KM3TPI Hands On Session

An introduction to the project and its current status…

Michalis Chadolias

IFIC (CSIC-UV) | VEGA

Agenda

Part 1 — Technical talk

- Project overview & architecture
- Preprocessing pipeline: prepare → discover → convert → validate
- ML module: dataset, trainer, tracking
- Status & roadmap

Part 2 — Hands-on session

- Install km3tpi on your laptop
- Run the preprocessing demo notebook
- Run the ML demo notebook
- Drive the processing Snakemake pipeline end-to-end
- Q&A and feedback

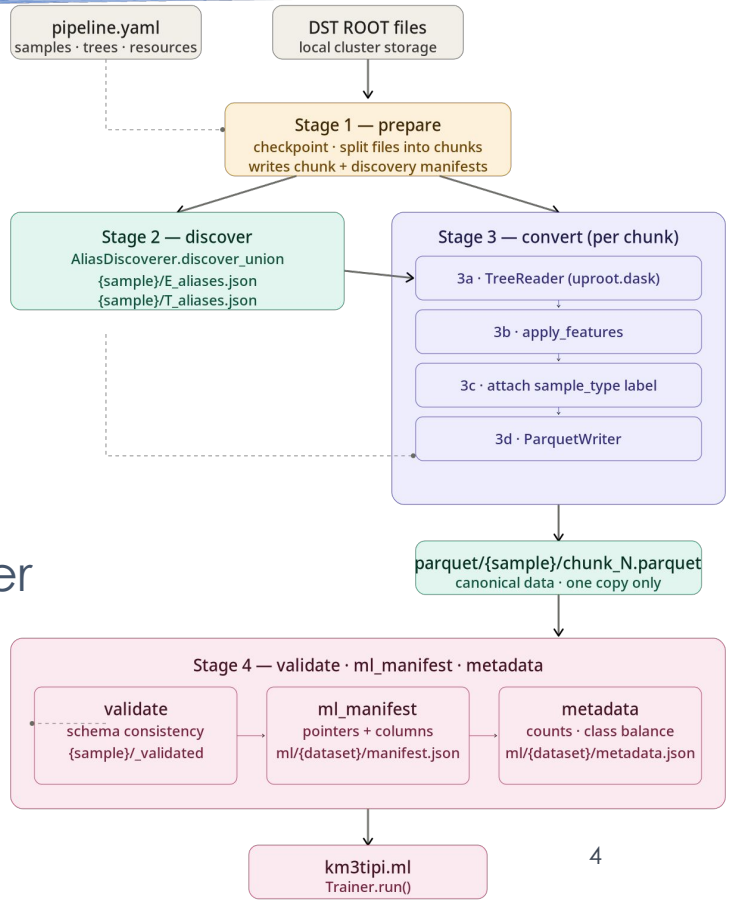
House Rules

- Ask questions any time! Interruptions are more than welcome. The point is feedback, not a polished demo.
- Honest feedback > polite feedback
- If things are confusing, or doesn't work the way you'd expect, tell me. That's why we're here.
- If you're stuck, raise your hand. We can pause a bit and look into it together.

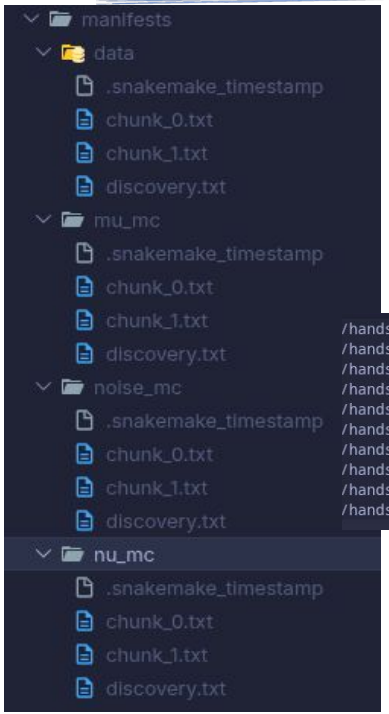
Introduction to KM3TPI

Idea Board:

- One repo to rule them all!
  - Data converter
  - Machine learning
- Agnostic approach (robust to changes)
- Modular design (km3tpi module)
  - preprocess (subpackage)
  - ml
- Snakemake compatible
  - Project maintained by one developer
  - User friendly for task delegation
- Environment handling
  - Conda
  - Singularity
- Testing & CI/CD

