

Generalized Linear Modelling in H2O4GPU

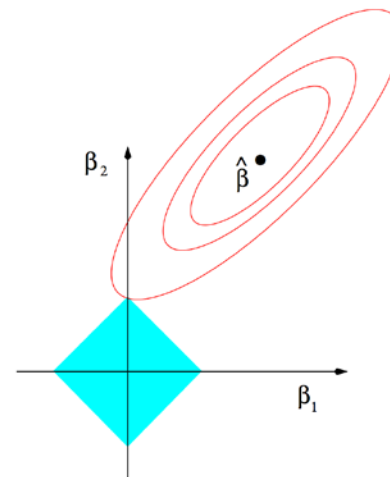
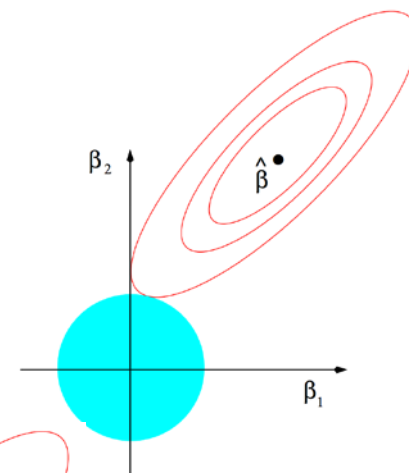
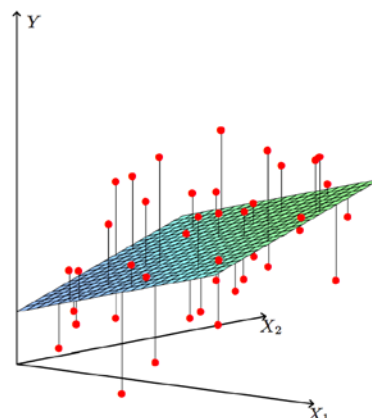
/ Framework utilizes Proximal Graph Solver (POGS) from Stephen Boyd & Chris Fougner ([Parameter Selection and Pre-Conditioning for a Graph Form Solver -- C. Fougner and S. Boyd](#))

- A solver for convex optimization problems in *graph form* using [Alternating Direction Method of Multipliers](#) (ADMM)

/ Solvers include Lasso, Ridge Regression, Logistic Regression, and Elastic Net Regularization

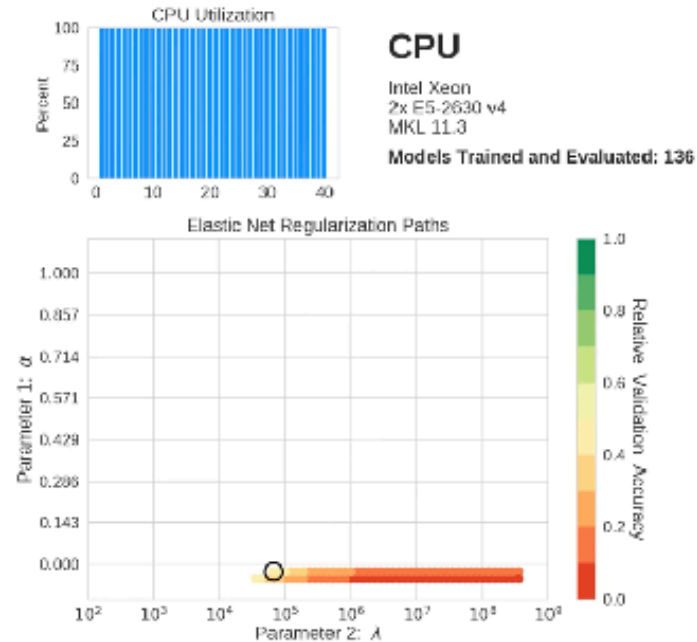
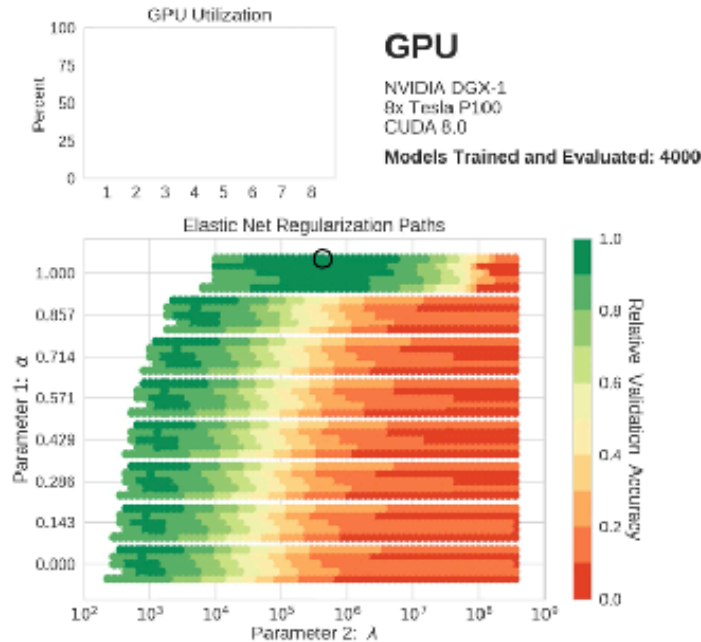
/ Improvements to original implementation of POGS:

