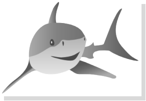
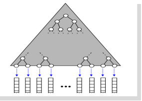
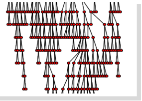
Faculty of Science

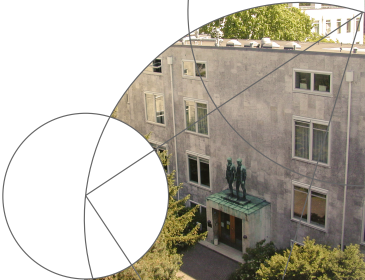
Three DIKU Open-Source Machine Learning Tools

Fabian Gieseke, Oswin Krause, Christian Igel
Department of Computer Science



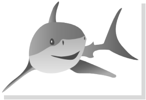




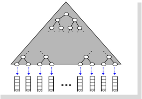


igel@diku.dk

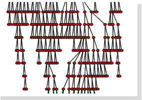
In this episode . . .



Highly efficient C++ machine learning library



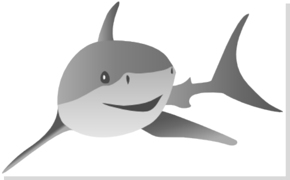
Ultrafast exact nearest neighbor computation using GPUs – processing millions of data points in seconds



Building random forest with 10^8 data points in less than ten minutes on desktop computers

Shark

- Object-oriented software library for machine learning and optimization
- Methods for supervised and unsupervised learning and a wide range of standard methods for classification and regression.
- Toolbox for direct and gradient-based single- and multi-objective optimization
- New version 3.0: Almost complete rewrite
- New linear algebra engine Remora



Igel, Glasmachers, Heidrich-Meisner. Shark, *Journal of Machine Learning Research*, 2008

Exemplary methods

kernel methods	support vector machines kernel regression kernel selection algorithms
neural networks	feed-forward neural networks auto-encoders restricted Boltzmann machines dropout training
standard techniques	K-means clustering LASSO regression classification and regression trees random forests
optimization	L-BFGS, CG, trust-region Newton steepest descent CMA-ES MO-CMA-ES

Code snippets

```
ClassificationDataset trainingSet;
importSparseData(trainingSet, "dataset");
```

```
FFNet<RectifierNeuron,LinearNeuron> model;
CrossEntropy loss;
ErrorFunction objFunc(trainingSet, &model, &loss);
```

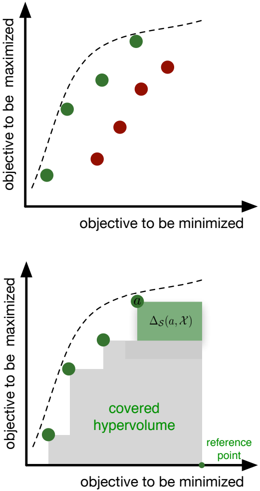
```
GradientDescent optimizer;
optimizer.init(objFunc);
for(std::size_t i = 0; i != 1000; ++i)
        optimizer.step(objFunc);
```

```
GaussianRbfKernel<> kernel(gamma);
KernelClassifier<RealVector> svm;
CSvmTrainer<RealVector> trainer(&kernel, C);
trainer.trainingSet(svm, trainingSet);
```

Developed in DABAI: UP-MO-CMA-ES

Multi-objective optimization with unbounded solution sets using UP-MO-CMA-ES:

- Derivative-free optimization algorithm
- Pareto front approximated by multivariate local search distributions
- All non-dominated points are kept
- Selection based on hypervolume
- Promising competition results
  - 1^{st} BBComp 3-objectives
  - 2^{nd} BBOB and BBComp 2-objectives

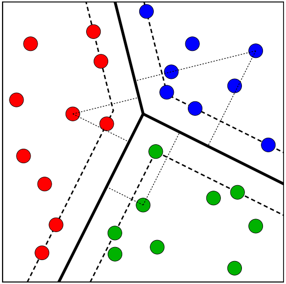


Krause, Arbonès, Igel. CMA-ES with Optimal Covariance Update and Storage Complexity. NIPS, 2016
▷ Krause et al. Unbounded population MO-CMA-ES for the bi-objective BBOB test suite. GECCO BBOB, ACM, 2016
▷ Krause, Glasmachers, Igel. Multi-objective Optimization with Unbounded Solution Sets. NIPS BayesOpt , 2016

True multi-class SVMs

- There is no canonical way to extend SVMs to multiple classes
- One-vs-all often works, but has conceptual problems
- We developed
  - a theoretical framework
  - efficient solvers in Shark

  for true "all-in-one" multi-class SVMs



Dogan, Glasmachers, Igel. A Unified View on Multi-class Support Vector Classification. *Journal of Machine Learning Research* 17, 2016

All-in-one multi-class SVMs

There are many different all-in-one SVMs in Shark, e.g.:

- The equivalent multi-class SVMs by Weston & Watkins (WW), Vapnik, and Bredensteiner & Bennett

  Bredensteiner, Bennett. Multicategory classification by support vector machines. Computational Optimization and Applications, 12(1):5379, 1999.