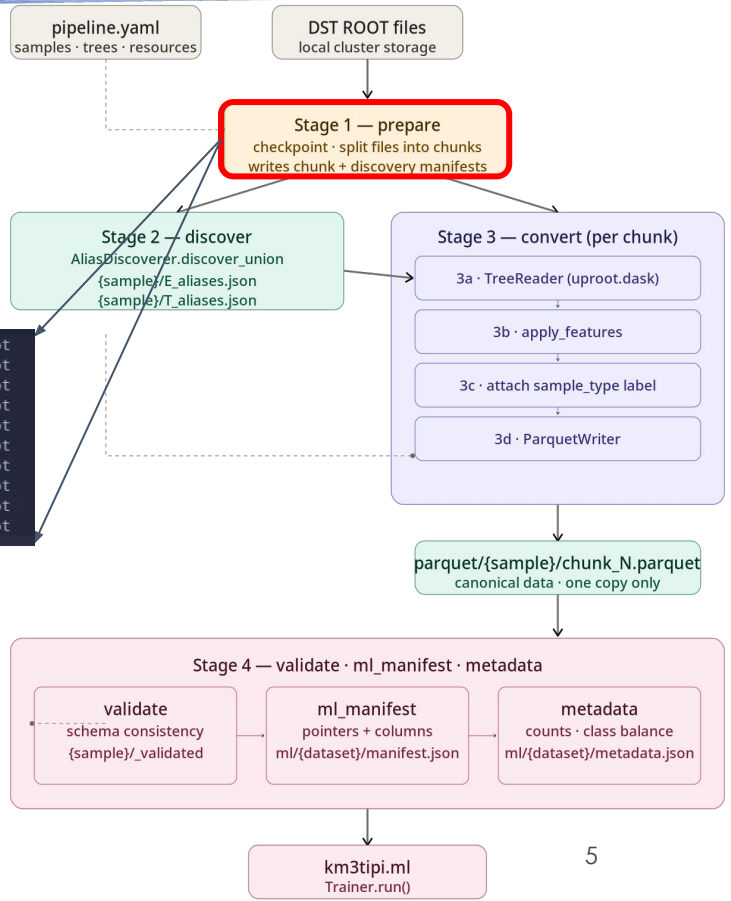


Introduction to KM3TPI



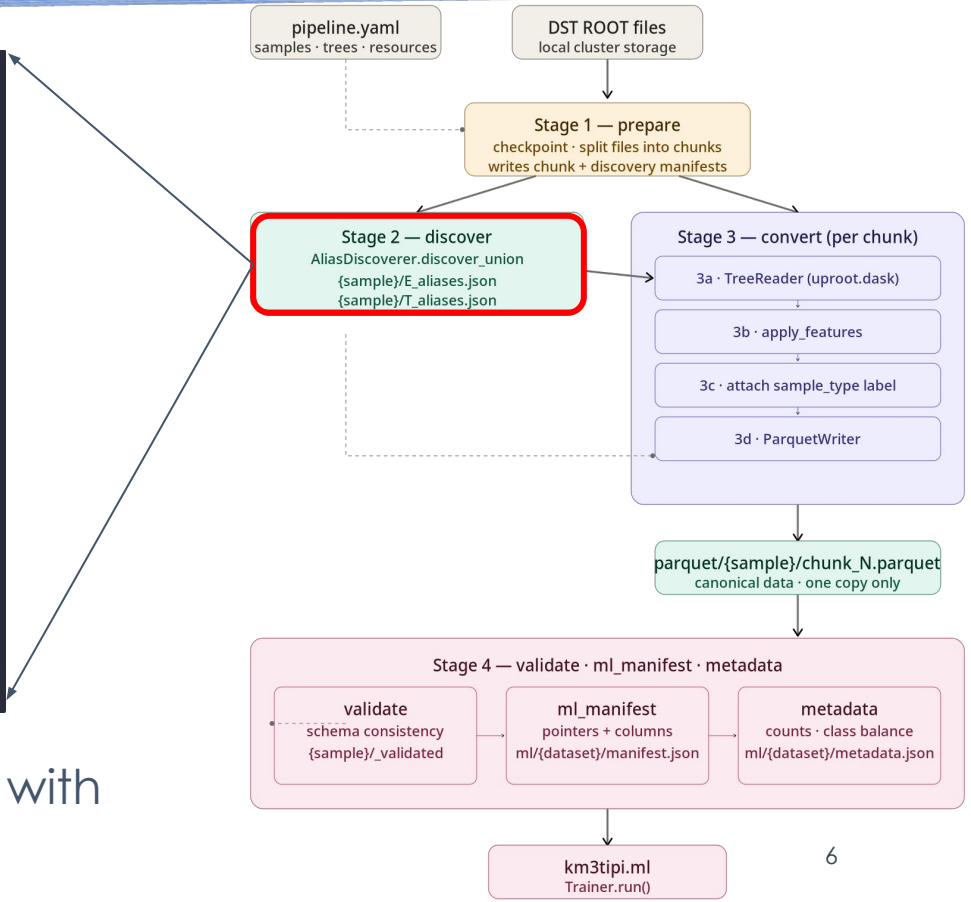

```
/hands-on-session/KM3NeT_00000117_00013853.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013859.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013868.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013871.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013877.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013883.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013906.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013929.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013948.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
/hands-on-session/KM3NeT_00000117_00013960.data.jppmuon_jppshower_dynamic.offline.dst.v9.2.root
```

Split files from the chosen directory into sublists for chunk creation



Introduction to KM3TPI


```
{
  "header": {
    "filename": "
examples/data/KM3NeT_00000117_00013762.mc.gsg_neutrinos.km3sim_sirene_merged.jterbr.jppmuon_jppshower-upgoing_static.offline.dst.v9.0.root
",
    "branch": "T",
    "skip_fields": [
      "hits"
    ],
    "filter_branch": "get_lowest",
    "filter_typename": "/^(?!.*(TObject)).*/"
  },
  "ntrks": {
    "location": "ntrks",
    "index": [],
    "preview": "12",
    "type": "int32"
  },
  "Etot": {
    "location": "Etot",
    "index": [],
    "preview": "135.60697475975138",
    "type": "float64"
  },
  "Emax": {
    "location": "Emax",
    "index": [],
    "preview": "78.3184118121905",
    "type": "float64"
  },
  "Evis": {
    "location": "Evis",
    "index": [],
    "preview": "135.60697475975138",
    "type": "float64"
  },}
```

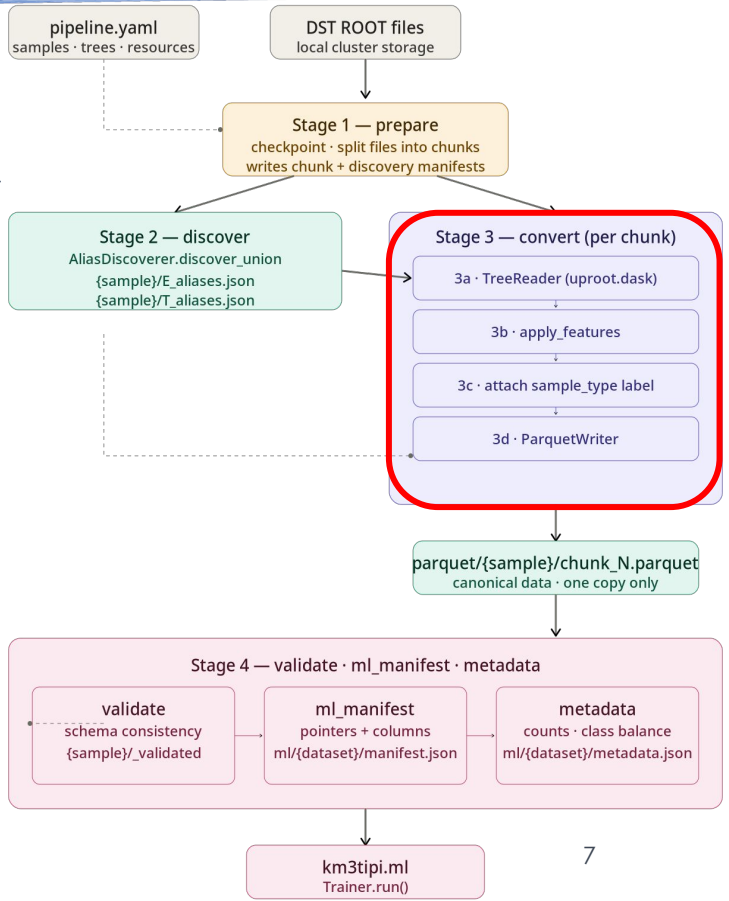



Mapping of the file's inner structure with **uproot** per tree (E,T)

Introduction to KM3TPI

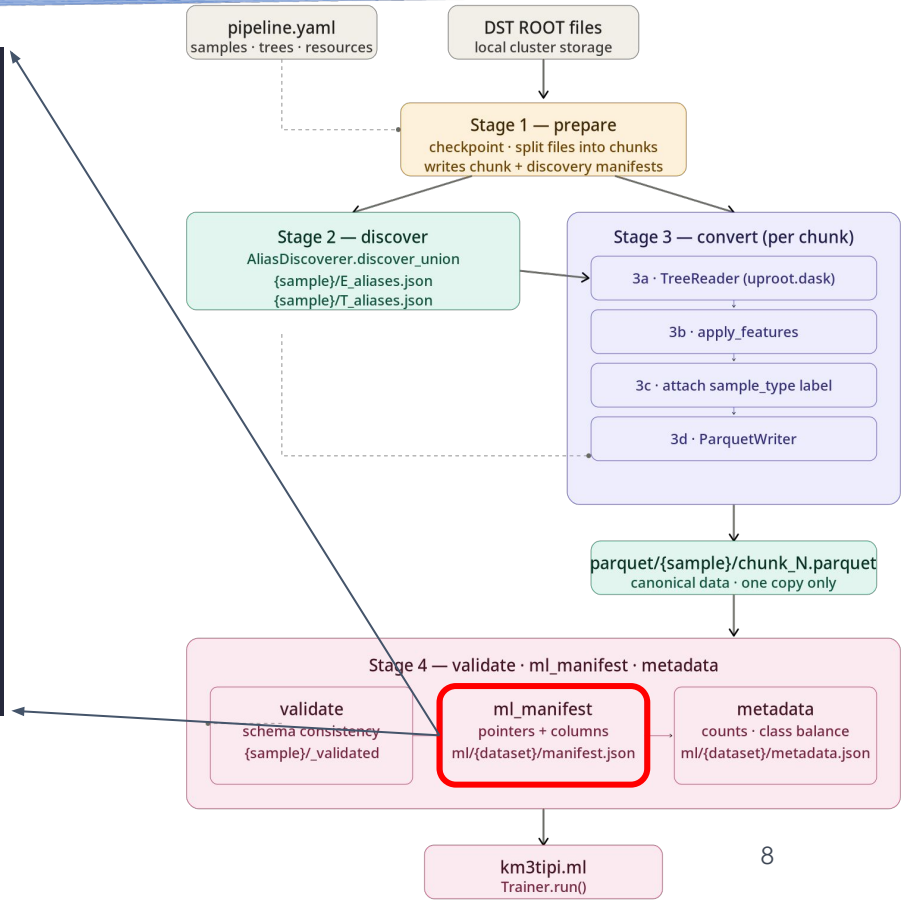
Stage 3 — Convert (per chunk)

