Flow Network [23]	0.979	0.927	633±31	729±13	726±11	82k
Particle Flow Network [23]	0.982	0.932	891±18	1063±21	1052±29	82k
GoaT	0.985	0.939	1368±140		1549±208	35k

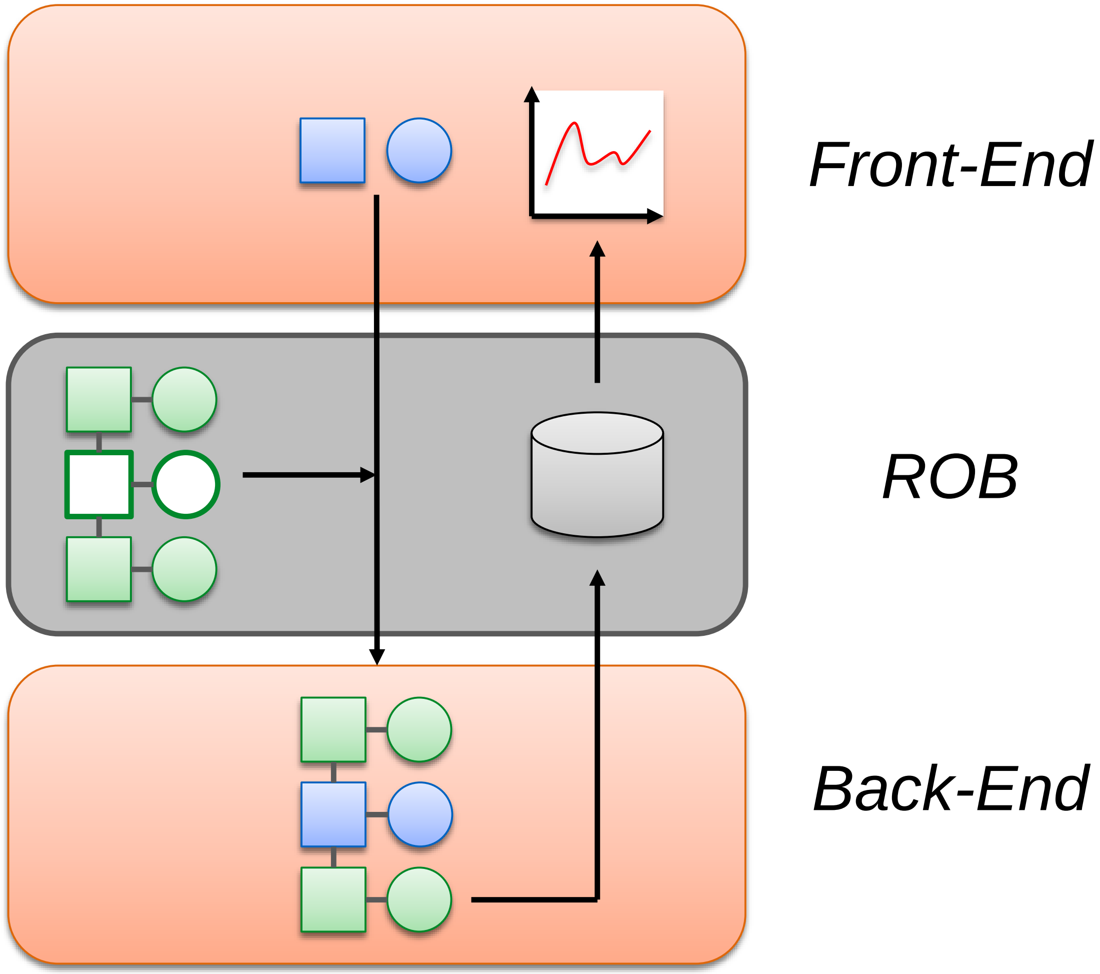
NYU

Reproducible Open Benchmarks Platform

Exploratory work for enabling such community benchmarks.

Components and Actors in ROB

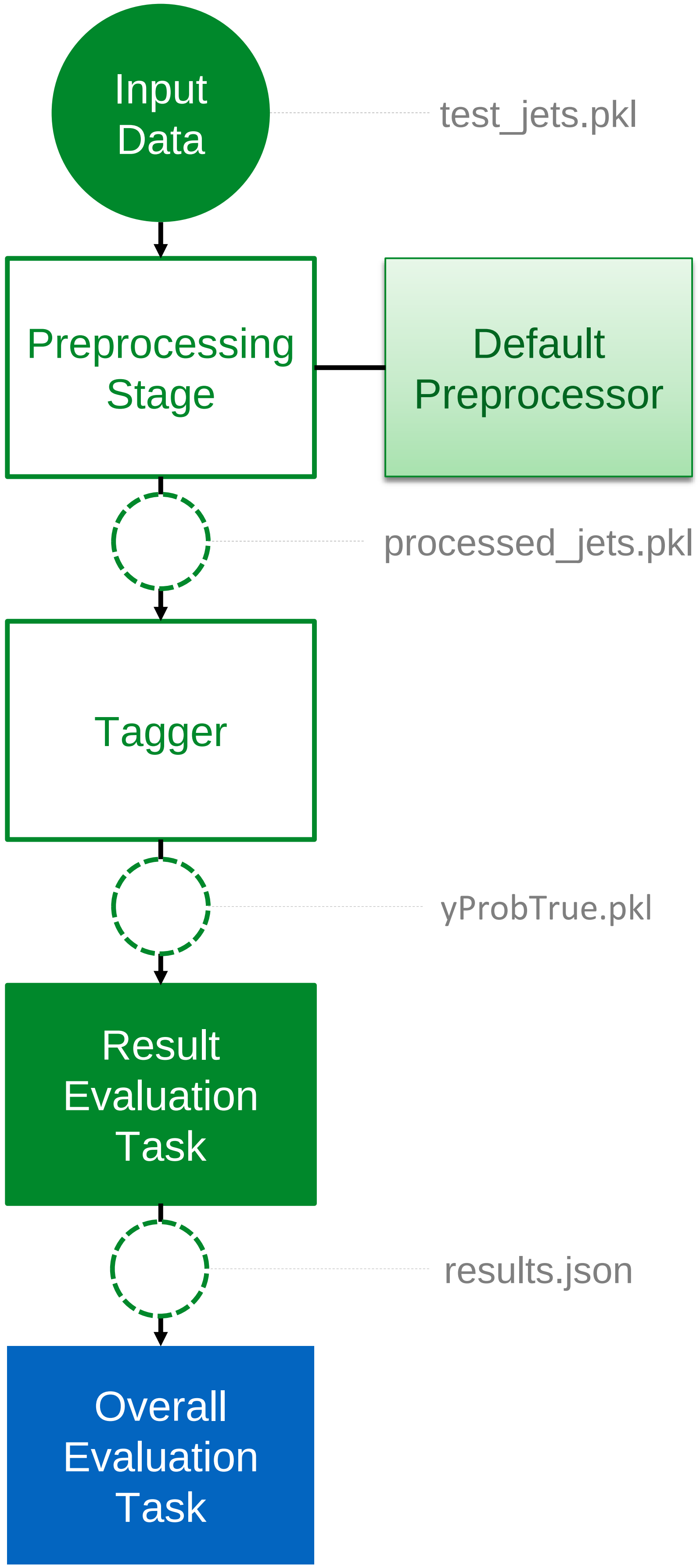
1. Benchmark workflow defined by **coordinator** along with input data.
2. **Users** provide code (e.g. docker containers) that satisfy workflow stages, input parameters, and input data (file upload).
3. **Back-end** processes workflows and evaluates metrics (powered for example by REANA).
4. **Front-end** to collect input and display results.



