

In-Database Machine Learning

Using Gradient Descent and Tensor Algebra

Maximilian Schüle, Frédéric Simonis, Thomas Heyenbrock, Alfons Kemper, Stephan Günemann, Thomas Neumann
 {schuele, simonis, kemper, guennemann, neumann}@in.tum.de

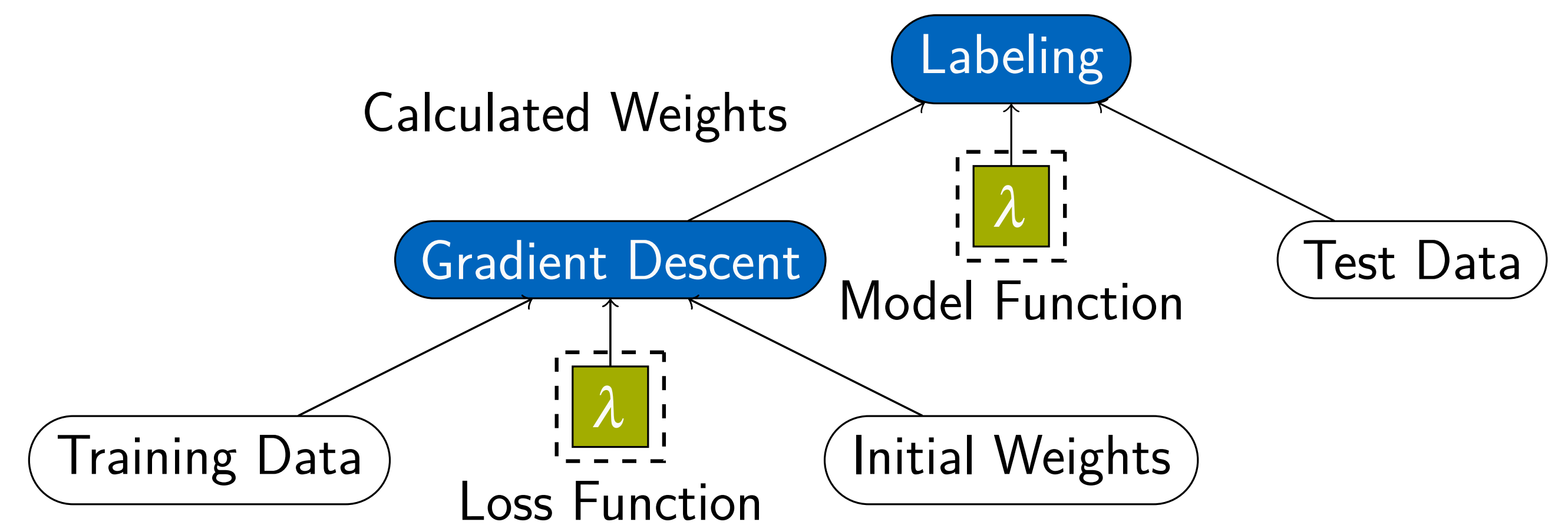
HyPer + Tensors + Gradient Descent

Machine Learning: Data in tensors and a loss function

- Operator for gradient descent:
 - Gradient needed for gradient descent: automatic differentiation necessary for arbitrary loss functions
 - Integration in relational algebra
 - Representation of a loss function