- One fused dask graph: read → features → label → write
- Lazy by default, only the branches in the alias JSON ever touch disk
- Memory bounded per tree, not per chunk
- One canonical Parquet per chunk



Introduction to KM3TPI


```
{
  "dataset": "nu_vs_mu",
  "description": "Neutrino vs atmospheric muon classifier",
  "samples": {
    "nu_mc": {
      "path": "results/hands-on-session/parquet/nu_mc",
      "n_chunks": 2,
      "n_events": 31006,
      "all_columns": [
      ],
      "sample_type": 2
    }
  },
  "columns": [
  ],
  "n_columns": 130,
  "n_total_events": 2022386,
  "column_strategy": "common"
}
```

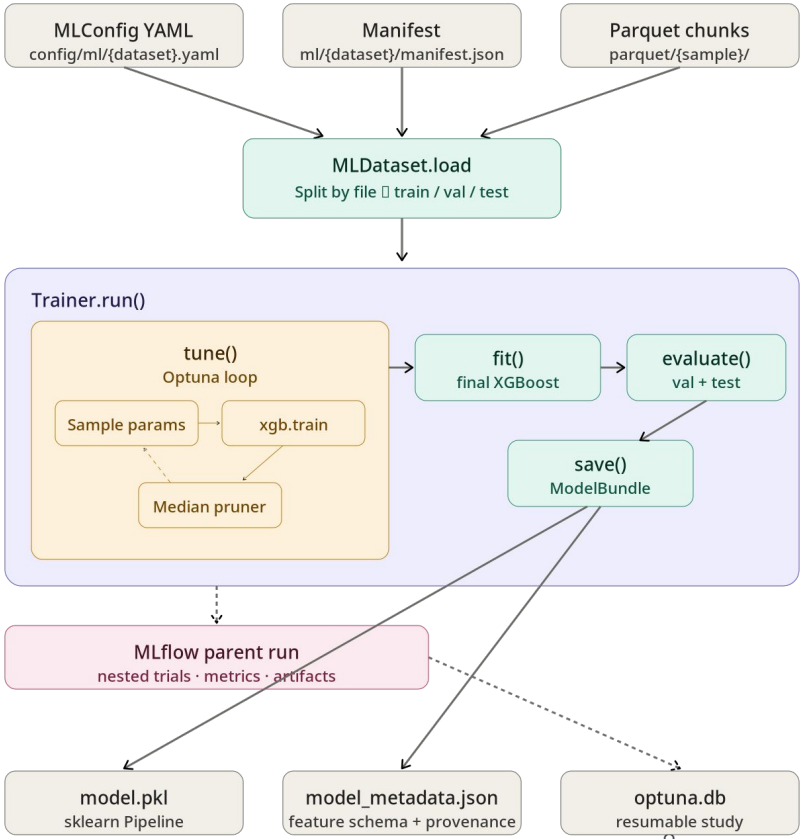
ML Module Overview

ML module — v.0.3.0

- MLProcessor (main entrypoint)
- Trainer (training & optimisation)

Design choices

- Schema versioning via metadata
- Split by file & by row
- Optuna + callback failsafes
- Class balancing: weights & multiplicity
- Decorator-based cuts (not yet)
- MLflow tracking



Summary & Next Steps

Current status:

- Preprocessing Module v1 ✅
- Preprocessing Pipeline ✅
- ML Module v1 ✅
- Documentation for both ✅
- Environment versioning ✅
- CI/CD & tests (cov: 58%) ✅
- Demo notebooks ✅

Soon to push new branch for **ml snakemake rules:**

features/ml-pipeline

Hands-On Session

Project Installation

We will work with the latest tagged version **v.0.2.1.dev0**

1. Clone the project

git clone
git@git.km3net.de:mchadolias/km3tpi.git

2. Create an environment

a. make venv
b. make install-conda (or mamba)

3. Install dev dependencies

make install-dev

4. Test everything works

make test

Dataset note

If I haven't shared the demo dataset with you, please tell me now.

Documentation

Check latest docs here

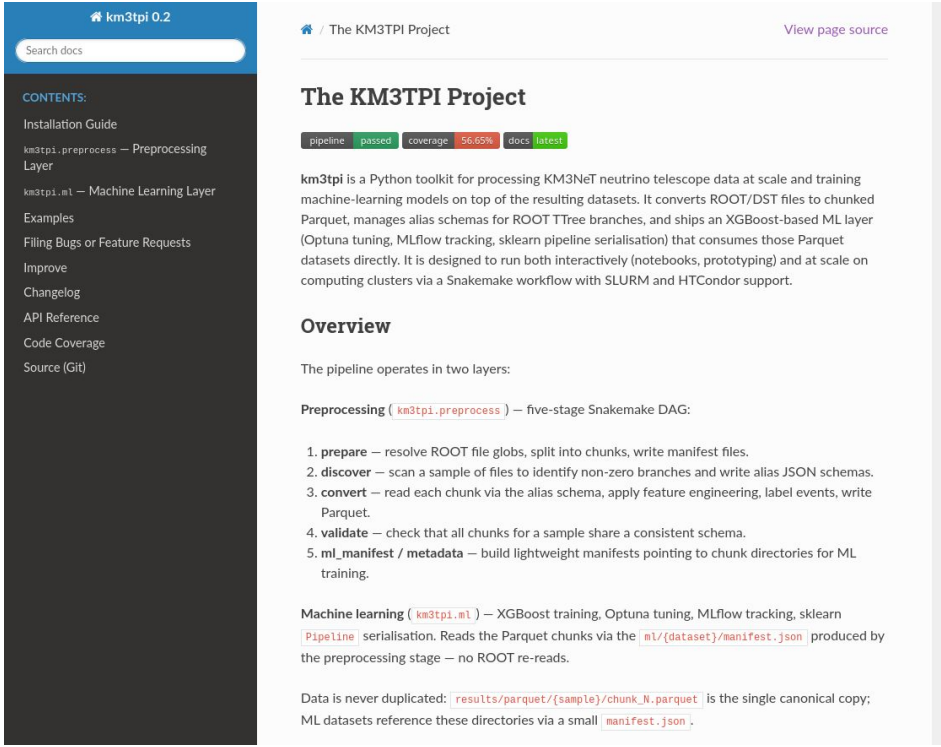
Chapters:

- Installation
- Preprocessing Layer
- ML Layer
- Examples
- API ref

To be continually updated with the latest tagged version pipeline job

Developer note

If the doc link is not working, check it via the project because it might have been transferred.



Run Simple Example

Let's test the following classes:

- AliasDiscoverer
- TreeReader
- ParquetWriter

Steps:

1. Transfer the file you want to process
2. Run command from project directory
3. ./examples/run_data_test.sh

```
echo "Running Example for KM3NeT Data"

echo "Running Aliases Schema"
echo "Step 1st: E Tree"
python examples/run_aliases.py \
      --files examples/data/KM3NeT_00000117_00013769.data.jppmuon_jppshower-upgoing_dynamic.offline.dst.v9.0.root
\
      --tree  E \
      --output examples/aliases/data_E_aliases.json \
      --skip  hits mc_trks\
      --filter tobject

echo "Step 2nd: T Tree"
python examples/run_aliases.py \
      --files examples/data/KM3NeT_00000117_00013769.data.jppmuon_jppshower-upgoing_dynamic.offline.dst.v9.0.root
\
      --tree  T \
      --output examples/aliases/data_T_aliases.json \
      --skip  hits mc_trks\
      --filter tobject

echo "Step 3rd: Export"
python examples/run_root_to_export.py \
      --files examples/data/KM3NeT_00000117_00013769.data.jppmuon_jppshower-upgoing_dynamic.offline.dst.v9.0.root
\
      --alias-E-json examples/aliases/data_E_aliases.json \
      --alias-T-json examples/aliases/data_T_aliases.json \
      --output examples/parquet/data_test.parquet \
      --sample-type data
```