reana

Reproducible research data analysis platform

Benchmark Workflow Example

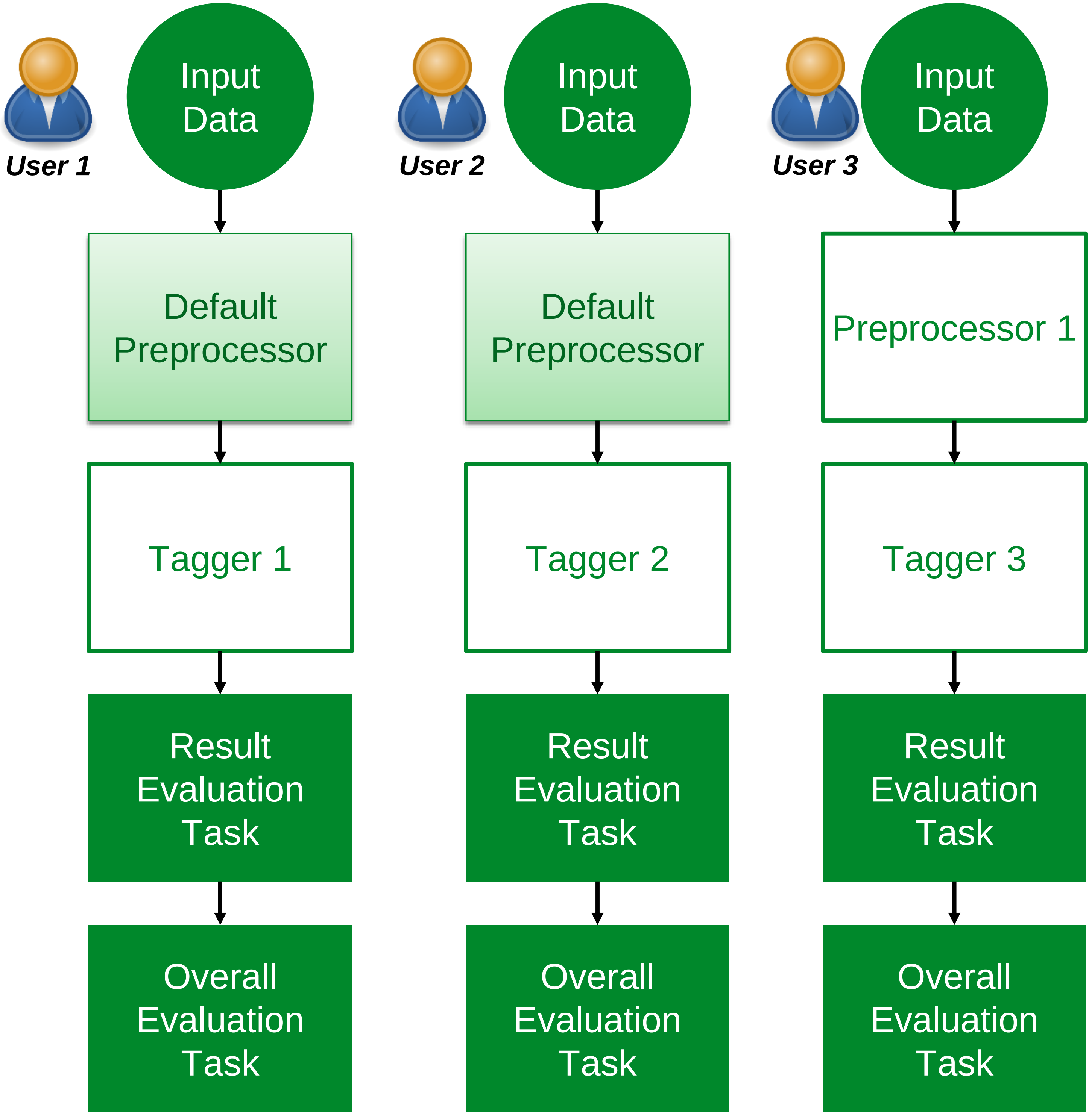


Workflow Templates

Coordinator defines structure of the workflow:

- ● Static input data
- ○ User-provided input data
- ◌ Intermediate output data
- ■ Implementation for static workflow stages
- ▨ Default implementation for variable workflow stages
- □ Variable (user-provided) workflow stages

Benchmark Workflow Example (cont.)



Benchmark Participants

Users create different instances of the workflow by providing **implementation for variable workflow stages** (and variable input data).

Workflow Templates

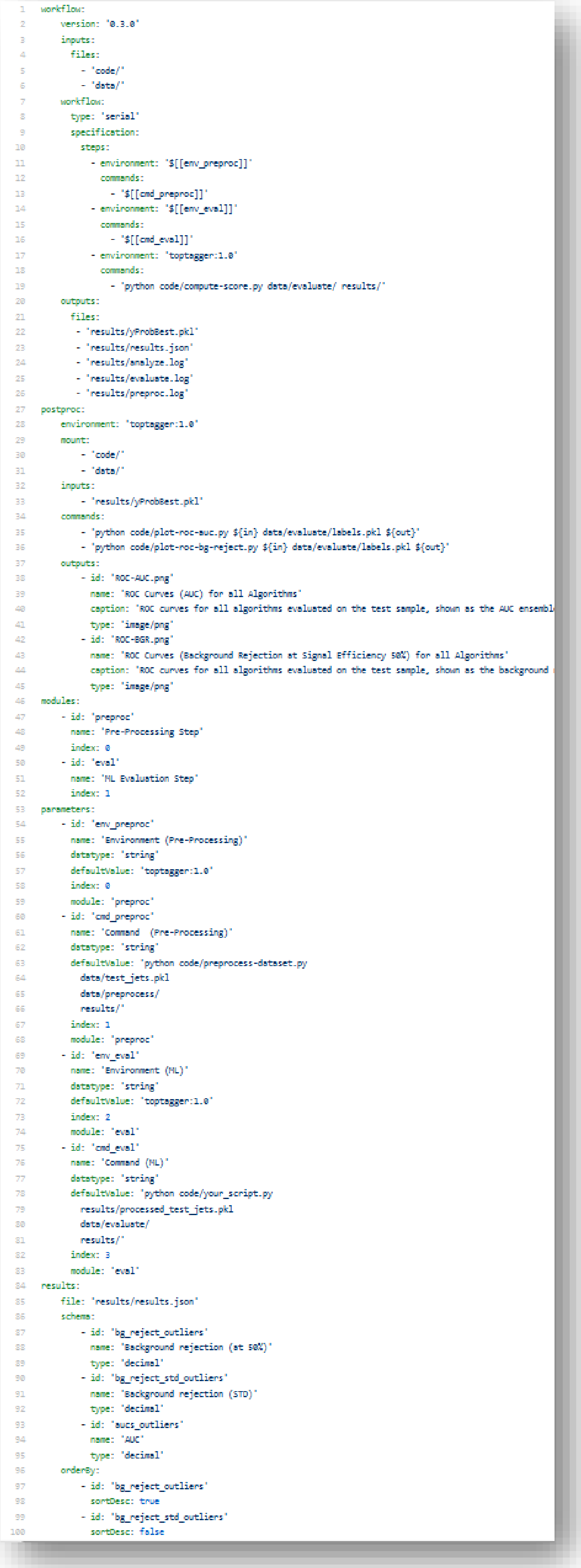
Components of Workflow Templates

1. Workflow specification (e.g. REANA serial workflow) with optional references to template parameters

2. Post-processing steps to evaluate overall performance (e.g. REANA serial workflow)

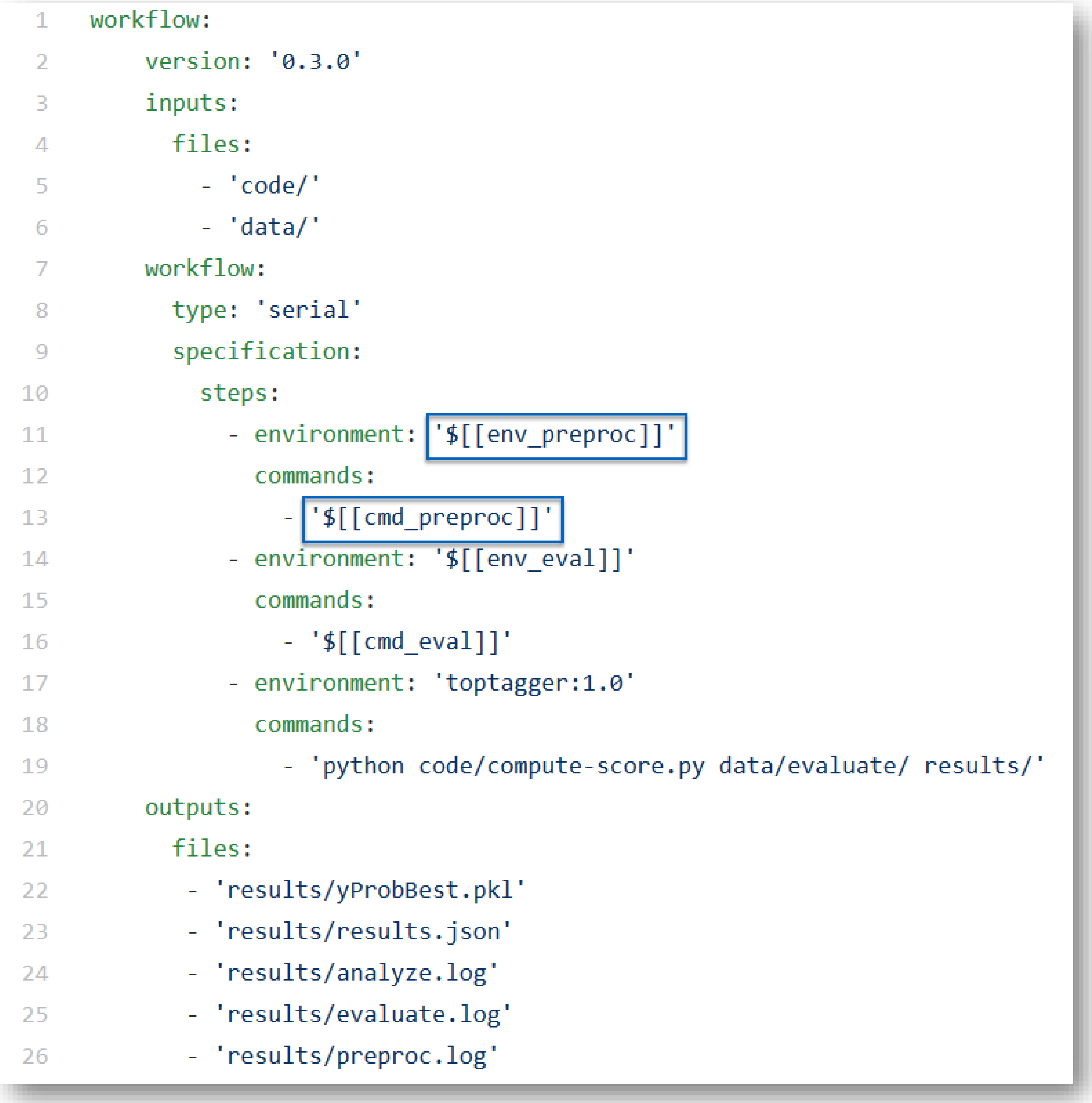
3. Declaration of template parameters (used by front-end for data input)