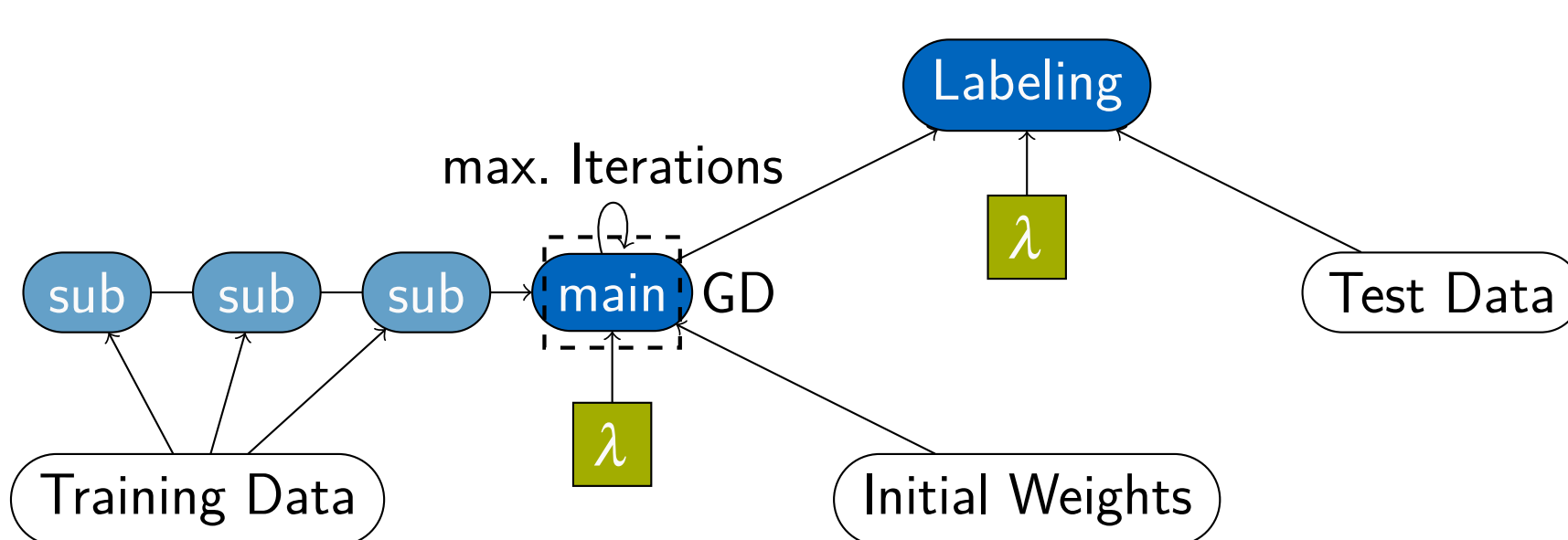
$$\lambda(R,S)(R.a * S.x + R.b - S.y)^2$$

- Tensors: datatype with algebra
- + optimisation problems solvable in the core of database systems



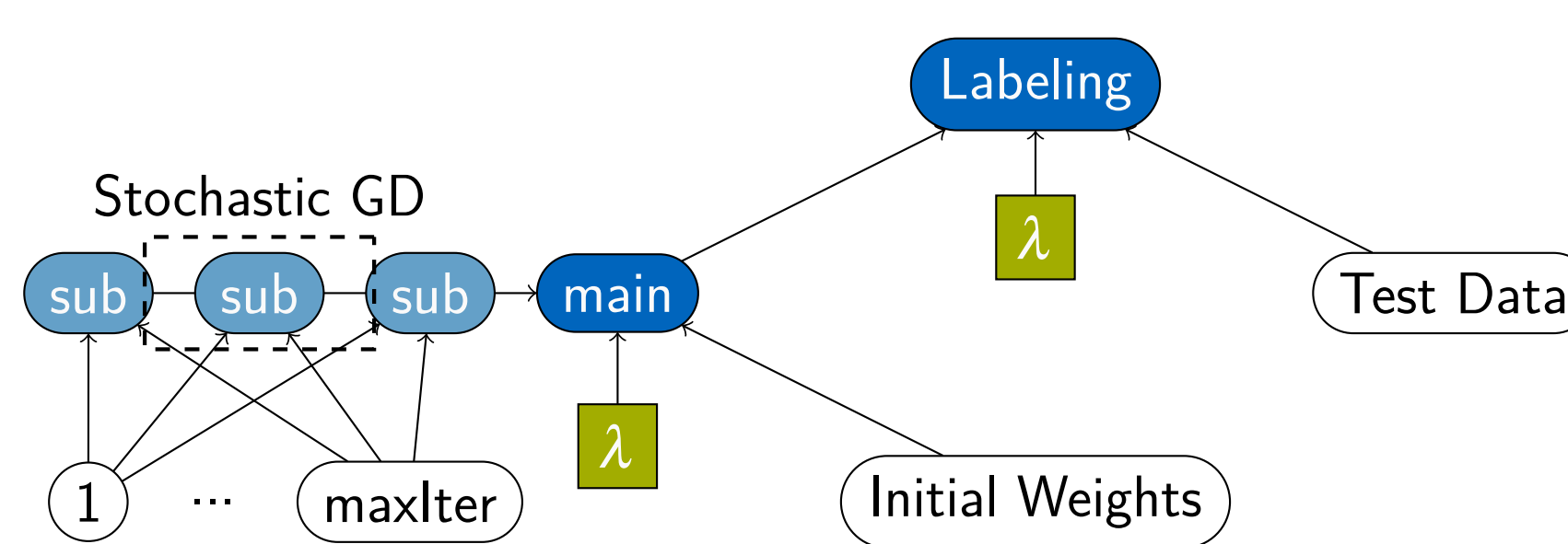
```
create table trainingdata (x float, y float);
create table weights(a float, b float);
insert into trainingdata ... insert into weights ...
select * from gradientdescent(
  -- the loss function is specified as lambda-expression
  lambda(data,weights) (weights.a*d.x+weights.b-d.y)^2,
  (select x,y from trainingdata d),
  (select a,b from weights),
  0.05, 100); -- learning rate and max. number of iterations
```