- Full alpha search
- Cross Validation
- Early Stopping (RMSE for regression problems and Logloss for classification)
- Various bug fixes from original implementation
- Added Scikit learn “like” API
- Supports multiple GPUs





H2O.ai Machine Learning – Generalized Linear Modeling

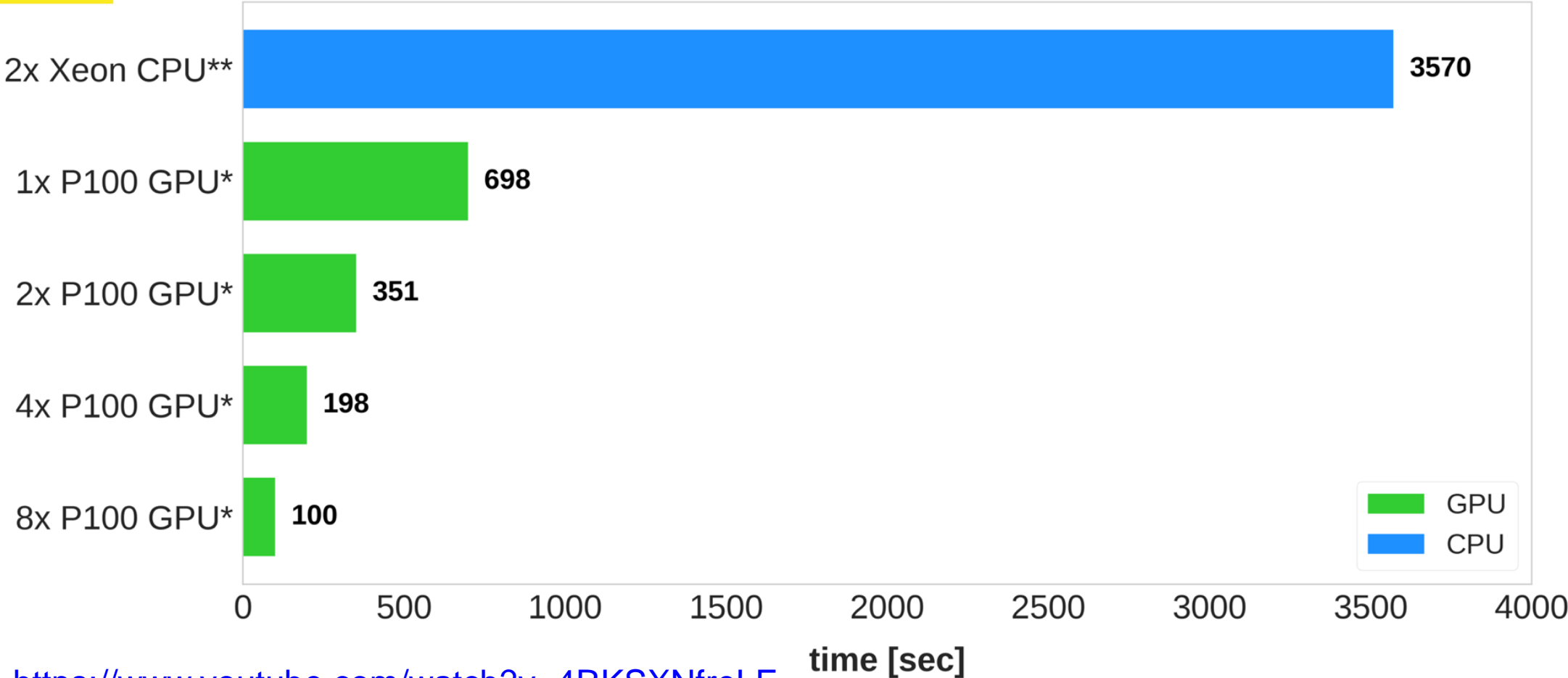


U.S. Census dataset (predict Income): 45k rows, 10k cols
Parameters: 5-fold cross-validation, $\alpha = \{\frac{i}{7}, i = 0 \dots 7\}$, full λ -search