4. Specification of result schema to generate '*leader board*'.



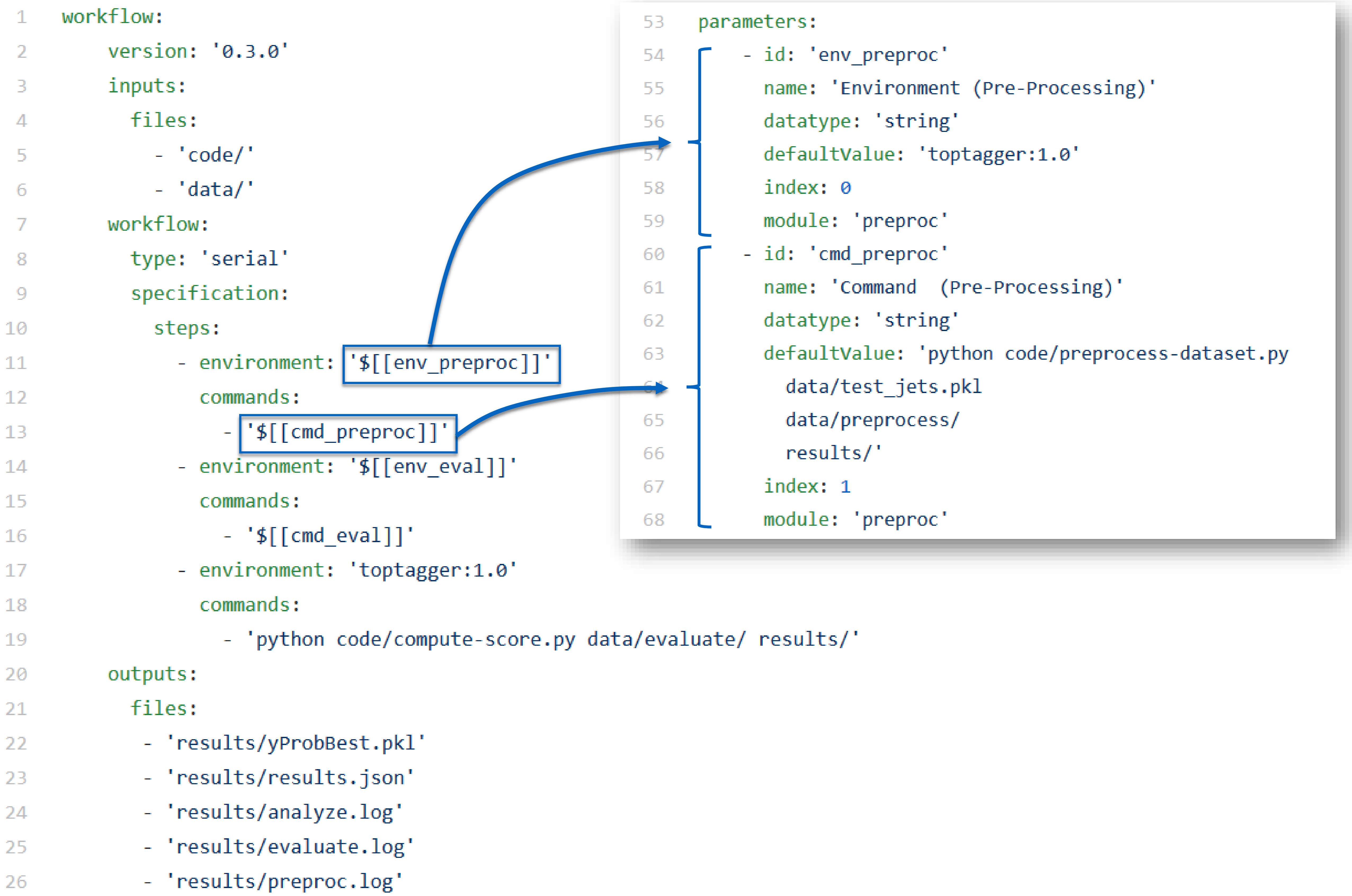
```
workflow:
  version: '0.3.0'
  inputs:
   files:
    - 'code/'
    - 'data/'
  workflow:
   type: 'serial'
   specification:
    steps:
     - environment: '$[[env_preproc]]'
      commands:
       - '$[[cmd_preproc]]'
     - environment: '$[[env_eval]]'
      commands:
       - '$[[cmd_eval]]'
     - environment: 'toptagger:1.0'
      commands:
       - 'python code/compute-score.py data/evaluate/ results/'
  outputs:
   files:
    - 'results/yProbBest.pkl'
    - 'results/results.json'
    - 'results/analyze.log'
    - 'results/evaluate.log'
    - 'results/preproc.log'
postproc:
  environment: 'toptagger:1.0'
  mount:
    - 'code/'
    - 'data/'
  inputs:
    - 'results/yProbBest.pkl'
  commands:
    - 'python code/plot-roc-auc.py ${in} data/evaluate/labels.pkl ${out}'
    - 'python code/plot-roc-bg-reject.py ${in} data/evaluate/labels.pkl ${out}'
  outputs:
    - id: 'ROC-AUC.png'
     name: 'ROC Curves (AUC) for all Algorithms'
     caption: 'ROC curves for all algorithms evaluated on the test sample, shown as the AUC ensembl
     type: 'image/png'
    - id: 'ROC-BGR.png'
     name: 'ROC Curves (Background Rejection at Signal Efficiency 50%) for all Algorithms'
     caption: 'ROC curves for all algorithms evaluated on the test sample, shown as the background
     type: 'image/png'
modules:
  - id: 'preproc'
   name: 'Pre-Processing Step'
   index: 0
  - id: 'eval'
   name: 'ML Evaluation Step'
   index: 1
parameters:
  - id: 'env_preproc'
   name: 'Environment (Pre-Processing)'
   datatype: 'string'
   defaultValue: 'toptagger:1.0'
   index: 0
   module: 'preproc'
  - id: 'cmd_preproc'
   name: 'Command  (Pre-Processing)'
   datatype: 'string'
   defaultValue: 'python code/preprocess-dataset.py
    data/test_jets.pkl
    data/preprocess/
    results/'
   index: 1
   module: 'preproc'
  - id: 'env_eval'
   name: 'Environment (ML)'
   datatype: 'string'
   defaultValue: 'toptagger:1.0'
   index: 2
   module: 'eval'
  - id: 'cmd_eval'
   name: 'Command (ML)'
   datatype: 'string'
   defaultValue: 'python code/your_script.py
    results/processed_test_jets.pkl
    data/evaluate/
    results/'
   index: 3
   module: 'eval'
results:
  file: 'results/results.json'
  schema:
    - id: 'bg_reject_outliers'
     name: 'Background rejection (at 50%)'
     type: 'decimal'
    - id: 'bg_reject_std_outliers'
     name: 'Background rejection (STD)'
     type: 'decimal'
    - id: 'aucs_outliers'
     name: 'AUC'
     type: 'decimal'
  orderBy:
    - id: 'bg_reject_outliers'
     sortDesc: true
    - id: 'bg_reject_std_outliers'
     sortDesc: false
```

Workflow Templates (cont.)



```yaml
workflow:
    version: '0.3.0'
    inputs:
      files:
        - 'code/'
        - 'data/'
    workflow:
      type: 'serial'
      specification:
        steps:
          - environment: '$[[env_preproc]]'
            commands:
              - '$[[cmd_preproc]]'
          - environment: '$[[env_eval]]'
            commands:
              - '$[[cmd_eval]]'
          - environment: 'toptagger:1.0'
            commands:
              - 'python code/compute-score.py data/evaluate/ results/'
    outputs:
      files:
       - 'results/yProbBest.pkl'
       - 'results/results.json'
       - 'results/analyze.log'
       - 'results/evaluate.log'
       - 'results/preproc.log'
```