Materializing



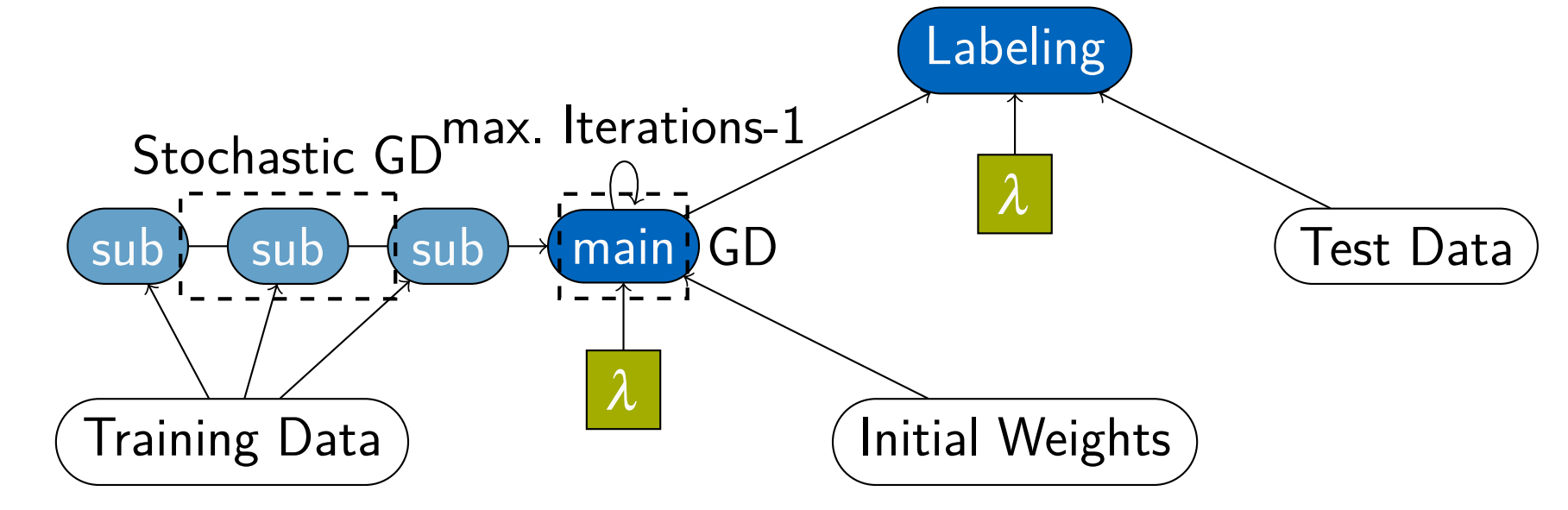
- + allows any optimization method
- Tuples need to be materialized