<https://www.youtube.com/watch?v=LrC3mBNG7WU>

H2O.ai Machine Learning – Generalized Linear Modeling

Time to Train and Evaluate 4000 Models



<https://www.youtube.com/watch?v=4RK SXNfreLE>

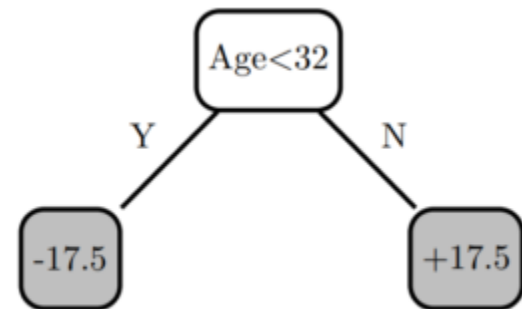
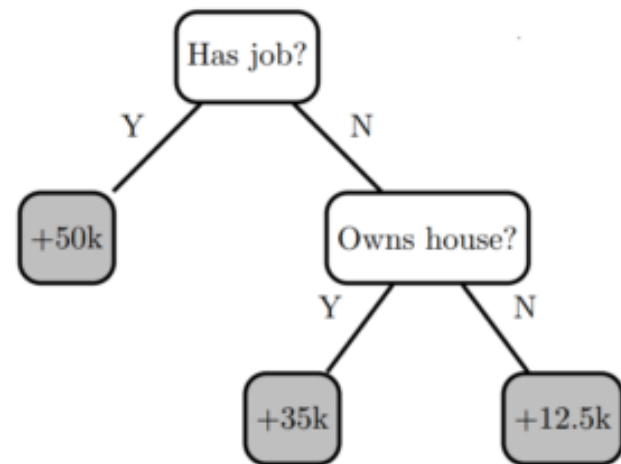
<http://github.com/h2oai/perf/>

*NVIDIA DGX-1, **Dual Intel Xeon E5-2630 v4
U.S. Census dataset (predict Income): 45k rows, 10k cols
Elastic Net Model Parameters: 5-fold cross-validation, $\alpha = \{\frac{i}{7}, i = 0 \dots 7\}$, full λ -search