Switch ROOT filename and run for other types

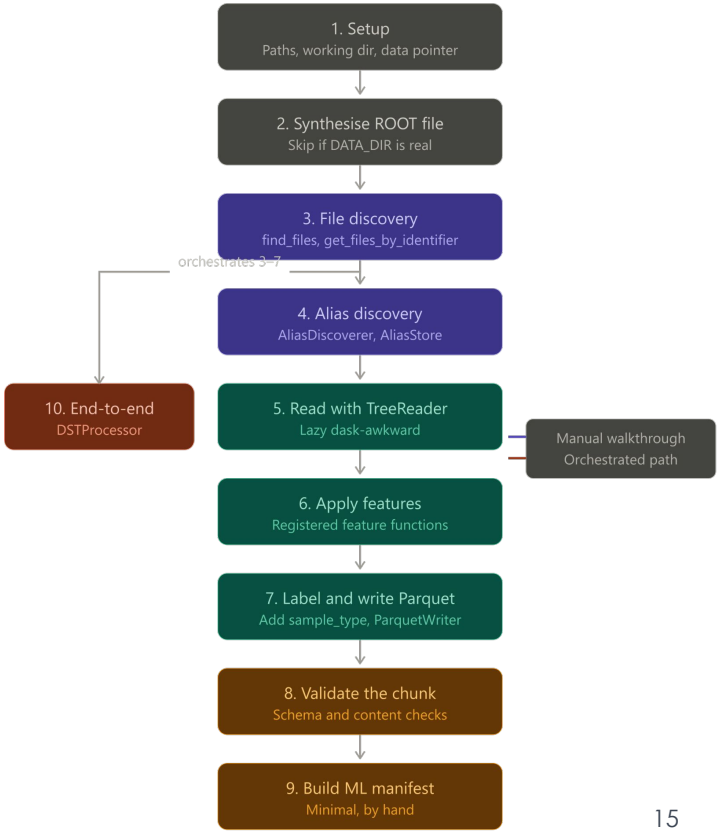
Run Preprocess Notebook

Check the *demo_processing.ipynb*

```
1. Setup — paths, working dir, optional pointer to real data
2. Synthesise a tiny ROOT file (skip if you point `DATA_DIR` at real ones)
3. File discovery — `find_files`, `get_files_by_identifier`
4. Alias discovery — `AliasDiscoverer` + `AliasStore`
5. Read with `TreeReader` (lazy dask-awkward)
6. Apply registered features
7. Add the `sample_type` label and write Parquet
8. Validate the chunk
9. Build a minimal ML manifest by hand
10. End-to-end with `DSTProcessor`

Self-contained: synthesises ROOT files if none are provided.
```

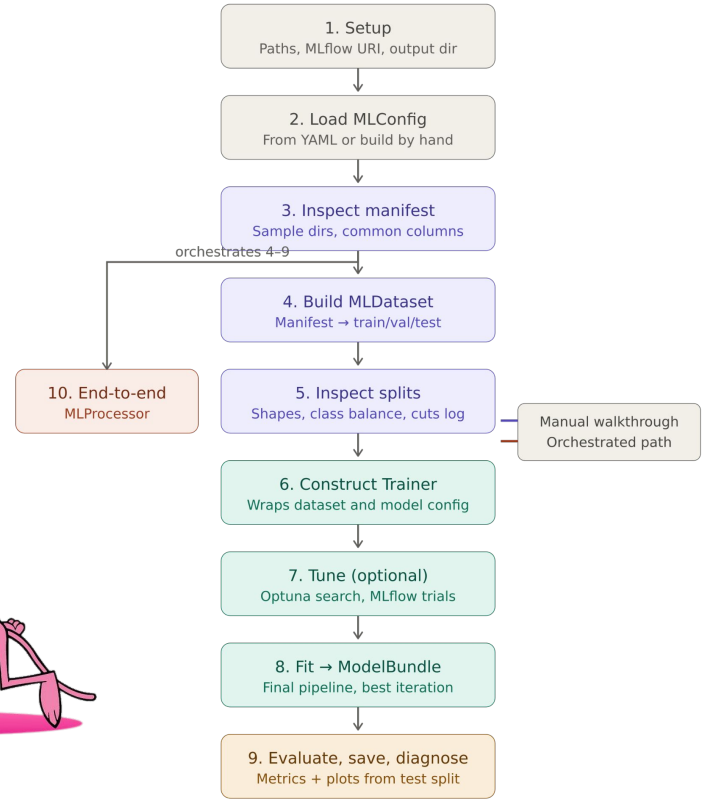
If you want to use a subset of the provided DSTs

```
WORK = Path("./_preprocess_demo").resolve()
if WORK.exists():
    shutil.rmtree(WORK)
WORK.mkdir(parents=True)

# If you have a real small dataset, set DATA_DIR to its directory and
# the synthesis step below will be skipped.
DATA_DIR: Path | None = None     # e.g. Path("/sps/km3net/.../small_subset")
print("WORK     :", WORK)
print("DATA_DIR :", DATA_DIR)
```



Run Machine Learning Notebook

```python
dataset_cfg = DatasetConfig(
    manifest_path=manifest_path,
    features=[
        "trks.lik_0", "trks.E_0", "trks.dir.z_0",
        "n_hits", "n_trig_hits",
    ],
    target="sample_type",
    weight_column="weight",          # set None to disable per-event weights
    exclude_samples=["data"],        # default — drops any sample with "data" in name
    cuts=[],                         # registry empty in M1
)
print(json.dumps(dataset_cfg.to_dict(), indent=2, default=str))
```



If you want to use a subset of the files that you produced in the previous, define the ML_MANIFEST

Snakemake Guide

snakemake

VEGA

- **Workflow management system**
- Snakemake is designed such that:
  - **Reproducibility** is easily implemented
  - It handles the communication with the **HPC cluster**
  - Best practices with little effort
- Quick installation with conda/mamba
- Check more here for documentation and tutorials [1,2]
- Most important file in a Snakemake workflow is the **Snakefile**
- A rule describes how to get one or more output files from input files

```
rule discover:
    input:
        manifest = str(MANIFEST_DIR / "{sample}" / "discovery.txt"),
    output:
        alias_json = str(ALIAS_DIR / "{sample}" / "{tree}_aliases.json"),
    params:
        n_preview   = config["discovery"]["n_preview_files"],
        sampling    = config["discovery"].get("sampling_strategy", "spread"),
        seed        = config["discovery"].get("seed", 42),
        skip_fields = lambda wc: " ".join(
            config["samples"][wc.sample].get("skip_fields", ["hits"])
        ),
    log:
        str(LOG_DIR / "discover" / "{sample}_{tree}.log"),
    benchmark:
        str(BENCHMARK_DIR / "discover" / "{sample}_{tree}.tsv")
    retries: 2
    shell:
        """
        mkdir -p $(dirname {output.alias_json}) $(dirname {log})

        python scripts/discover.py \
            --manifest    {input.manifest}       \
            --tree        {wildcards.tree}       \
            --sample      {wildcards.sample}     \
            --output      {output.alias_json}    \
            --n-preview   {params.n_preview}     \
            --sampling    {params.sampling}       \
            --seed        {params.seed}           \
            --skip-fields {params.skip_fields}   \
        > {log} 2>&1
        """
```

Snakemake Guide

snakemake

VEGA

- The user requests Snakemake to produce files via the command line
- The wildcards “sample” & “tree” has to be filled with a concrete value (in this example a data & T)

```
rule discover:
    input: results/hands-on-session/manifests/data/discovery.txt
    output: results/hands-on-session/aliases/data/T_aliases.json
    log: results/hands-on-session/logs/discover/data_T.log
    jobid: 19
    benchmark: results/hands-on-session/benchmarks/discover/data_T.tsv
    reason: Missing output files: results/hands-on-session/aliases/data/T_aliases.json
    wildcards: sample=data, tree=T
    resources: tmpdir=/tmp, disk_mb=1000, disk=1 GB, disk_mib=954, mem_mb=2000, mem=2 GB, mem_mib=1908, runtime=15
```