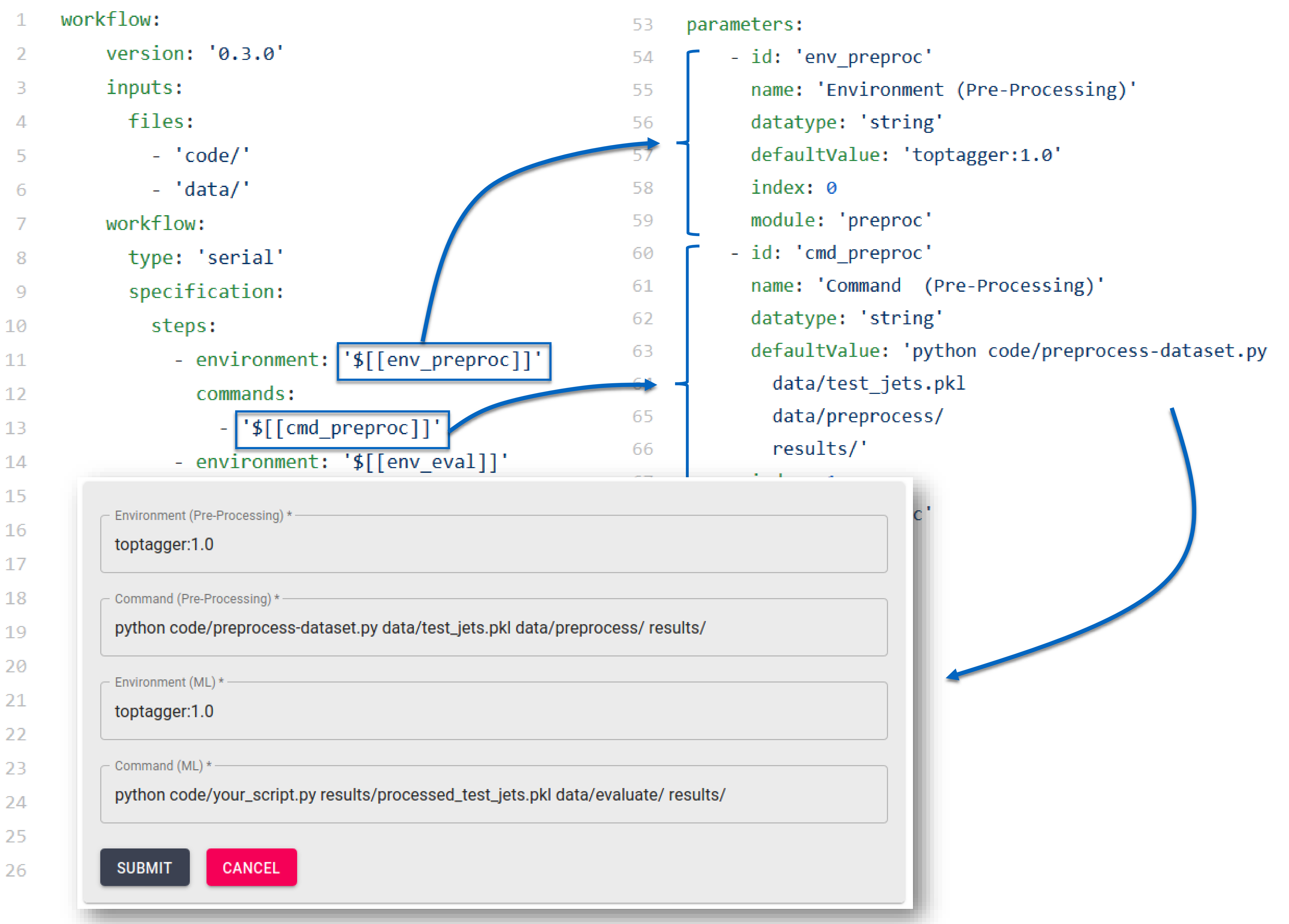
Workflow Templates (cont.)



```
workflow:
    version: '0.3.0'
    inputs:
      files:
        - 'code/'
        - 'data/'
    workflow:
      type: 'serial'
      specification:
        steps:
          - environment: '$[[env_preproc]]'
            commands:
              - '$[[cmd_preproc]]'
          - environment: '$[[env_eval]]'
            commands:
              - '$[[cmd_eval]]'
          - environment: 'toptagger:1.0'
            commands:
              - 'python code/compute-score.py data/evaluate/ results/'
    outputs:
      files:
       - 'results/yProbBest.pkl'
       - 'results/results.json'
       - 'results/analyze.log'
       - 'results/evaluate.log'
       - 'results/preproc.log'
```

```
parameters:
    - id: 'env_preproc'
      name: 'Environment (Pre-Processing)'
      datatype: 'string'
      defaultValue: 'toptagger:1.0'
      index: 0
      module: 'preproc'
    - id: 'cmd_preproc'
      name: 'Command  (Pre-Processing)'
      datatype: 'string'
      defaultValue: 'python code/preprocess-dataset.py
        data/test_jets.pkl
        data/preprocess/
        results/'
      index: 1
      module: 'preproc'
```

Workflow Templates (cont.)



```
workflow:
    version: '0.3.0'
    inputs:
      files:
        - 'code/'
        - 'data/'
    workflow:
      type: 'serial'
      specification:
        steps:
          - environment: '$[[env_preproc]]'
            commands:
              - '$[[cmd_preproc]]'
          - environment: '$[[env_eval]]'
```

```
parameters:
  - id: 'env_preproc'
    name: 'Environment (Pre-Processing)'
    datatype: 'string'
    defaultValue: 'toptagger:1.0'
    index: 0
    module: 'preproc'
  - id: 'cmd_preproc'
    name: 'Command  (Pre-Processing)'
    datatype: 'string'
    defaultValue: 'python code/preprocess-dataset.py
      data/test_jets.pkl
      data/preprocess/
      results/'
```

Environment (Pre-Processing) *

toptagger:1.0

Command (Pre-Processing) *

python code/preprocess-dataset.py data/test_jets.pkl data/preprocess/ results/

Environment (ML) *

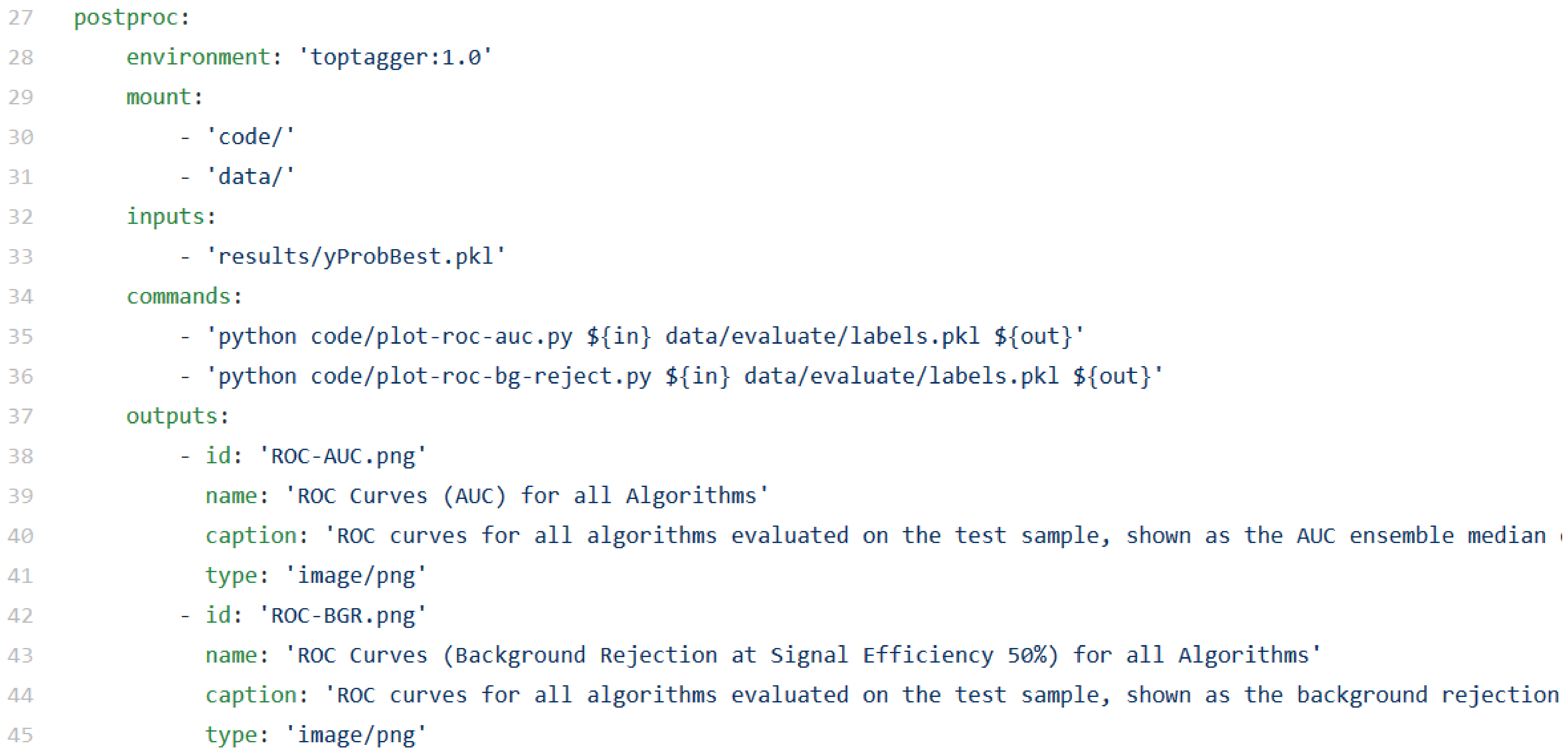
toptagger:1.0

Command (ML) *

python code/your_script.py results/processed_test_jets.pkl data/evaluate/ results/

Workflow Templates (cont.)