  Weston, Watkins. Support vector machines for multi-class pattern recognition. ESANN, pp. 219224, 1999.

  Vapnik. Statistical Learning Theory. John Wiley and Sons, 1998.

- Crammer and Singer's popular variant (CS)

  Crammer, Singer. On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research, 2:265292, 2002.

- Lee, Lin and Wahba's consistent multi-class SVM (LLW)

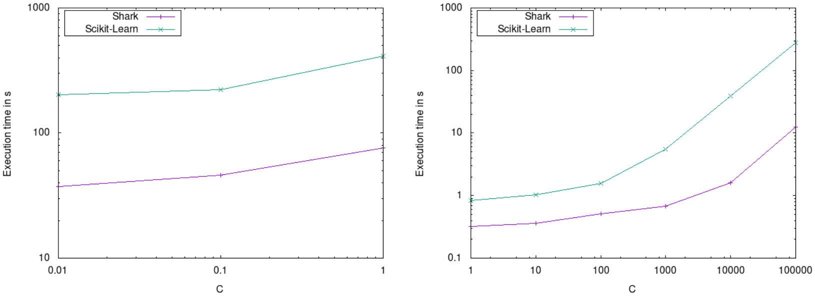
  Lee, Lin, Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. Journal of the American Statistical Association, 99(465):67–82, 2004.

- Multi-class maximum margin regression (MMR)

  Szedmak, Shawe-Taylor, Parado-Hernandez. Learning via linear operators: Maximum margin regression. Technical report, PASCAL, 2006.

- ...

How fast is the Shark? SVMs

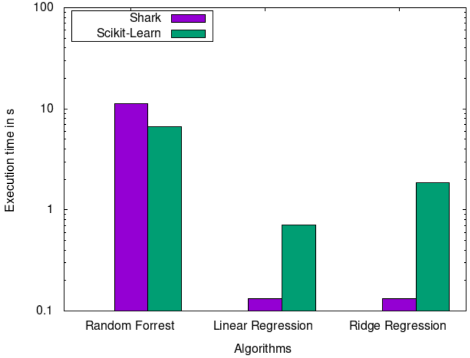


Left: Gaussian kernel SVM, `cod-rna` data, $\ell = 59535$, $d = 8$, $\varepsilon = 0.001$, $C = 0.01, \ldots, 1$, $\gamma = 1$, kernel cache 256MB

Right: linear SVM, `rcv1.binary` data, $\ell = 20,242$, $d = 47236$, $\varepsilon = 0.001$, $C = 1, \ldots, 100000$

Glasmachers, Igel. Maximum-Gain Working Set Selection for SVMs. *Journal of Machine Learning Research*, 2006
Glasmachers, Dogan. Accelerated Coordinate Descent with Adaptive Coordinate Frequencies. *JMLR W&CP 29*, 2013

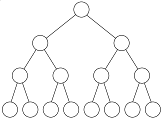
How fast is the Shark? Other examples ...



Random forest on `cod-rna` data, linear/ridge regression on `BlogFeedback` ($\ell = 60,021, d = 281$)

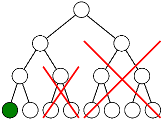
Nearest neighbor queries

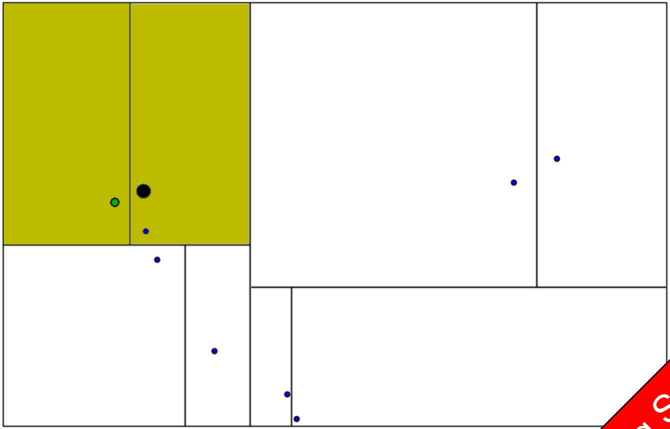
- Determining nearest neighbors is a fundamental task in machine learning, e.g., for regression, classification, outlier detection, density estimation,
- Applying (exact) nearest neighbor queries to huge data sets is a challenge

 **+** 

Idea: Combine k-d trees and massively-parallel programming

K-d trees



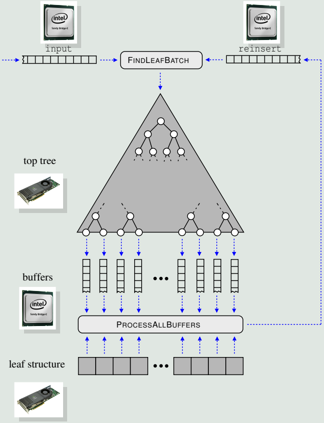


Big Speed-Up!
But can still be too slow . . .