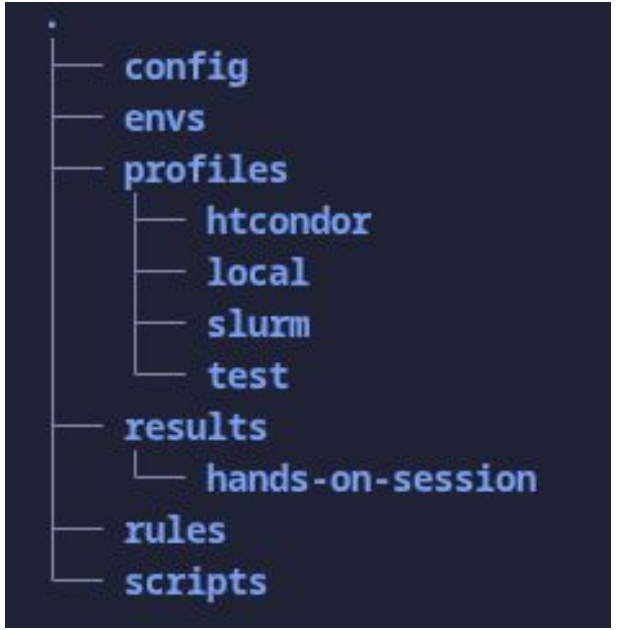
```
rule discover:
    input:
        manifest = str(MANIFEST_DIR / "{sample}" / "discovery.txt"),
    output:
        alias_json = str(ALIAS_DIR / "{sample}" / "{tree}_aliases.json"),
    params:
        n_preview   = config["discovery"]["n_preview_files"],
        sampling    = config["discovery"].get("sampling_strategy", "spread"),
        seed        = config["discovery"].get("seed", 42),
        skip_fields = lambda wc: " ".join(
            config["samples"][wc.sample].get("skip_fields", ["hits"])
        ),
    log:
        str(LOG_DIR / "discover" / "{sample}_{tree}.log"),
    benchmark:
        str(BENCHMARK_DIR / "discover" / "{sample}_{tree}.tsv")
    retries: 2
    shell:
        """
        mkdir -p $(dirname {output.alias_json}) $(dirname {log})

        python scripts/discover.py \
            --manifest    {input.manifest}       \
            --tree        {wildcards.tree}       \
            --sample      {wildcards.sample}     \
            --output      {output.alias_json}    \
            --n-preview   {params.n_preview}     \
            --sampling    {params.sampling}       \
            --seed        {params.seed}           \
            --skip-fields {params.skip_fields}   \
        > {log} 2>&1
        """
```

- A second request to produce the file is not executed since “nothing to be done”

Run Snakemake Preprocess

The folder `pipeline` houses all workflow related files:

- Snakefile: main orchestrator file
- rules: contains the individual rules
- scripts: entrypoint scripts
- profiles: job & resource allocation
- config: user-defined pipeline options



Today the testing profile will be used, a scaled down version of the local one!

Run Snakemake Configuration

```yaml
detector_id: "hands-on-session"
base_dir: "/run/media/mchadolias/SteamML/Datasets/KM3NeT/ml_training/"

samples:
  data:
    root_files: "{base_dir}/{detector_id}/*.root"
    identifiers: ["data"]
    skip_fields: ["hits", "mc_trks"]
    sample_type: "data"
    n_jobs: 2

  nu_mc:
    root_files: "{base_dir}/{detector_id}/*.root"
    identifiers: ["neutrinos", "gsg_neutrinos"]
    skip_fields: ["hits"]
    sample_type: "nu_mc"
    n_jobs: 2

n_jobs: 8 # fallback if a sample omits n_jobs

# ── Alias Discovery ──────────────────────────────────────
trees: ["E", "T"]

discovery:
  n_preview_files: 4
  sampling_strategy: "spread" # "first" | "spread" | "random"
  seed: 42

# ── Parquet Writer ───────────────────────────────────────
writer:
  row_group_size: 250000
  extensionarray: false
  log_memory: true

# ── Feature Engineering ──────────────────────────────────
# Empty list = raw export only. Add feature names once registered:
#   E: ["cos_zenith", "log_energy"]

features:
  E: []
  T: []

# ── ML Datasets ──────────────────────────────────────────

ml:
  datasets:
    nu_vs_mu:
      samples: ["nu_mc", "mu_mc"]
      columns: "common"
      description: "Neutrino vs atmospheric muon classifier"

# ── Dask (per-job, in-process) ───────────────────────────

dask:
  scheduler: "synchronous" # "synchronous" | "threads" | "processes"
  n_workers: 1
  threads_per_worker: 1
```

define folder path & classes

which trees to map and & how to export them

do I want to add new features?

What to modify:

- data directory
- writer
- skip fields
- n_jobs
- discovery

Run Snakemake locally

How can I run the example?

User-defined configs

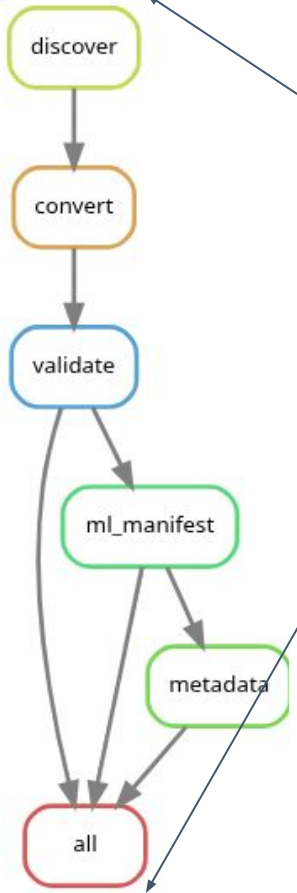
snakemake --profile profiles/local --configfile config/pipeline.yaml -n

Scheduler config

dry run flag

Tip of the day:
Always run with the dry run flag first!

Snakemake DAG & job stats




```
Job stats:
job            count
-----------  -------
all                1
metadata           2
ml_manifest        2
prepare            4
validate           4
total             13
```

Prepare = checkpoint: inspects data, then expands jobs

Run Snakemake on a cluster (Lyon)

How can I run the example, in Lyon?

User-defined configs

snakemake --profile profiles/slurm --configfile config/pipeline.yaml -n

Scheduler config

dry run flag

Tip of the day:
Create a profile for the specific scheduler and go ahead

What you should have by the end

If something doesn't work:

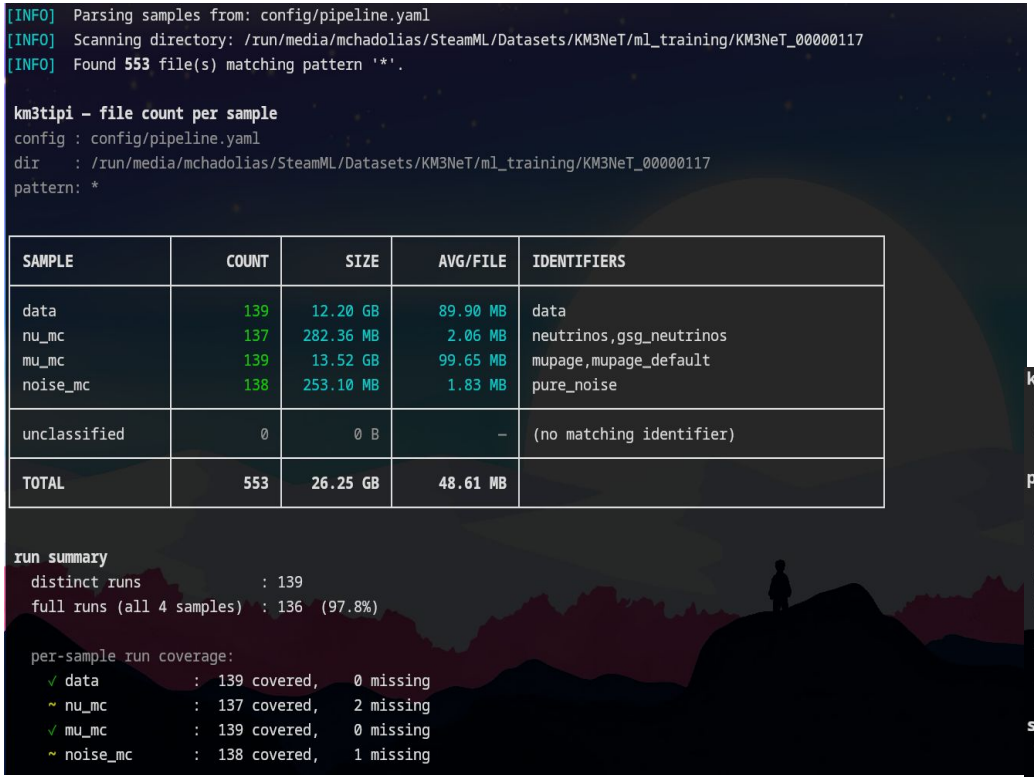
- Catch me during the session, the feedback is the point!
- Open an issue on git.km3net.de/mchadolias/km3tpi