Pipelined



- + No materialization required
- Iterations must be precompiled

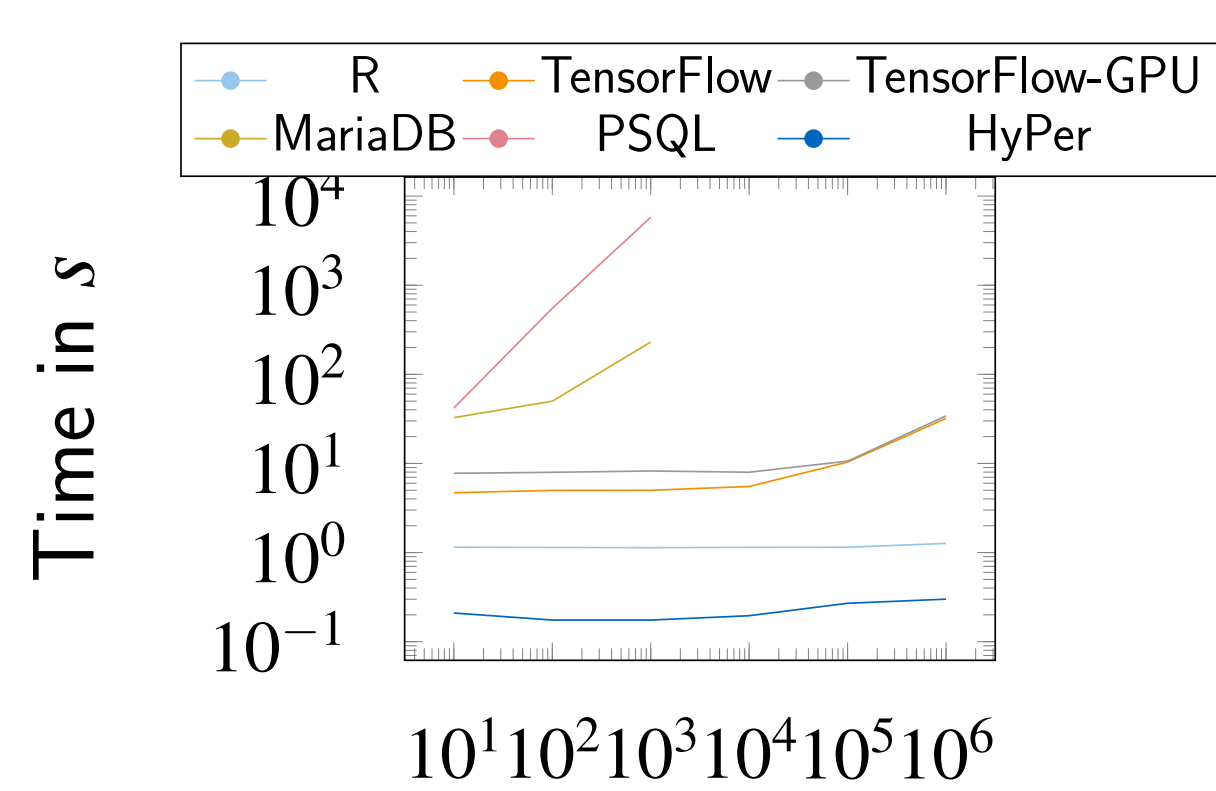
Combined



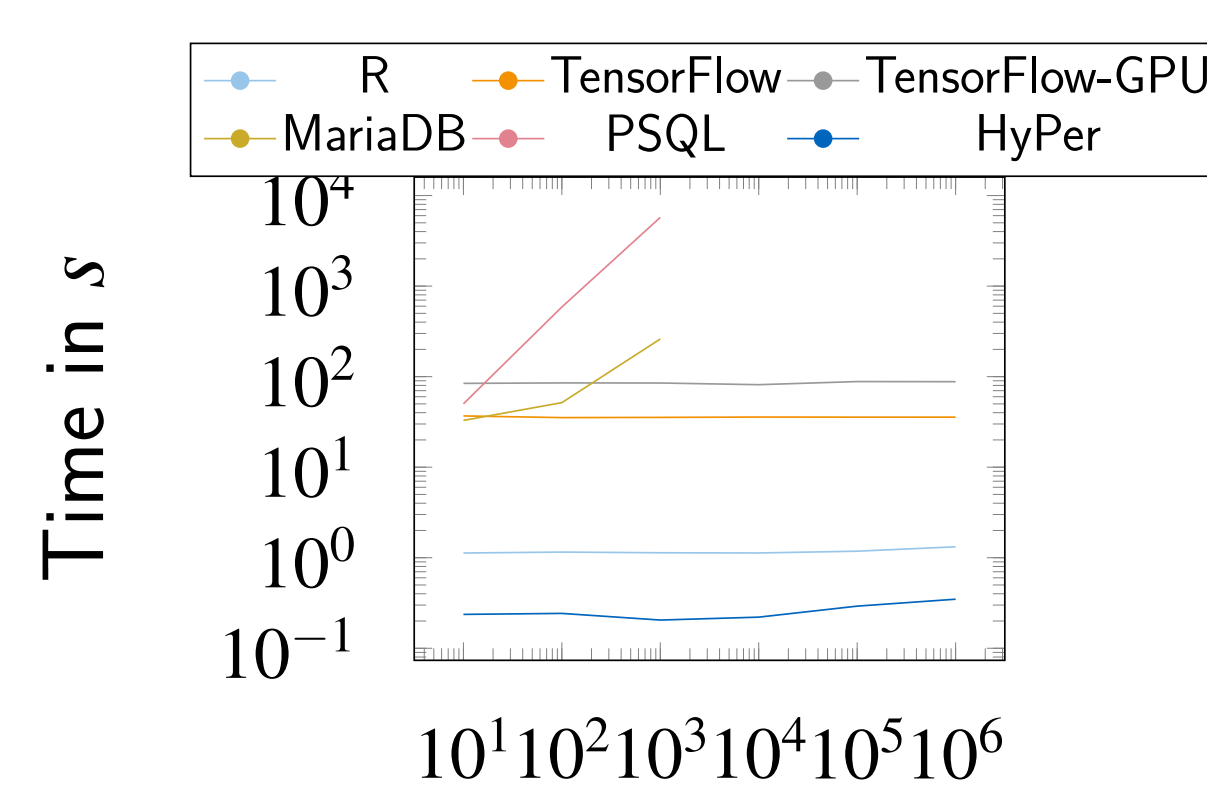
- + Precomputes weights in pipelines
- Little performance gains