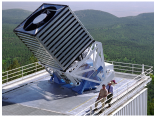
Buffer K-d trees

Buffer k-d Trees (Sketch)

1. *Top tree:* First levels of a standard k-d tree, pointer-less memory layout
2. *Leaf structure:* Training patterns, sorted *in-place* during top tree construction
3. *Buffers:* One buffer for each leaf of the top tree storing query indices
4. *Queues* `input` & `reinsert`*:* FIFO queues



Buffer K-d trees speed



Evaluation on real-world astronomy task:

	$d=5$	$d=10$	$d=15$	$d=12$	$d=27$
CPU	57	527	4616	16394	–
GPU	12	36	210	478	1717
	×5	×15	×22	×34	–

CPU: k-d tree
GPU: buffer k-d tree

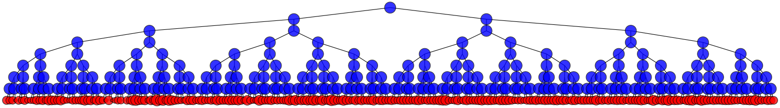
Time in s, Intel i7@3.40GHz (4 cores), GeForce GTX 770 (1536 cores, 4GB RAM), $2 \cdot 10^6$ training and 10^7 test patterns

Gieseke, Heinermann, Oancea, Igel. Buffer k-d Trees: Processing Massive Nearest Neighbor Queries on GPUs. *International Conference on Machine Learning* (ICML), 2014

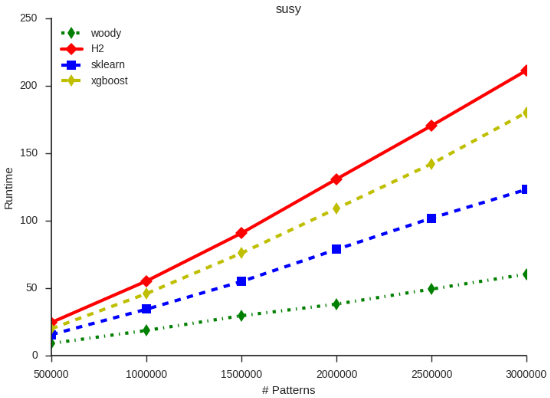
Gieseke, Oancea, Igel. bufferkdtree: A Python Library for Massive Nearest Neighbor Queries on Multi-Many-Core Devices. *Knowledge-Based Systems*, 2017

Woody: Large-scale random forests

- Random forests are among the most powerful machine learning techniques in practice – how do we apply them to millions of data points without expensive compute resources?
- We would like to grow full trees to tackle class imbalance.
- Solution: Built top-tree(s) that lead to more balanced tree(s)!



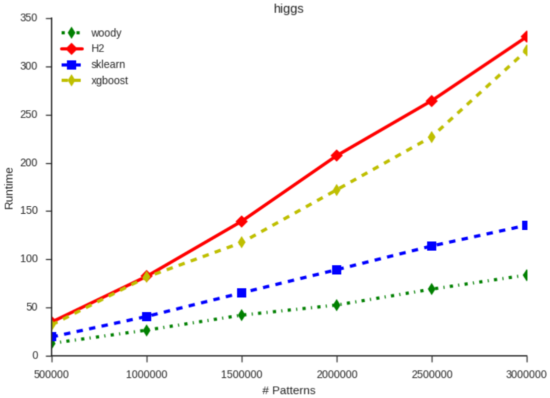
Woody: First results I



$d = 18$, full trees, checking all features per split, 4 trees, Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz (4 cores), 32GB RAM

Similar classification accuracies; H2/XGBoost optimized for shallow trees (max_depth=1000/max_depth=100)

Woody: First results II



$d = 28$, full trees, checking all features per split, 4 trees, Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz (4 cores), 32GB RAM

Similar classification accuracies; H2/XGBoost optimized for shallow trees (max_depth=1000/max_depth=100)