Post-processing workflow to summarize overall results (e.g., generate plots)

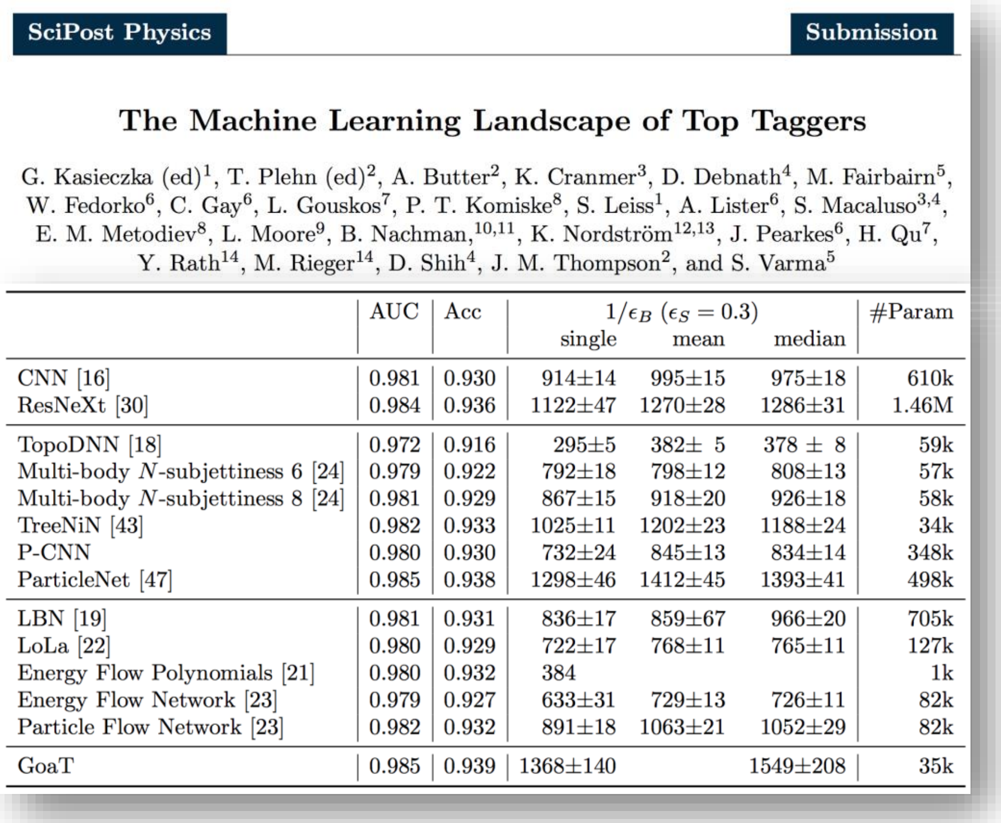


```
postproc:
    environment: 'toptagger:1.0'
    mount:
        - 'code/'
        - 'data/'
    inputs:
        - 'results/yProbBest.pkl'
    commands:
        - 'python code/plot-roc-auc.py ${in} data/evaluate/labels.pkl ${out}'
        - 'python code/plot-roc-bg-reject.py ${in} data/evaluate/labels.pkl ${out}'
    outputs:
        - id: 'ROC-AUC.png'
          name: 'ROC Curves (AUC) for all Algorithms'
          caption: 'ROC curves for all algorithms evaluated on the test sample, shown as the AUC ensemble median
          type: 'image/png'
        - id: 'ROC-BGR.png'
          name: 'ROC Curves (Background Rejection at Signal Efficiency 50%) for all Algorithms'
          caption: 'ROC curves for all algorithms evaluated on the test sample, shown as the background rejection
          type: 'image/png'
```

Workflow Templates (cont.)

Result schema to store benchmark results in database and to generate ranking

```
results:
    file: 'results/results.json'
    schema:
        - id: 'bg_reject'
          name: 'Background rejection (at 50%)'
          type: 'decimal'
        - id: 'bg_reject_std'
          name: 'Background rejection (STD)'
          type: 'decimal'
        - id: 'auc'
          name: 'AUC'
          type: 'decimal'
    orderBy:
        - id: 'bg_reject'
          sortDesc: true
        - id: 'bg_reject_std'
          sortDesc: false
```



SciPost Physics

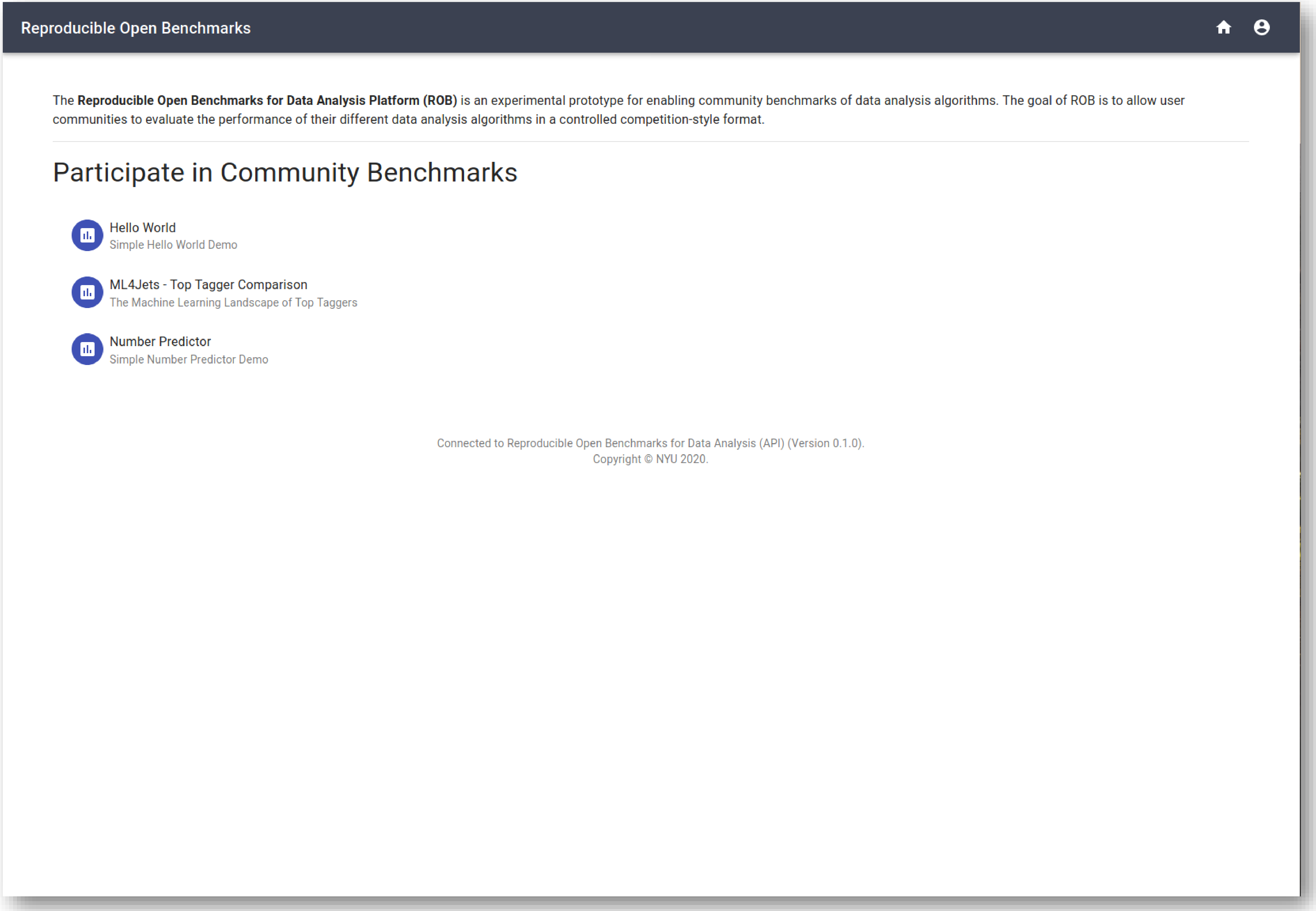
Submission

The Machine Learning Landscape of Top Taggers

G. Kasieczka (ed)[1], T. Plehn (ed)[2], A. Butter[2], K. Cranmer[3], D. Debnath[4], M. Fairbairn[5], W. Fedorko[6], C. Gay[6], L. Gouskos[7], P. T. Komiske[8], S. Leiss[1], A. Lister[6], S. Macaluso[3,4], E. M. Metodiev[8], L. Moore[9], B. Nachman,[10,11], K. Nordström[12,13], J. Pearkes[6], H. Qu[7], Y. Rath[14], M. Rieger[14], D. Shih[4], J. M. Thompson[2], and S. Varma[5]

	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$)			#Param
			single	mean	median	
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	867±15	918±20	926±18	58k
TreeNiN [43]	0.982	0.933	1025±11	1202±23	1188±24	34k
P-CNN	0.980	0.930	732±24	845±13	834±14	348k
ParticleNet [47]	0.985	0.938	1298±46	1412±45	1393±41	498k
LBN [19]	0.981	0.931	836±17	859±67	966±20	705k
LoLa [22]	0.980	0.929	722±17	768±11	765±11	127k
Energy Flow Polynomials [21]	0.980	0.932	384			1k
Energy Flow Network [23]	0.979	0.927	633±31	729±13	726±11	82k
Particle Flow Network [23]	0.982	0.932	891±18	1063±21	1052±29	82k
GoaT	0.985	0.939	1368±140		1549±208	35k