Evaluation

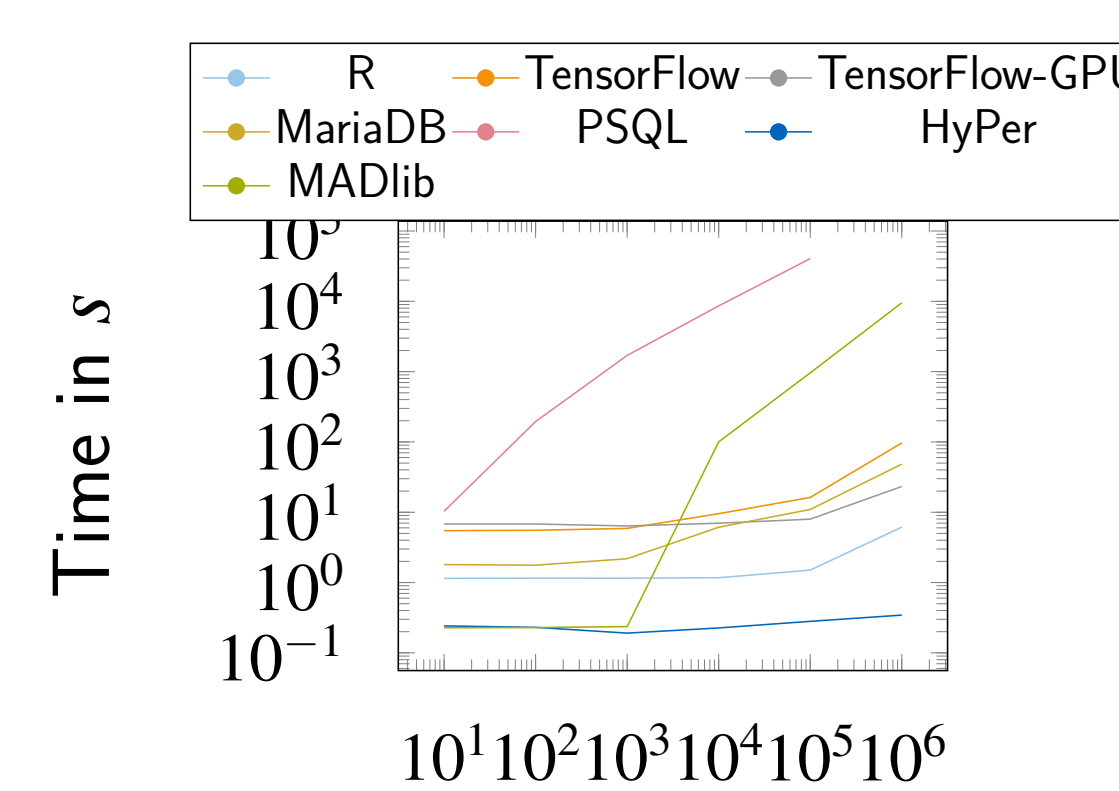
Linear Regression