Gradient Boosting Machines in H2O4GPU

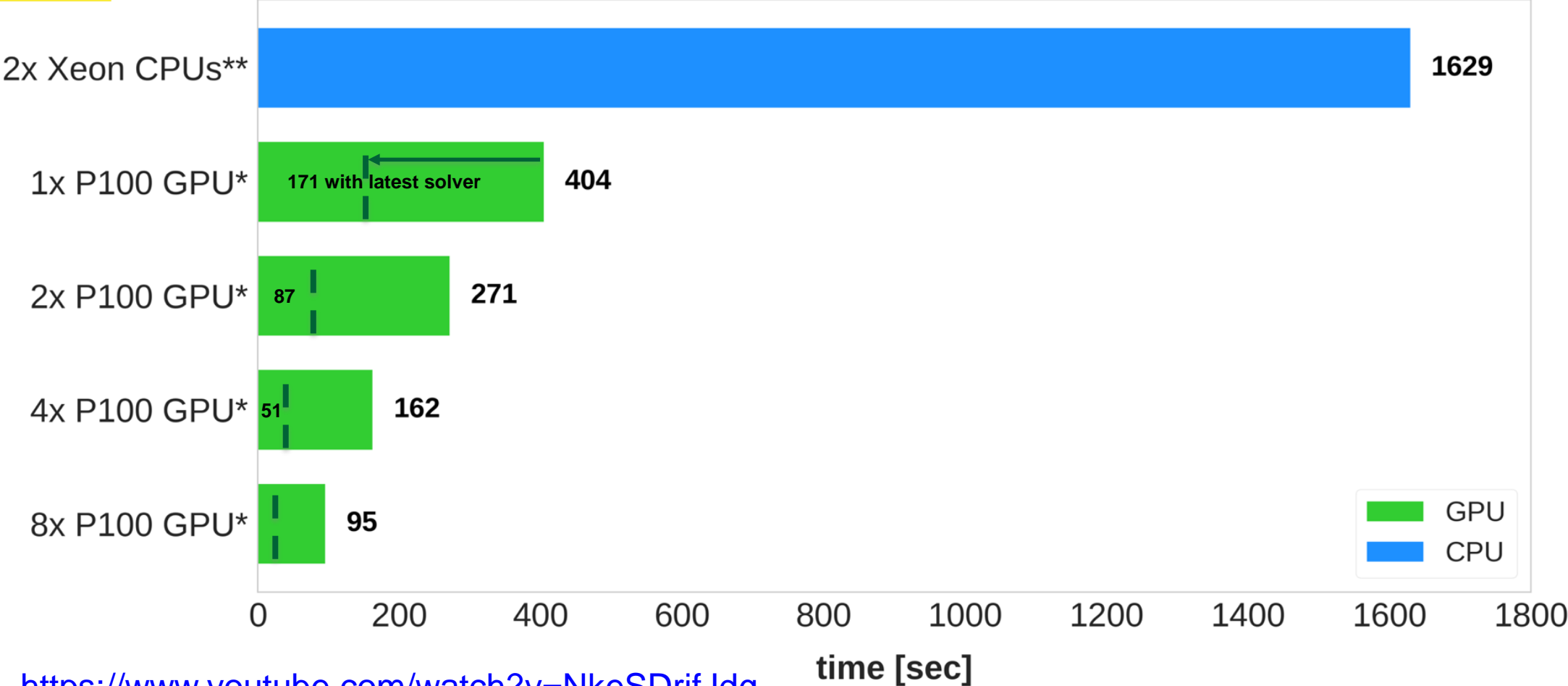
- / Based upon XGBoost
- / Raw floating point data -> Binned into Quantiles
- / Quantiles are stored as compressed instead of floats
- / Compressed Quantiles are efficiently transferred to GPU
- / Sparsity is handled directly with highly GPU efficiency
- / Multi-GPU by sharding rows using NVIDIA NCCL AllReduce

https://github.com/h2oai/h2o4gpu/blob/master/examples/py/xgboost_simple_demo.ipynb