Demo

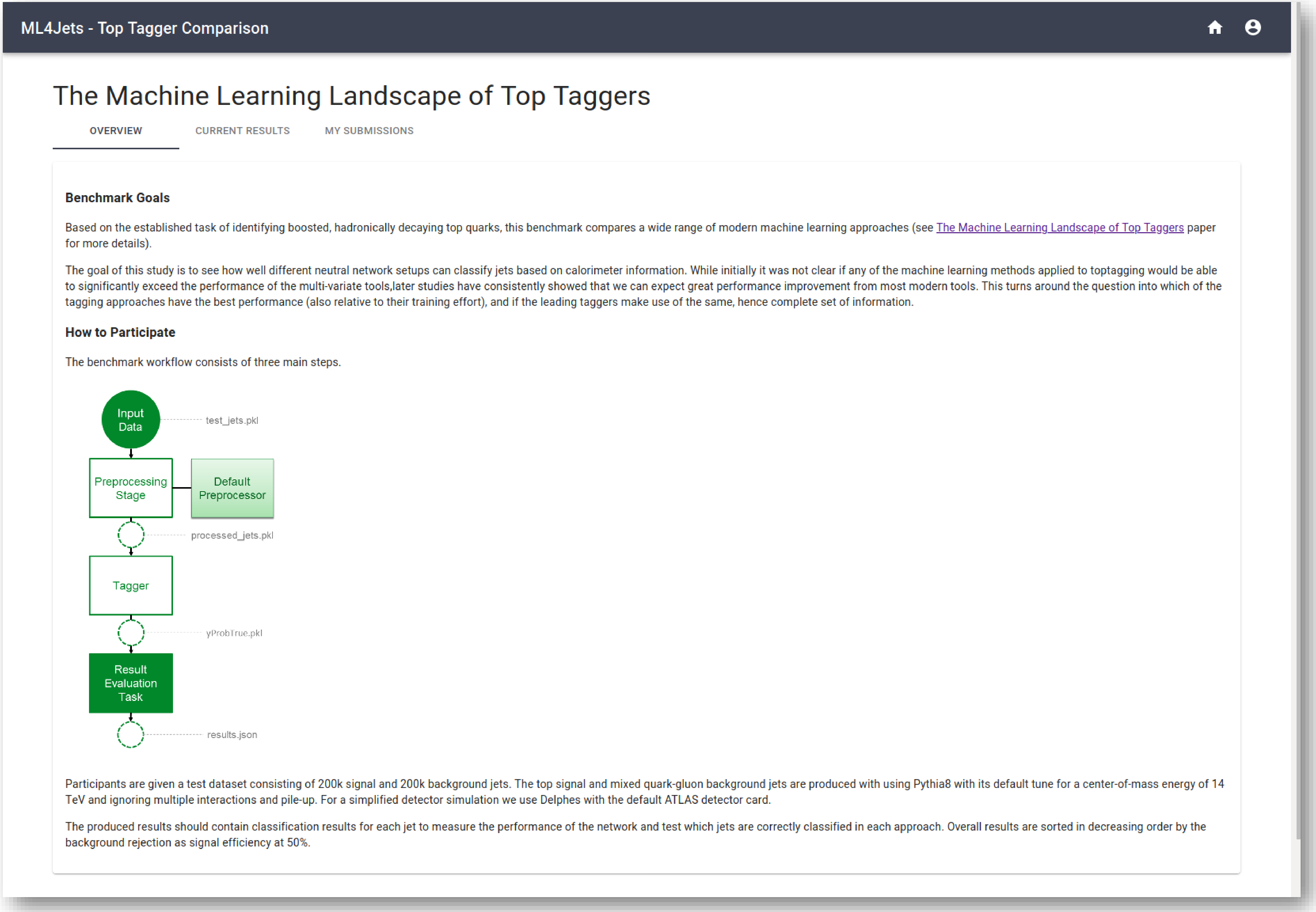


The **Reproducible Open Benchmarks for Data Analysis Platform (ROB)** is an experimental prototype for enabling community benchmarks of data analysis algorithms. The goal of ROB is to allow user communities to evaluate the performance of their different data analysis algorithms in a controlled competition-style format.

Participate in Community Benchmarks

- Hello World
  Simple Hello World Demo
- ML4Jets - Top Tagger Comparison
  The Machine Learning Landscape of Top Taggers
- Number Predictor
  Simple Number Predictor Demo

Connected to Reproducible Open Benchmarks for Data Analysis (API) (Version 0.1.0).
Copyright © NYU 2020.



The Machine Learning Landscape of Top Taggers

Benchmark Goals

Based on the established task of identifying boosted, hadronically decaying top quarks, this benchmark compares a wide range of modern machine learning approaches (see The Machine Learning Landscape of Top Taggers paper for more details).

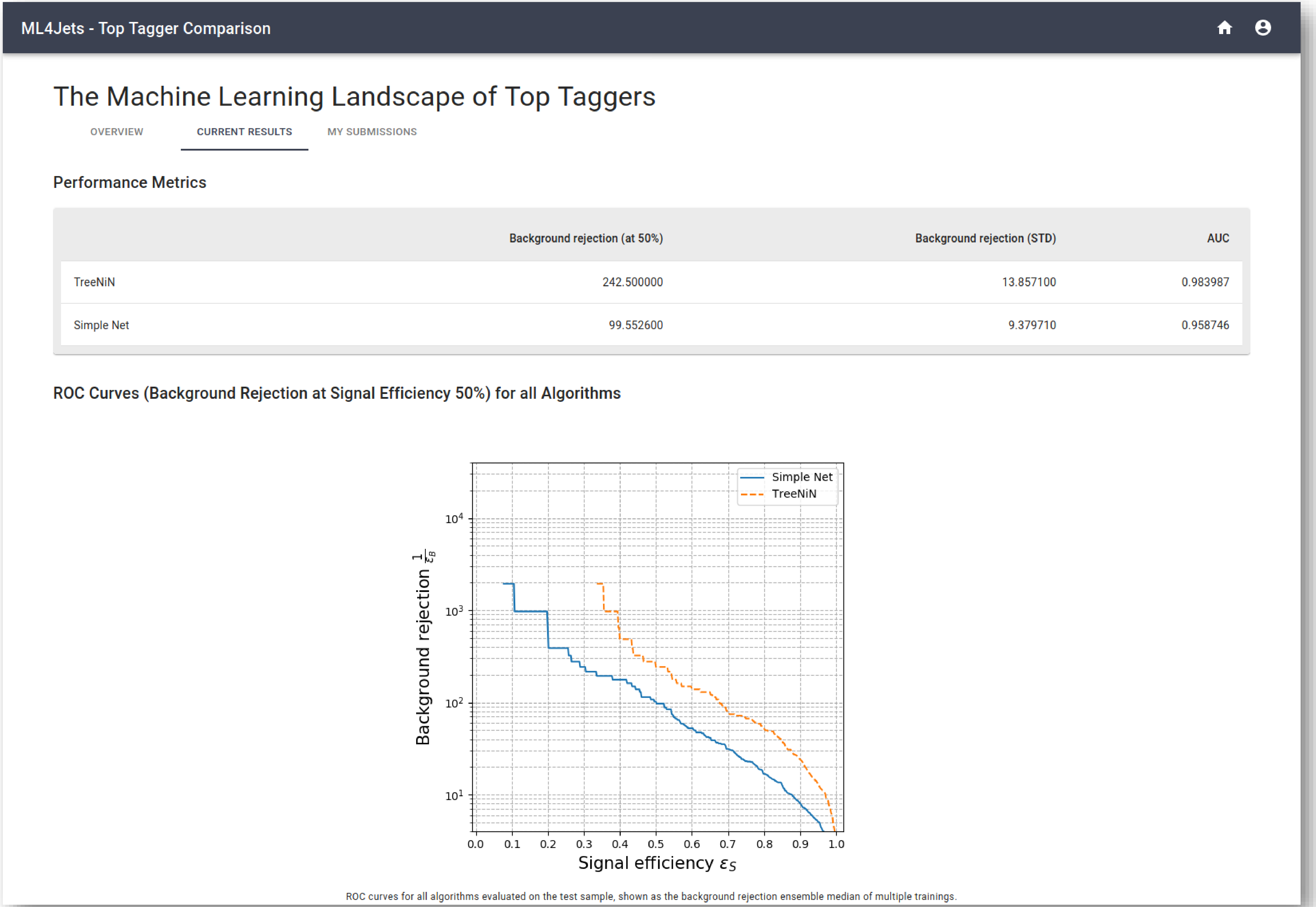
The goal of this study is to see how well different neutral network setups can classify jets based on calorimeter information. While initially it was not clear if any of the machine learning methods applied to toptagging would be able to significantly exceed the performance of the multi-variate tools,later studies have consistently showed that we can expect great performance improvement from most modern tools. This turns around the question into which of the tagging approaches have the best performance (also relative to their training effort), and if the leading taggers make use of the same, hence complete set of information.

How to Participate

The benchmark workflow consists of three main steps.

Participants are given a test dataset consisting of 200k signal and 200k background jets. The top signal and mixed quark-gluon background jets are produced with using Pythia8 with its default tune for a center-of-mass energy of 14 TeV and ignoring multiple interactions and pile-up. For a simplified detector simulation we use Delphes with the default ATLAS detector card.

The produced results should contain classification results for each jet to measure the performance of the network and test which jets are correctly classified in each approach. Overall results are sorted in decreasing order by the background rejection as signal efficiency at 50%.