By the end of the session you should have:

- km3tpi cloned and installed in a conda env on your machine
- `make test` passing locally
- Run the preprocess notebook end-to-end (ROOT → Parquet)
- Run the ML notebook (Parquet → trained XGBoost model)
- Executed at least one full Snakemake run on the demo dataset

Back-Up Slides

Preprocessing benchmarks: full ORCA13



```
[INFO]  Parsing samples from: config/pipeline.yaml
[INFO]  Scanning directory: /run/media/mchadolias/SteamML/Datasets/KM3NeT/ml_training/KM3NeT_00000117
[INFO]  Found 553 file(s) matching pattern '*'.

km3tipi — file count per sample
config : config/pipeline.yaml
dir    : /run/media/mchadolias/SteamML/Datasets/KM3NeT/ml_training/KM3NeT_00000117
pattern: *
```

SAMPLE	COUNT	SIZE	AVG/FILE	IDENTIFIERS
data	139	12.20 GB	89.90 MB	data
nu_mc	137	282.36 MB	2.06 MB	neutrinos,gsg_neutrinos
mu_mc	139	13.52 GB	99.65 MB	mupage,mupage_default
noise_mc	138	253.10 MB	1.83 MB	pure_noise
unclassified	0	0 B	—	(no matching identifier)
TOTAL	553	26.25 GB	48.61 MB	

```
run summary
  distinct runs             : 139
  full runs (all 4 samples) : 136  (97.8%)

  per-sample run coverage:
    √ data          :  139 covered,    0 missing
    ~ nu_mc         :  137 covered,    2 missing
    √ mu_mc         :  139 covered,    0 missing
    ~ noise_mc      :  138 covered,    1 missing
```

Successful execution the pipeline in work laptop in less than 5 minutes!

```
km3tipi monitor  — 2026-05-03 22:06:25
 config:  config/pipeline.yaml
 results: results/KM3NeT_00000117

pipeline  (done/total — = all done · * running · X has failed · . pending)
                        prepar  discov  conver  valida  ml_man  metada
 data                   1/1 =   2/2 =   7/8 .   1/1 =   -       -
 nu_mc                  1/1 =   2/2 =   5/5 =   1/1 =   -       -
 mu_mc                  1/1 =   2/2 =   8/8 =   1/1 =   -       -
 noise_mc               1/1 =   2/2 =   5/5 =   1/1 =   -       -

 nu_vs_mu               -       -       -       -       1/1 =   1/1 =
 signal_vs_background   -       -       -       -       1/1 =   1/1 =

scheduler  (none, user=mchadolias)
 no scheduler available — pipeline view only
```

Testing Status

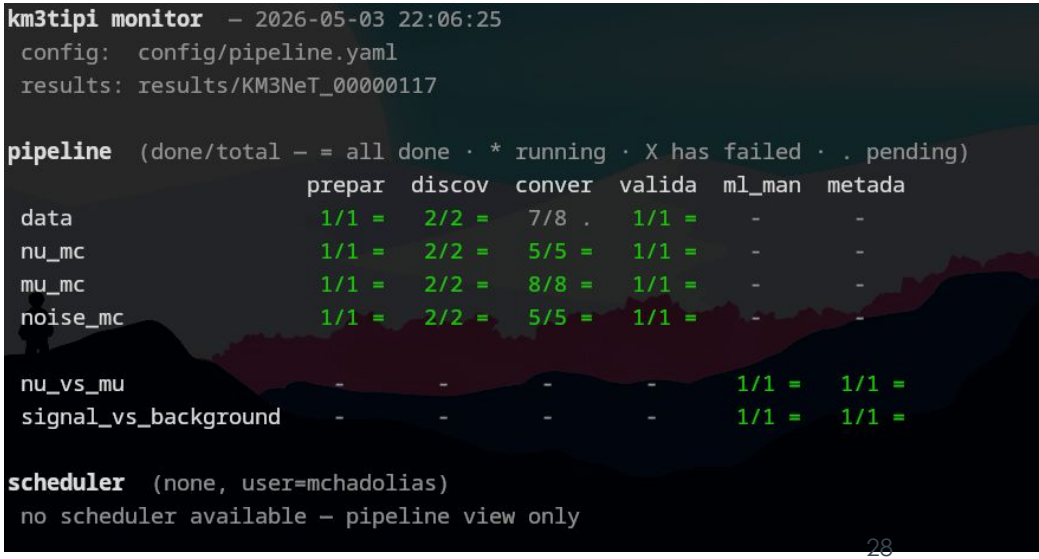
```
============================= tests coverage ==============================
_____ coverage: platform linux, python 3.12.13-final-0 _____

Name                                Stmts   Miss  Cover   Missing
-----------------------------------------------------------------
src/km3tpi/__init__.py                  7      0   100%
src/km3tpi/config.py                   48      2    96%   47, 53
src/km3tpi/ml/__init__.py              11      0   100%
src/km3tpi/ml/config.py               118      1    99%   132
src/km3tpi/ml/dataset.py              179     24    87%   170, 197-198, 203, 226, 261, 339-344, 392, 396, 403, 413, 489-500
src/km3tpi/ml/evaluate.py             155    139    10%   73-100, 121-123, 139, 171-206, 228-256, 288-320, 337-354, 385-409, 442-469
src/km3tpi/ml/inference.py             84     41    51%   63-67, 71-86, 98, 101-111, 115-122, 129-132, 140-141
src/km3tpi/ml/models.py                58      2    97%   45, 91
src/km3tpi/ml/preselect.py             65      0   100%
src/km3tpi/ml/processor.py            164    139    15%   67-72, 76, 81-95, 98-101, 107-119, 124-139, 144-148, 158-182, 193-227, 230-245, 252-282, 286-313
src/km3tpi/ml/splits.py                40      1    98%   144
src/km3tpi/ml/tracking.py             110     88    20%   20-25, 29-34, 50-62, 66-71, 75-83, 88-102, 106-110, 114-122, 131-150, 162-181, 186-192
src/km3tpi/ml/train.py                 94     77    18%   58-69, 87-108, 121-161, 165-166, 170-178, 186-211, 228-238
src/km3tpi/ml/tune.py                 132    132     0%   10-317
src/km3tpi/preprocess/__init__.py       7      0   100%
src/km3tpi/preprocess/aliases.py      168    118    30%   107-205, 238-269, 318, 330-341, 355-359, 367-381, 389-399, 406-413
src/km3tpi/preprocess/config.py       127      2    98%   274-275
src/km3tpi/preprocess/features.py      59      0   100%
src/km3tpi/preprocess/pipeline.py     211     35    83%   199-206, 230-231, 256-268, 569-570, 611-612, 673-678, 743, 755-757, 764-769, 777
src/km3tpi/preprocess/reader.py        71     11    85%   71, 85-87, 92-96, 117-121
src/km3tpi/preprocess/writer.py       119     22    82%   109-110, 143-145, 165, 186, 193-195, 229, 239-240, 265-266, 281-282, 288-292
src/km3tpi/utils/__init__.py            4      0   100%
src/km3tpi/utils/cli.py                24     20    17%   16-48
src/km3tpi/utils/logger.py             53     29    45%   59-69, 74-77, 105, 127-152
src/km3tpi/utils/utils.py              45     32    29%   34-35, 55-58, 81, 98, 132-148, 174-189
-----------------------------------------------------------------
TOTAL                                2153    915    58%
Coverage HTML written to dir reports/coverage
Coverage XML written to file reports/coverage.xml
============================== 310 passed in 6.54s ==============================
```