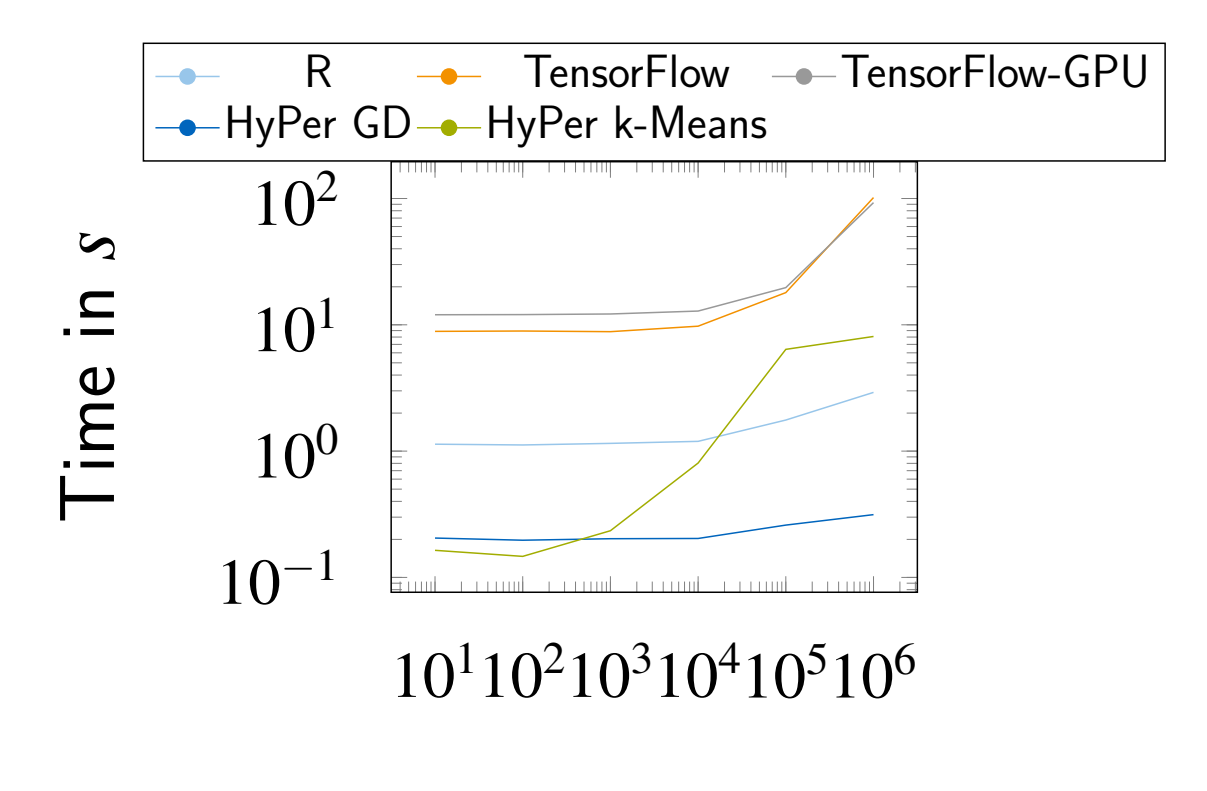
Multiple Linear Regression



Logistic Regression



k-Means