ML4Jets - Top Tagger Comparison

The Machine Learning Landscape of Top Taggers

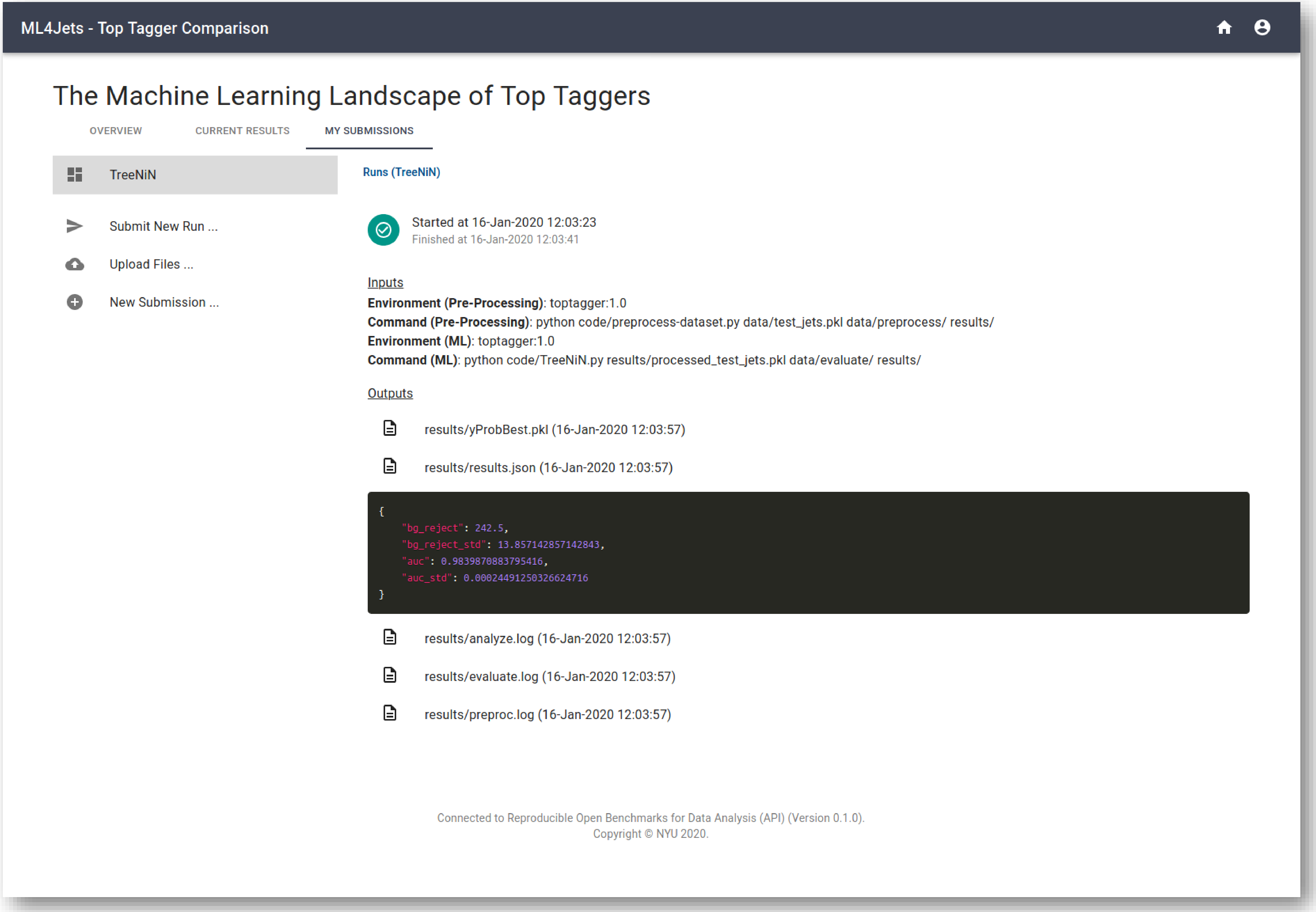
Performance Metrics

	Background rejection (at 50%)	Background rejection (STD)	AUC
TreeNiN	242.500000	13.857100	0.983987
Simple Net	99.552600	9.379710	0.958746

ROC Curves (Background Rejection at Signal Efficiency 50%) for all Algorithms

Simple Net
TreeNiN
10^4
10^3
10^2
10^1
Background rejection $\frac{1}{\varepsilon_B}$
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Signal efficiency ε_S

ROC curves for all algorithms evaluated on the test sample, shown as the background rejection ensemble median of multiple trainings.



The Machine Learning Landscape of Top Taggers

Runs (TreeNiN)

Started at 16-Jan-2020 12:03:23
Finished at 16-Jan-2020 12:03:41

Inputs

Environment (Pre-Processing): toptagger:1.0
Command (Pre-Processing): python code/preprocess-dataset.py data/test_jets.pkl data/preprocess/ results/
Environment (ML): toptagger:1.0
Command (ML): python code/TreeNiN.py results/processed_test_jets.pkl data/evaluate/ results/

Outputs

results/yProbBest.pkl (16-Jan-2020 12:03:57)

results/results.json (16-Jan-2020 12:03:57)

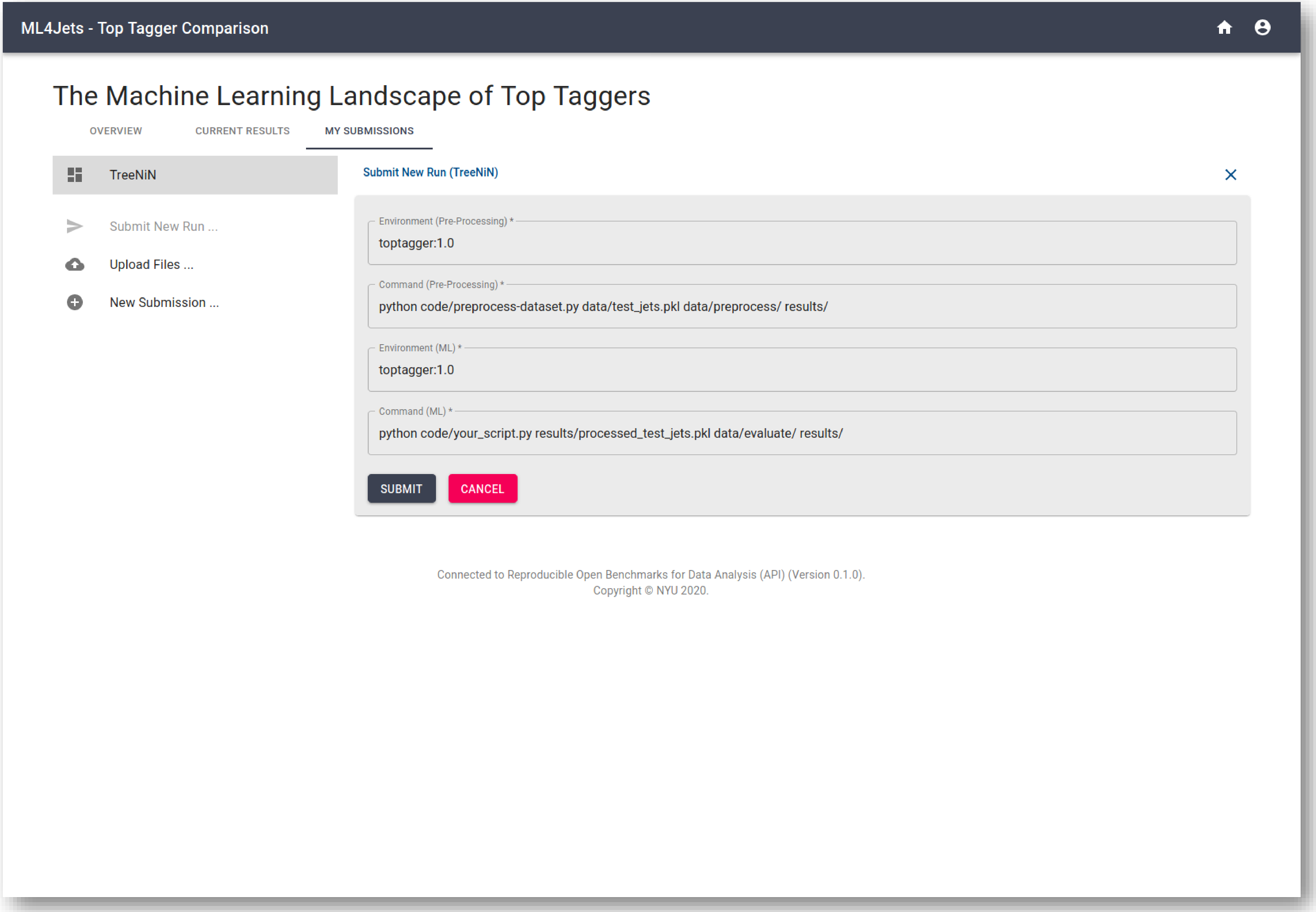
```
{
    "bg_reject": 242.5,
    "bg_reject_std": 13.857142857142843,
    "auc": 0.9839870883795416,
    "auc_std": 0.00024491250326624716
}
```

results/analyze.log (16-Jan-2020 12:03:57)

results/evaluate.log (16-Jan-2020 12:03:57)

results/preproc.log (16-Jan-2020 12:03:57)

Connected to Reproducible Open Benchmarks for Data Analysis (API) (Version 0.1.0).
Copyright © NYU 2020.