Monitoring

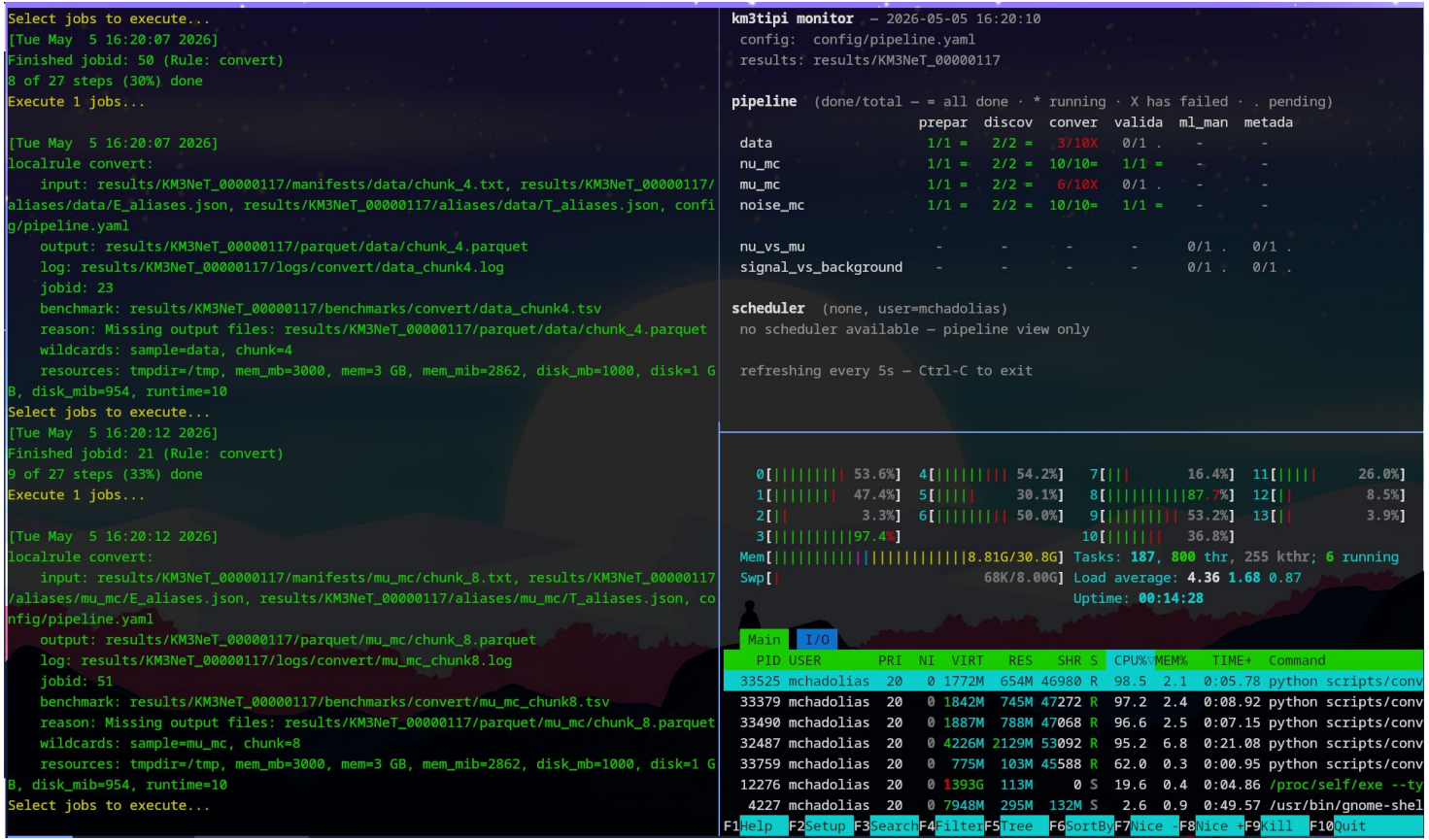
- Supplementary to Snakemake
- Dedicated monitoring script
  - checks pipeline steps
  - submitted jobs

Check cluster tool project!

- Config file required for the snakemake pipeline
- Will display per-job resource usage (CPU/mem/disk)

