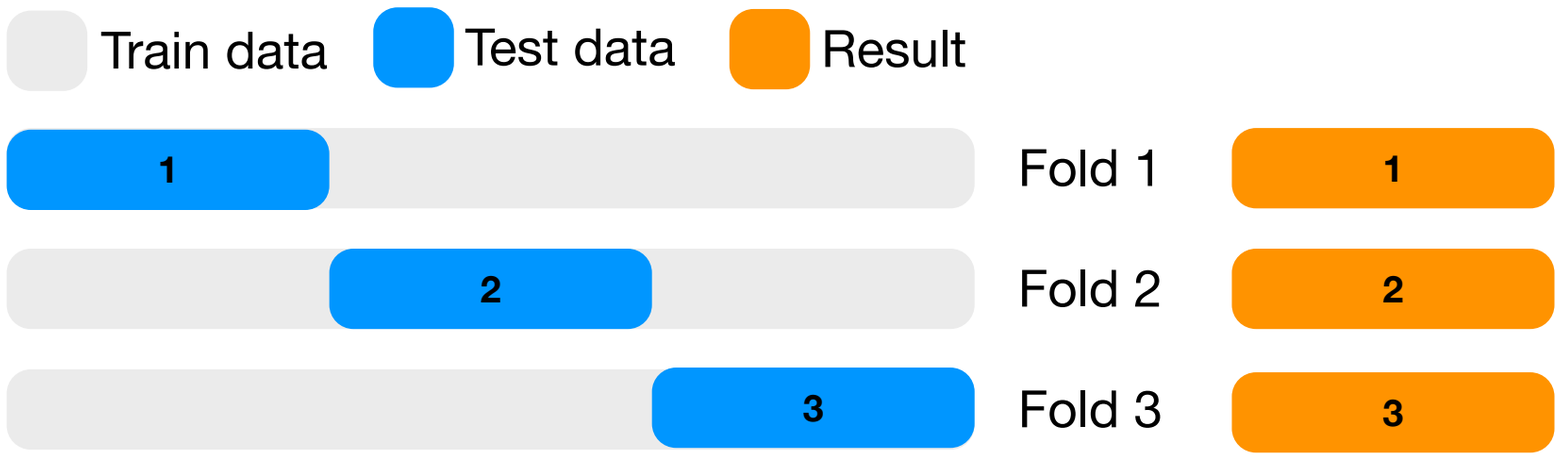
TMVA Parallelisation

Kim Albertsson

2017-09-21

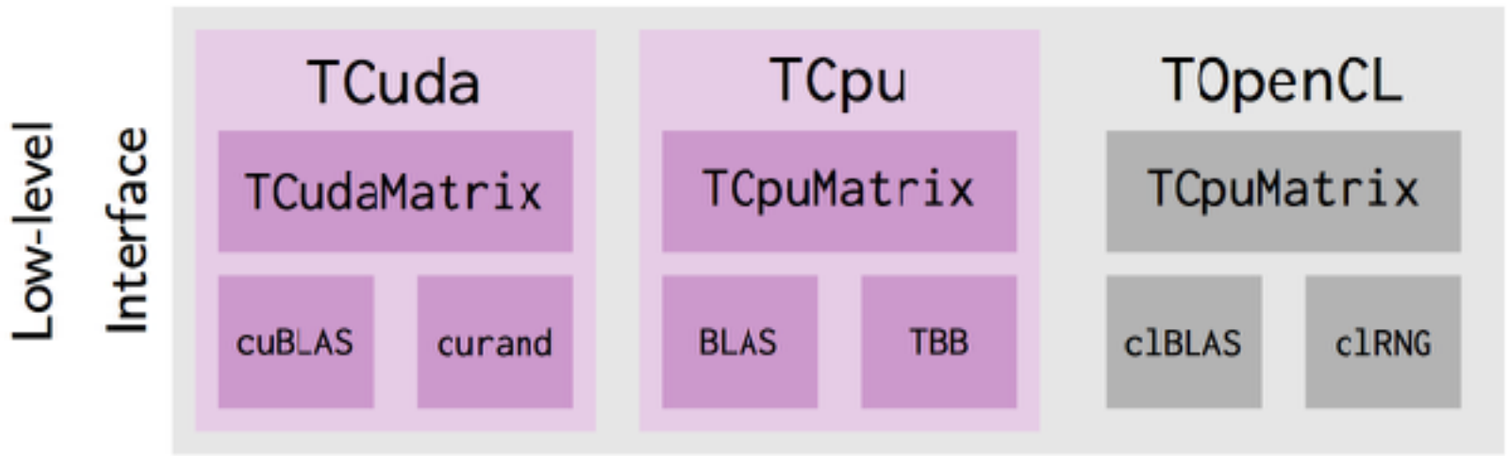
Part 1: ML Models

Cross validation



- CrossValidation, HyperParameterOptimisation, VariableImportance
- Uses TProcessExecutor to run independent folds
- Because of boosting, events need to be copied