H2O.ai Machine Learning – Gradient Boosting Machine

Time to Train 16 H2O XGBoost Models



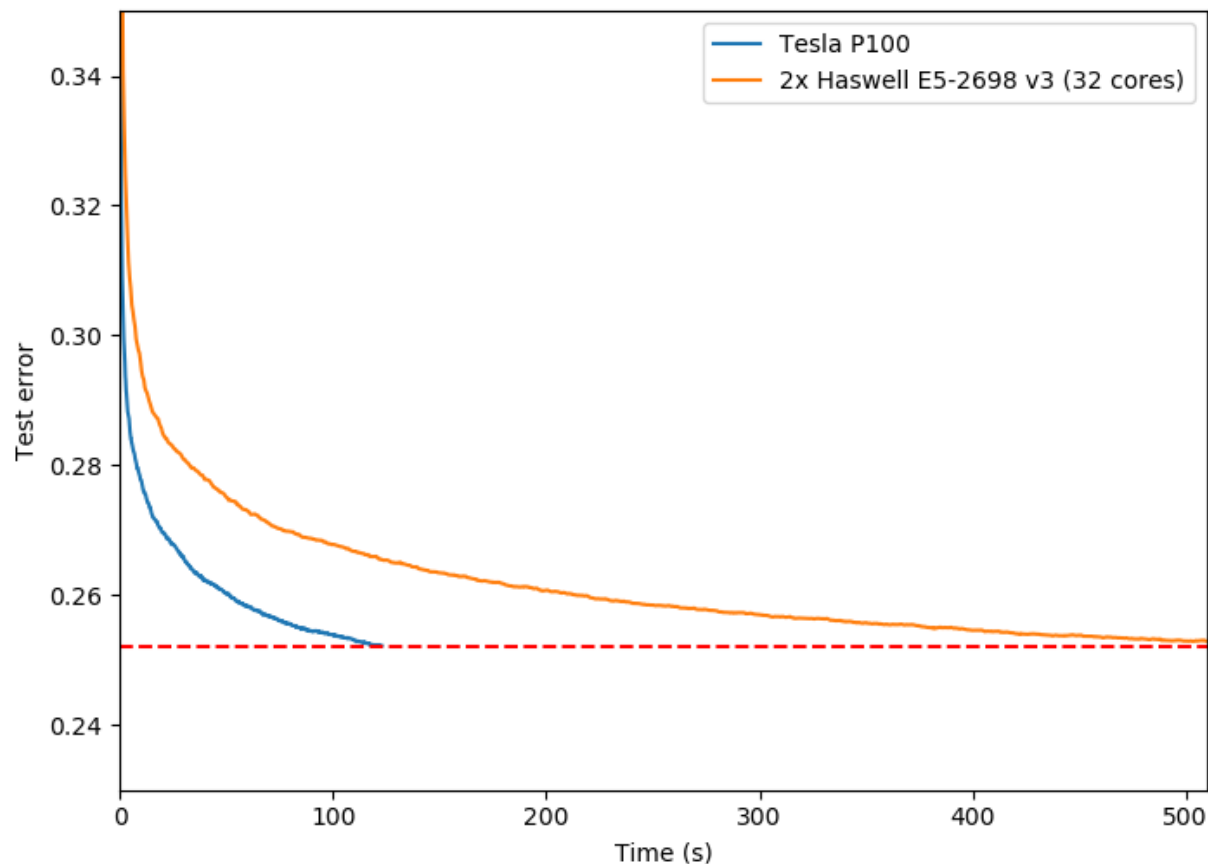
<https://www.youtube.com/watch?v=NkeSDrifJdg>

<http://github.com/h2oai/perf/>

Higgs dataset (binary classification): 1M rows, 29 cols; max_depth: {6,8,10,12}, sample_rate: {0.7,0.8,0.9,1.0}

*NVIDIA DGX-1, **Dual Intel Xeon E5-2630 v4

CPU vs. GPU on Higgs (Classification)



<https://devblogs.nvidia.com/parallelforall/gradient-boosting-decision-trees-xgboost-cuda/>

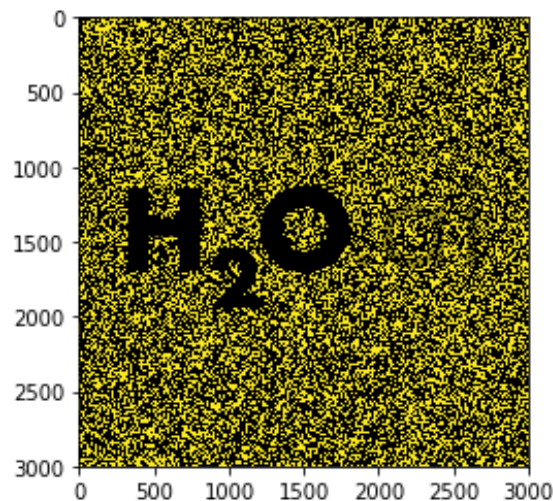
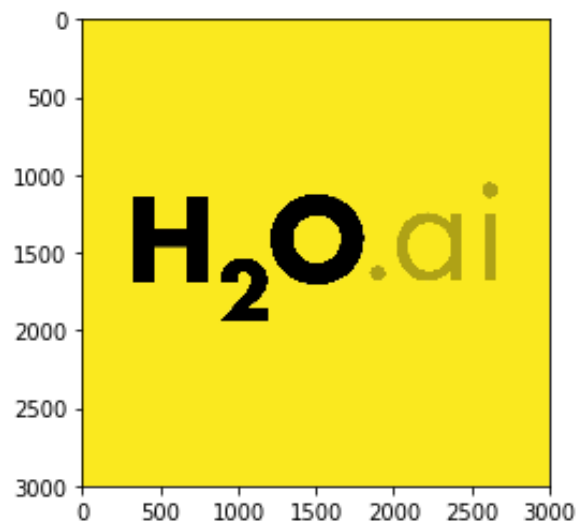
K-Means on H2O4GPU