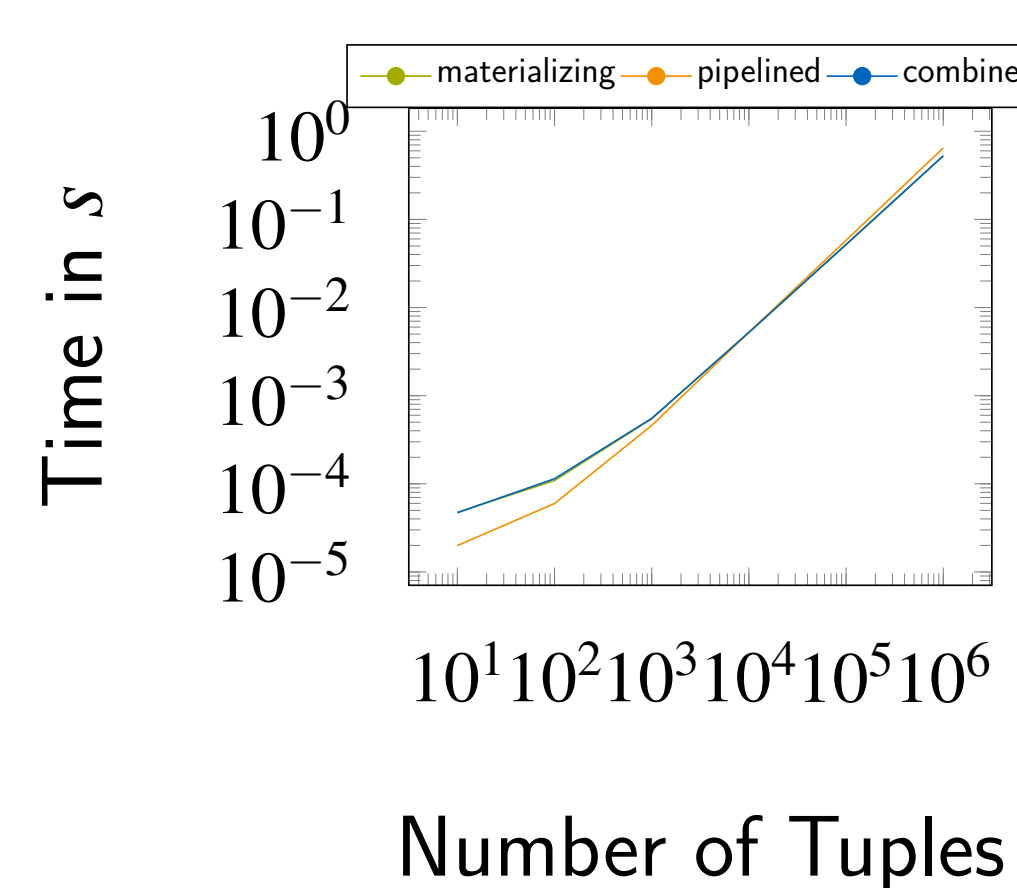
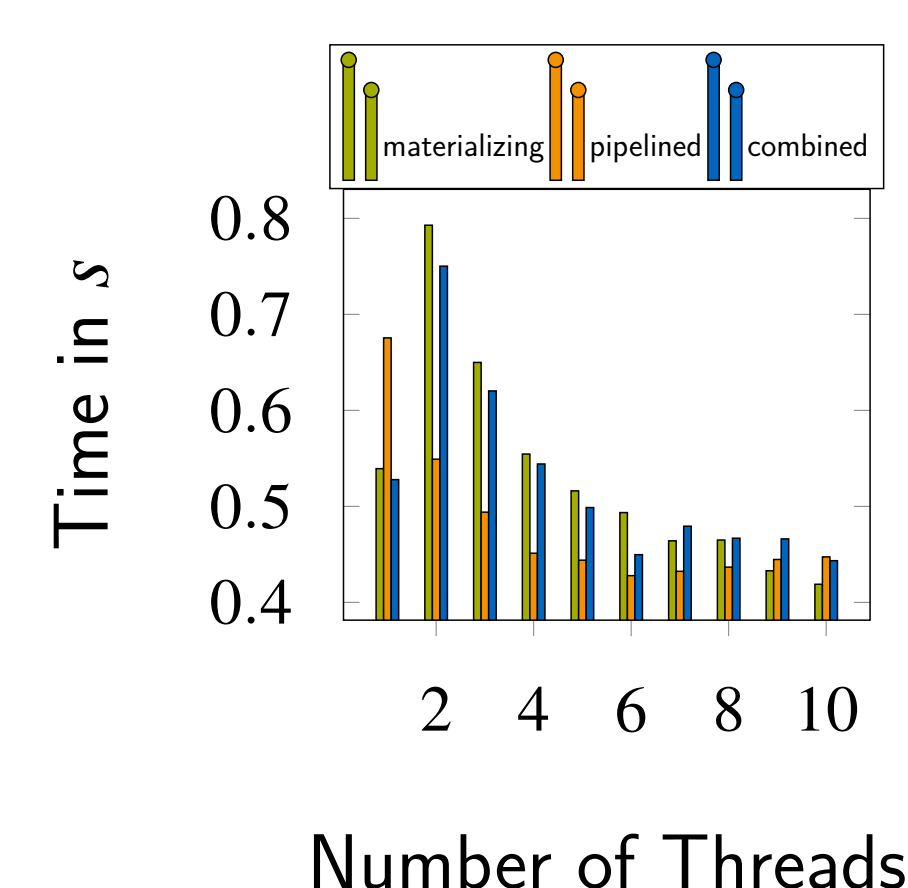