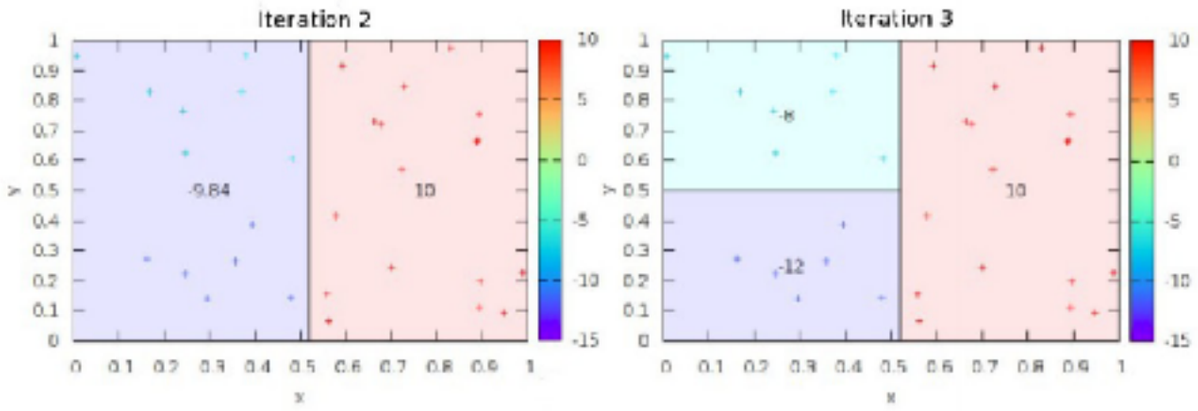
DNN



- CPU Matrix operations
  - BLAS: multiplication, transpose, sum
  - TThreadExec: hadamard, activation functions, regularisation

- TThreadExec::Map
- Each matrix holds ref to thread pool

GBDT



- **Set targets** for new tree
- **Build tree**
  - Search each feature for optimal split
  - Split region
  - Fit constants in children
  - Repeat for daughters until *stopping criteria*
- **Set optimal fits in terminal nodes**
  - Fit constant in each region to minimise loss
  - Update predictions
- **Repeat** until forest

thread-safe vectors?

Conclusion

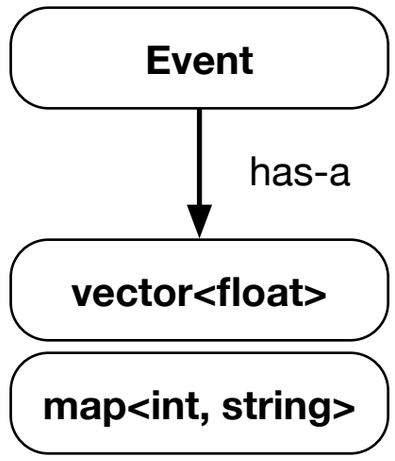
- Thread parallelisim used in DNN and soon BDT
- No vectorisation
- Process parallelism for cross validation etc.

- Future improvements
  - Make event collection read-only (move generalised boosting)
  - Vectorisation e.g. DNN
  - Thread parallelism e.g. BDT eval
  - IMT cuda/opencl???

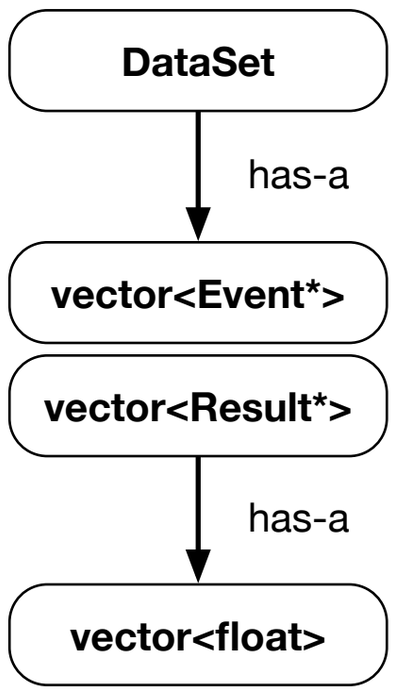
Part 2: Data handling

Events

- TMVA::Event
  - std::vector<float>
  - mapping of variable index to name



- TMVA::DataSet
  - std::vector<Event*>
  - std::vector<float> // results



Events - Declaration

TTreeFormula

```
DataLoader d {"name"};
d.AddVariable("x", ‘F');
d.AddVariable("y := var1+var2", 'F');
d.AddTree(tree1, "Signal"    , weight, "pt<10");
d.AddTree(tree2, "Background", weight, "pt<10");
```