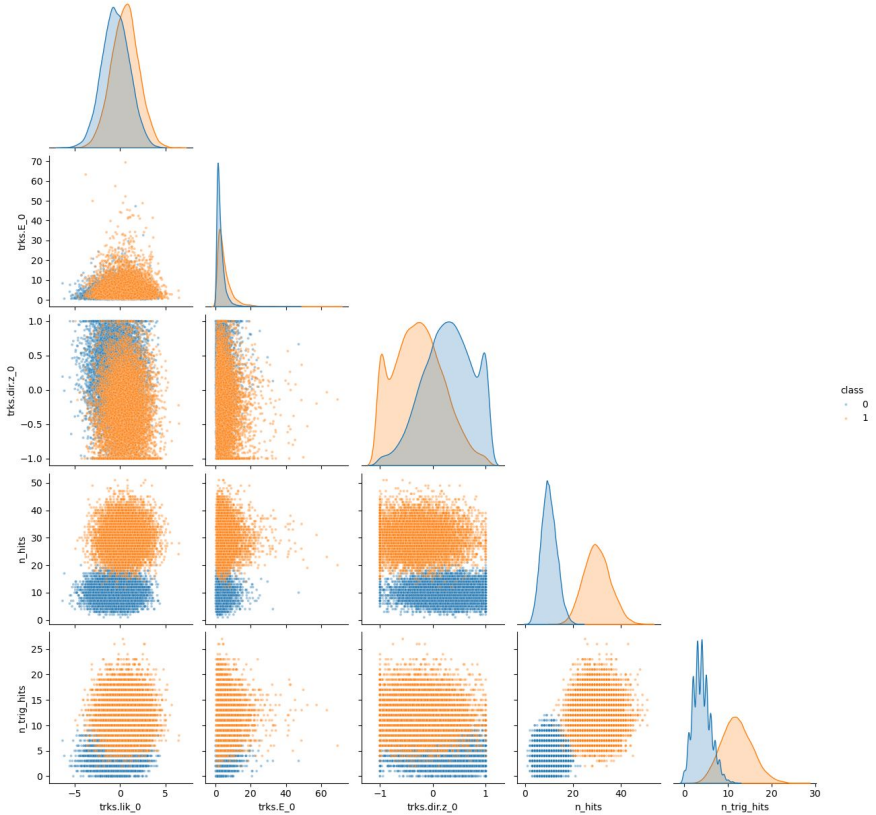


Monitoring


Select jobs to execute...
[Tue May 5 16:20:07 2026]
Finished jobid: 50 (Rule: convert)
8 of 27 steps (30%) done
Execute 1 jobs...
[Tue May 5 16:20:07 2026]
localrule convert:
input: results/KM3NeT_00000117/manifests/data/chunk_4.txt, results/KM3NeT_00000117/aliases/data/E_aliases.json, results/KM3NeT_00000117/aliases/data/T_aliases.json, config/pipeline.yaml
output: results/KM3NeT_00000117/parquet/data/chunk_4.parquet
log: results/KM3NeT_00000117/logs/convert/data_chunk4.log
jobid: 23
benchmark: results/KM3NeT_00000117/benchmarks/convert/data_chunk4.tsv
reason: Missing output files: results/KM3NeT_00000117/parquet/data/chunk_4.parquet
wildcards: sample=data, chunk=4
resources: tmpdir=/tmp, mem_mb=3000, mem=3 GB, mem_mib=2862, disk_mb=1000, disk=1 GB, disk_mib=954, runtime=10
Select jobs to execute...
[Tue May 5 16:20:12 2026]
Finished jobid: 21 (Rule: convert)
9 of 27 steps (33%) done
Execute 1 jobs...
[Tue May 5 16:20:12 2026]
localrule convert:
input: results/KM3NeT_00000117/manifests/mu_mc/chunk_8.txt, results/KM3NeT_00000117/aliases/mu_mc/E_aliases.json, results/KM3NeT_00000117/aliases/mu_mc/T_aliases.json, config/pipeline.yaml
output: results/KM3NeT_00000117/parquet/mu_mc/chunk_8.parquet
log: results/KM3NeT_00000117/logs/convert/mu_mc_chunk8.log
jobid: 51
benchmark: results/KM3NeT_00000117/benchmarks/convert/mu_mc_chunk8.tsv
reason: Missing output files: results/KM3NeT_00000117/parquet/mu_mc/chunk_8.parquet
wildcards: sample=mu_mc, chunk=8
resources: tmpdir=/tmp, mem_mb=3000, mem=3 GB, mem_mib=2862, disk_mb=1000, disk=1 GB, disk_mib=954, runtime=10
Select jobs to execute...
km3tipi monitor — 2026-05-05 16:20:10
config: config/pipeline.yaml
results: results/KM3NeT_00000117
pipeline (done/total — = all done · * running · X has failed · . pending)
prepar discov conver valida ml_man metada
data 1/1 = 2/2 = 3/10X 0/1 . - -
nu_mc 1/1 = 2/2 = 10/10= 1/1 = - -
mu_mc 1/1 = 2/2 = 6/10X 0/1 . - -
noise_mc 1/1 = 2/2 = 10/10= 1/1 = - -
nu_vs_mu - - - - 0/1 . 0/1 .
signal_vs_background - - - - 0/1 . 0/1 .
scheduler (none, user=mchadolias)
no scheduler available — pipeline view only
refreshing every 5s — Ctrl-C to exit
0[|||||||| 53.6%] 4[|||||||||| 54.2%] 7[||| 16.4%] 11[||||| 26.0%]
1[|||||||| 47.4%] 5[|||| 30.1%] 8[|||||||||87.7%] 12[|| 8.5%]
2[|| 3.3%] 6[|||||||||| 50.0%] 9[|||||||| 53.2%] 13[|| 3.9%]
3[|||||||||97.4%] 10[|||||| 36.8%]
Mem[||||||||||||||||||8.81G/30.8G] Tasks: 187, 800 thr, 255 kthr; 6 running
Swp[| 68K/8.00G] Load average: 4.36 1.68 0.87
Uptime: 00:14:28
Main I/O
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
33525 mchadolias 20 0 1772M 654M 46980 R 98.5 2.1 0:05.78 python scripts/conv
33379 mchadolias 20 0 1842M 745M 47272 R 97.2 2.4 0:08.92 python scripts/conv
33490 mchadolias 20 0 1887M 788M 47068 R 96.6 2.5 0:07.15 python scripts/conv
32487 mchadolias 20 0 4226M 2129M 53092 R 95.2 6.8 0:21.08 python scripts/conv
33759 mchadolias 20 0 775M 103M 45588 R 62.0 0.3 0:00.95 python scripts/conv
12276 mchadolias 20 0 1393G 113M 0 S 19.6 0.4 0:04.86 /proc/self/exe --ty
4227 mchadolias 20 0 7948M 295M 132M S 2.6 0.9 0:49.57 /usr/bin/gnome-shel
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice - F8Nice + F9Kill F10Quit

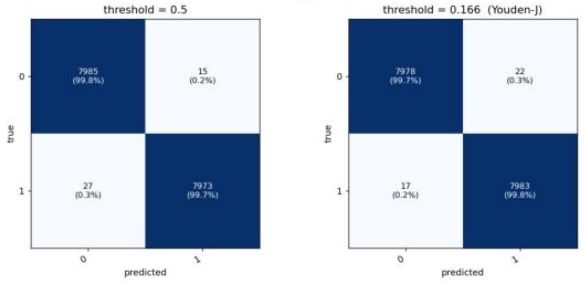
ML Test with synthetic data

- train/val/test datasets
- 16k events per set
- Class balance: 1:1
- 5 “features”:
  - “trks.lik_0"
  - "trks.E_0"
  - "trks.dir.z_0"
  - "n_hits"
  - "n_trig_hits"

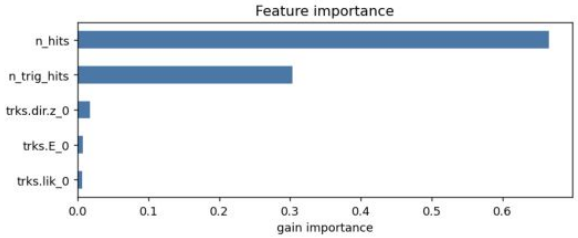


Diagnostics

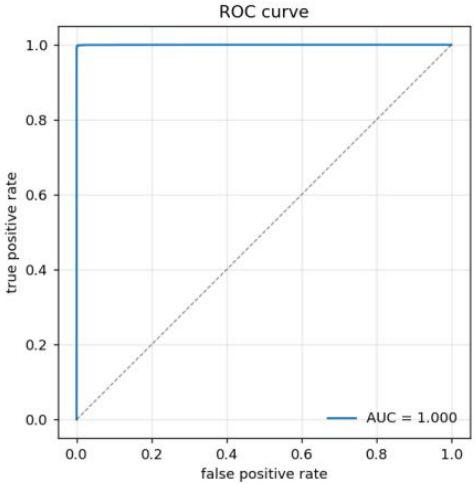
confusion_matrix



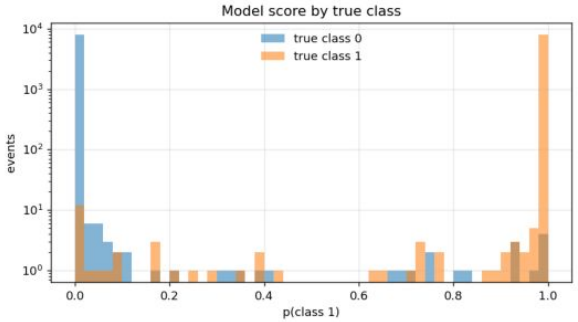
feature_importance



roc_curve



score_histogram




```
{
  "feature_names": [
    "trks.lik_0",
    "trks.E_0",
    "trks.dir.z_0",
    "n_hits",
    "n_trig_hits"
  ],
  "target": "sample_type",
  "objective": "binary:logistic",
  "num_classes": null,
  "class_mapping": {
    "0": 0,
    "1": 1
  },
  "xgb_params": {
    "tree_method": "hist",
    "n_estimators": 117,
    "max_depth": 5,
    "learning_rate": 0.22648248189516848,
    "n_jobs": 1,
    "random_state": 0,
    "subsample": 0.892797576724562,
    "colsample_bytree": 0.8394633936788146,
    "min_child_weight": 0.20513382630874505,
    "reg_alpha": 2.5348407664333426e-07,
    "reg_lambda": 3.3323645788192616e-08
  },
  "best_iteration": 116,
  "metric": "roc_auc",
  "metric_value": 0.9999489324103895,
  "km3tpi_version": null,
  "xgboost_version": "3.2.0",
  "sklearn_version": "1.8.0",
  "created_at": "2026-05-07T19:25:15+00:00",
  "training_manifest": "
/mnt/SteamML/Code/km3tpi/notebooks/jupyter_tests/ml/nu_vs_mu/manifest.js
on
",
  "extra": {}
}
```


Model Metadata