/ Based upon NVIDIA prototype of K-Means algorithm in CUDA

/ Improvements to original implementation:

- Significantly faster than scikit-learn implementation (50x)
- Significantly faster than other GPU implementations (<https://github.com/src-d/kmcuda>) (5x-10x)
- Various bug fixes
- Supports multiple GPUs

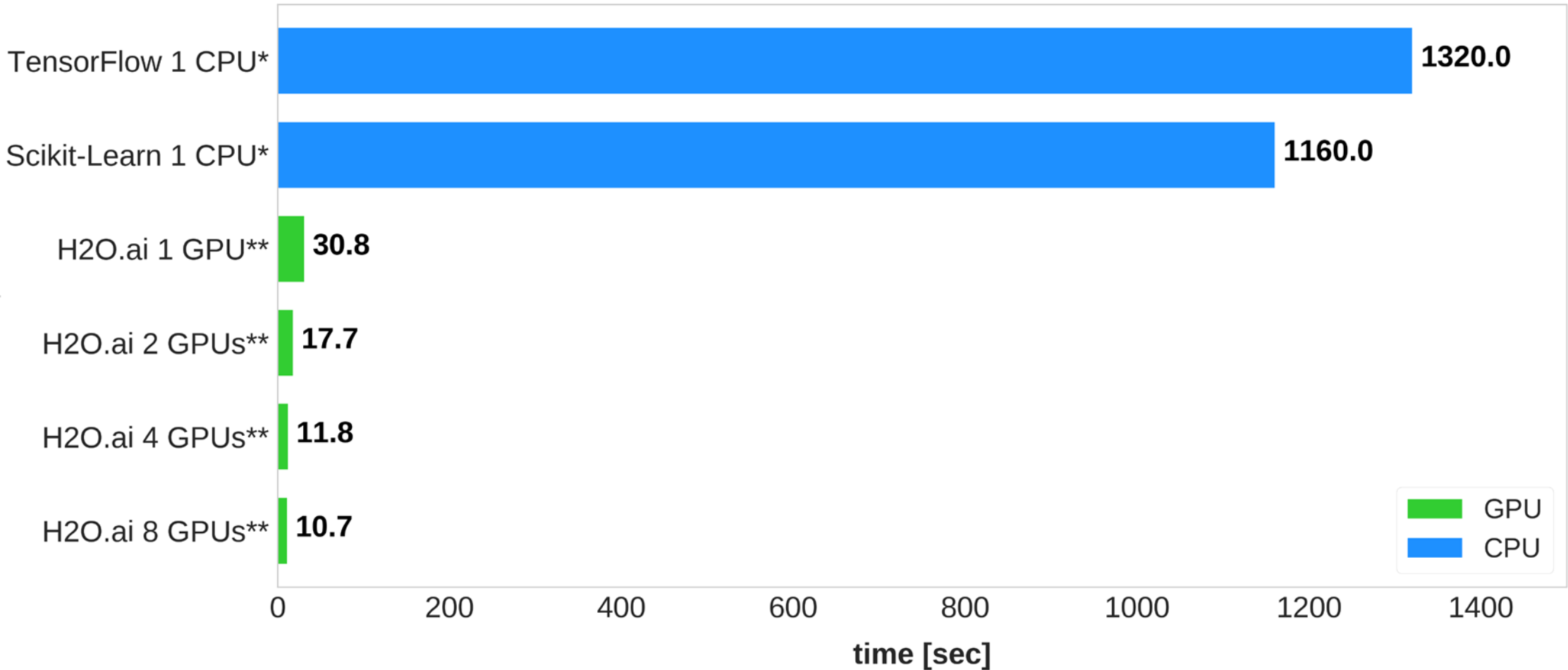
K-Means



https://github.com/h2oai/h2o4gpu/blob/master/examples/py/demos/H2O4GPU_KMeans_Images.ipynb

H2O.ai Machine Learning – k-Means Clustering

Time to run 1000 Lloyds iterations for k=1000 clusters



Kaggle Homesite Home Insurance Claims Predictions Dataset (261k rows, 298 cols)
k-Means Clustering (Lloyds), random initialization, 1000 centroids, 1000 iterations
Hardware: *Intel i7 5820k (6-core), **NVIDIA Tesla P100 (DGX-1)

<http://github.com/h2oai/perf/>