Cut

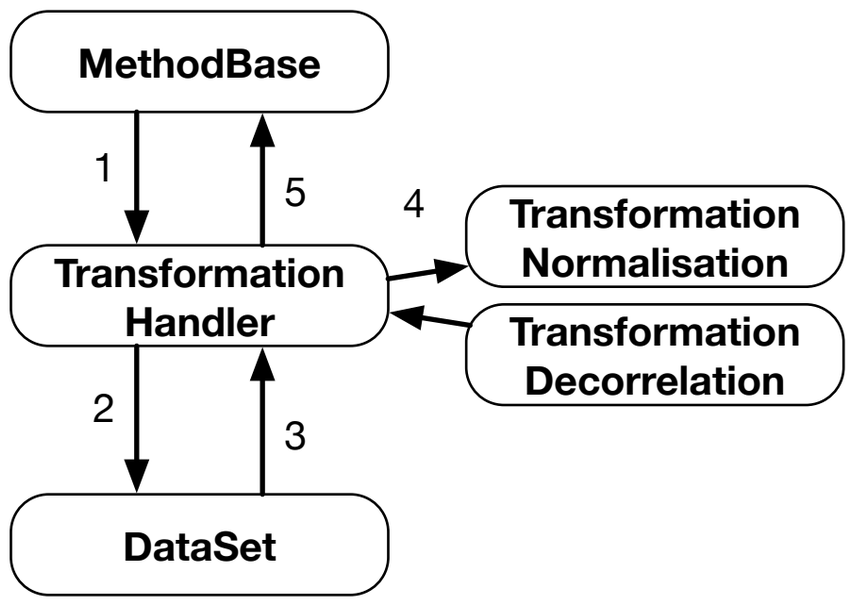
Class name

- Declare used input variables
- Add input sources (TTree, txt/csv/dat) to handler

Events - Generation

- Magic happens in TMVA::DataSetFactory::BuildEventVector
  - Loops through all input trees/chains
  - Loops through all rows
  - Build Event by eval all TTreeFormulas
  - Inserts Event into std::vector
- Vector stored in DataSet (DataSet also stores Results)

Events - Utilisation



Transformations (e.g. normalisation, decorrelation…)

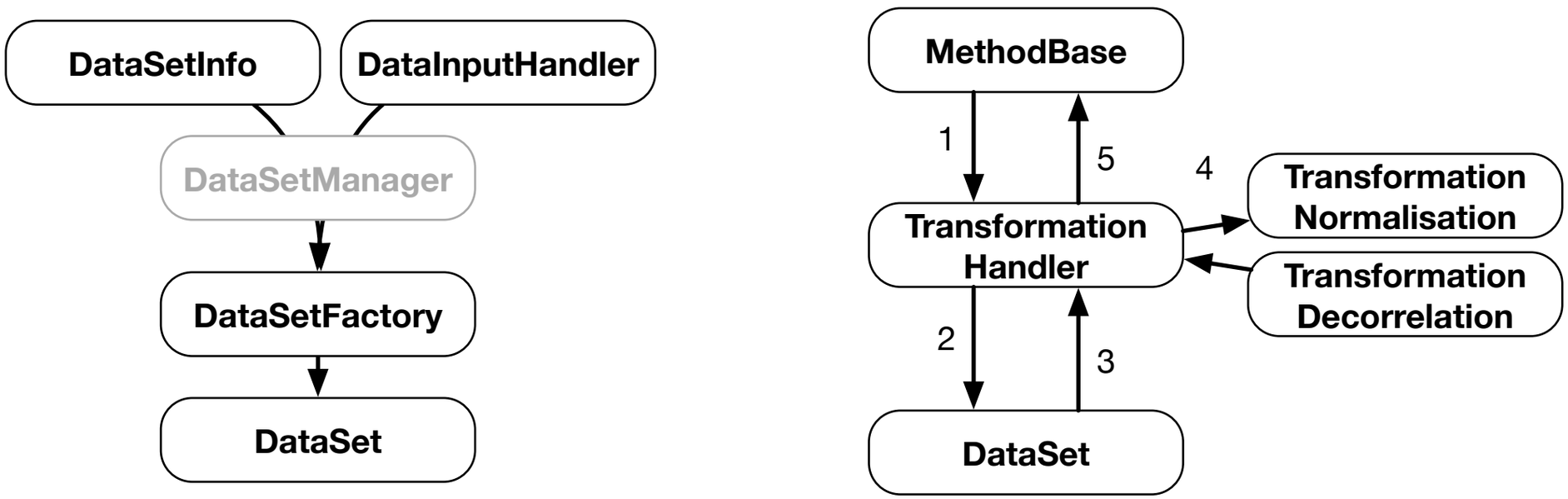
```
Event*                  MethodBase::GetEvent();
std::vector<Event*> MethodBase::GetEventCollection();
```

Events - Storage

For TMVA Gui

We write std::vector<Event*> back to TTree

Conclusion



- Replacing DataInputHandler and TTreeFormulas
  -> Very nice

- Replace event collection completely?
  -> Super nice!