The Machine Learning Landscape of Top Taggers

Environment (Pre-Processing) *
toptagger:1.0

Command (Pre-Processing) *

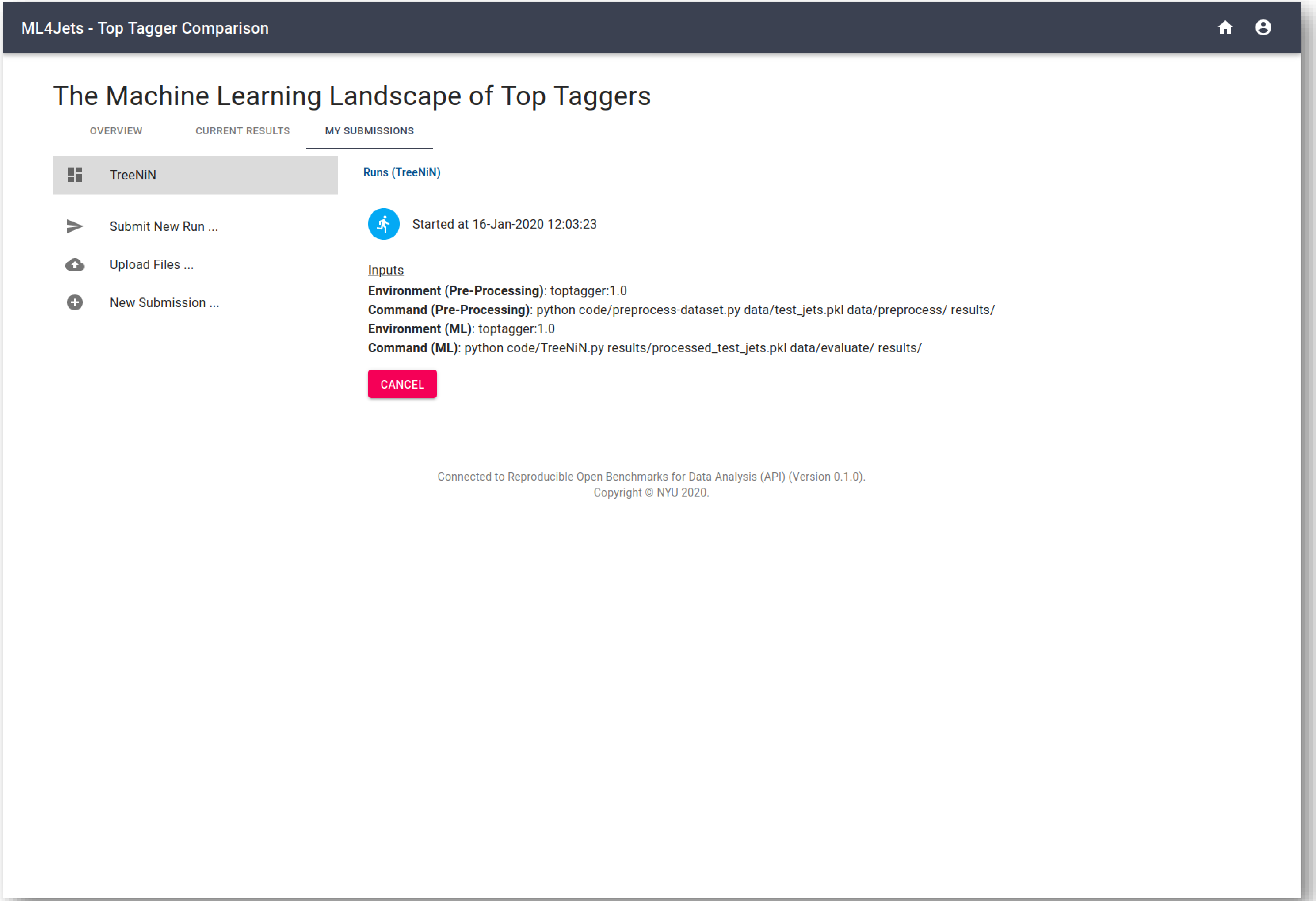
```
python code/preprocess-dataset.py data/test_jets.pkl data/preprocess/ results/
```

Environment (ML) *
toptagger:1.0

Command (ML) *

```
python code/your_script.py results/processed_test_jets.pkl data/evaluate/ results/
```

Connected to Reproducible Open Benchmarks for Data Analysis (API) (Version 0.1.0).
Copyright © NYU 2020.



The Machine Learning Landscape of Top Taggers

Started at 16-Jan-2020 12:03:23

Inputs

Environment (Pre-Processing): toptagger:1.0

Command (Pre-Processing): python code/preprocess-dataset.py data/test_jets.pkl data/preprocess/ results/

Environment (ML): toptagger:1.0

Command (ML): python code/TreeNiN.py results/processed_test_jets.pkl data/evaluate/ results/

Connected to Reproducible Open Benchmarks for Data Analysis (API) (Version 0.1.0).

Continue …

Ongoing and Future Work

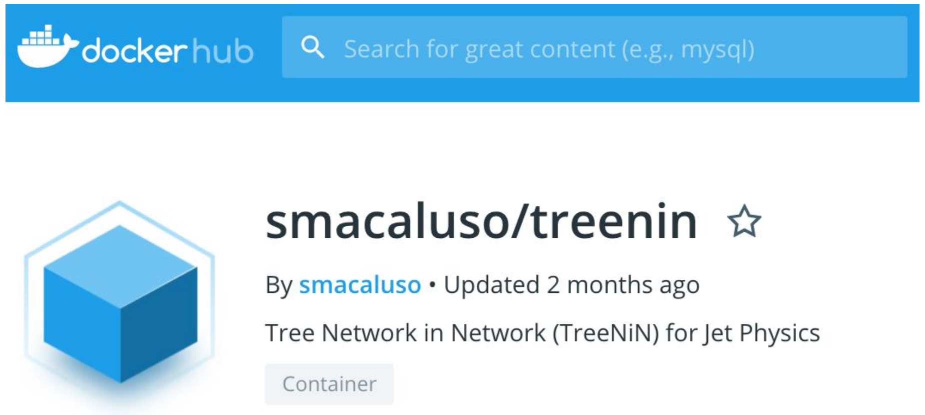
(Re-)run competition using REANA as the backend

Full Yadage Workflow for TreeNiN

https://github.com/cha-suaysom/reana-demo-treenin

Docker Container

https://github.com/diana-hep/TreeNiN
https://hub.docker.com/r/smacaluso/treenin



smacaluso/treenin ☆

By smacaluso • Updated 2 months ago

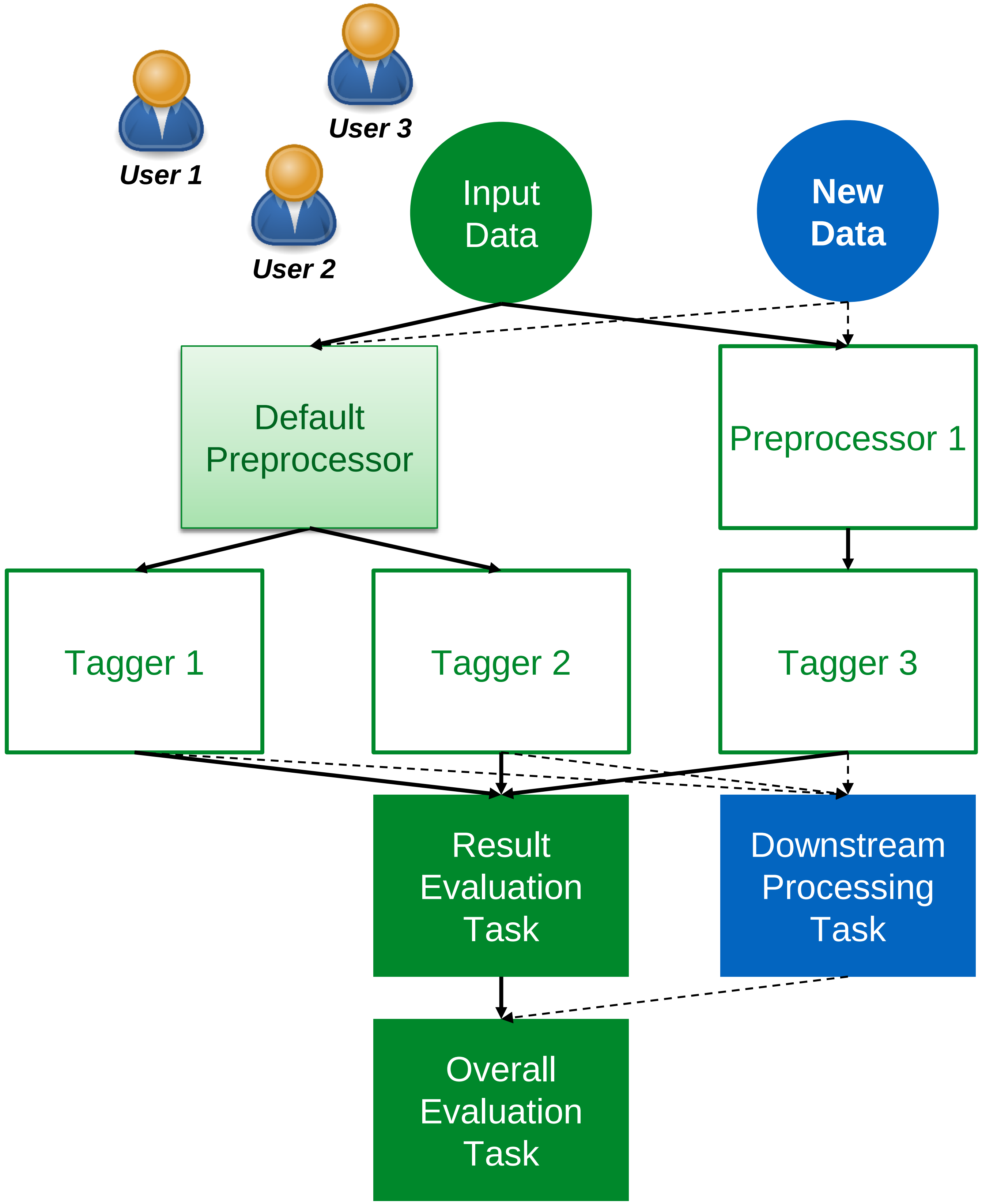
Tree Network in Network (TreeNiN) for Jet Physics

Anyone else interested?

Code Repositories for ROB

https://github.com/scailfin/rob-core
https://github.com/scailfin/rob-webapi-flask
https://github.com/scailfin/rob-ui

Ongoing and Future Work (cont.)



Other applications …

Run provided model with different input data.

Compare different input dataset and models against each other (e.g. **Standard Cortical Observer**).

Apply different/additional downstream processing tasks to the model results.