Number of Tuples

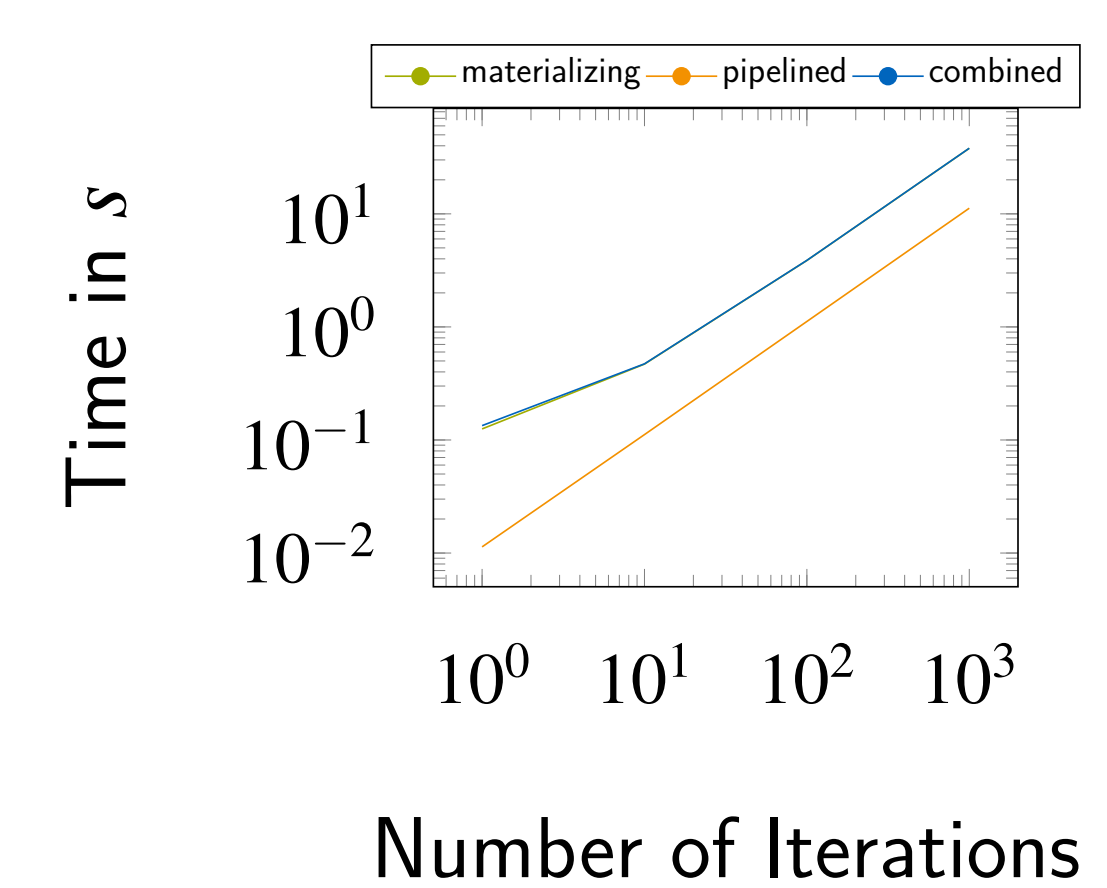
Number of Tuples

Number of Tuples

Number of Tuples



Number of Tuples



Number of Iterations