Big trees on small machines

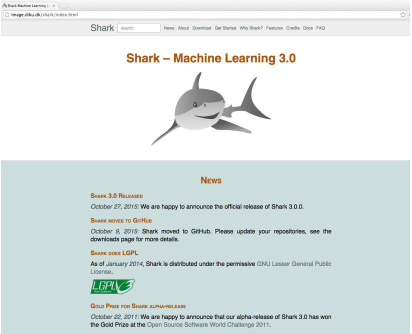
Training Output

```
13:51:20,588 - Number of training patterns:    113212922
13:51:20,588 - Dimensionality of the data:     11
13:51:20,588 - Fitting forest ...
13:51:20,596 - Setting n_top to 532007.
13:51:20,596 - Setting n_patterns_leaf to 106401.
13:51:20,596 - (I) Retrieving random subsets for top trees ...
13:51:20,596 - Retrieving random subsets for all estimators...
13:52:10,882 - Storing subsets for all estimators ...
13:52:10,977 - (II) Fitting all top trees ...
13:52:10,977 - Fitting top tree for estimator 0 ...
13:52:13,629 - Saving top tree for estimator 0 ...
13:52:13,629 - (III) Distributing all patterns to leaves ...
...
13:59:52,777 - [2054/2054] Fitting bottom subforest 4018 for 56551 patterns ...
13:59:52,908 - -------------------------------------------------------------------
13:59:52,908 - Fitting Statistics
13:59:52,908 - -------------------------------------------------------------------
13:59:52,908 - (I)   Retrieving subsets:                      53.034 (s)
13:59:52,909 - (II)  Top tree constructions:                  53.034 (s)
13:59:52,909 - (III) Distributing to top tree leaves:         240.208 (s)
13:59:52,909 - (IV)  Bottom trees constructions:              219.071 (s)
13:59:52,909 -                                                512.312 (s)
```

Shark

`image.diku.dk/shark`

Igel, Glasmachers, Heidrich-Meisner: Shark, *Journal of Machine Learning Research* 9:993–996, 2008



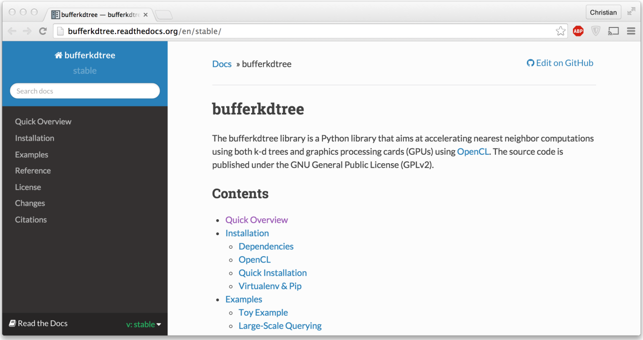
Gold Prize at Open Source Software Wold Challenge 2011

Buffer k-d trees

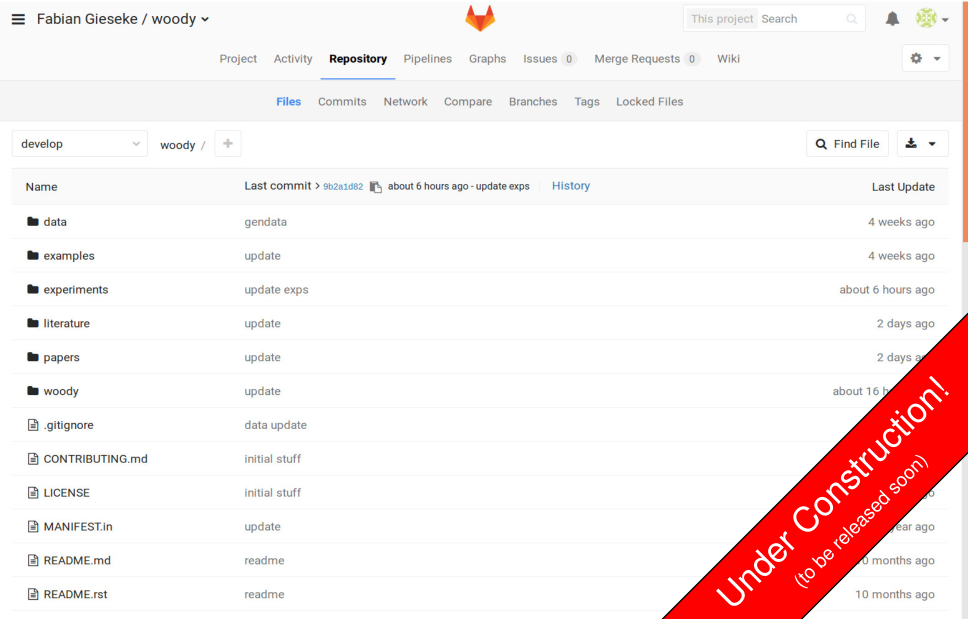
`http://bufferkdtree.readthedocs.org`

Gieseke, Heinermann, Oancea, Igel. Buffer k-d Trees: Processing Massive Nearest Neighbor Queries on GPUs. *International Conference on Machine Learning* (ICML), 2014

Gieseke, Oancea, Igel. bufferkdtree: A Python Library for Massive Nearest Neighbor Queries on Multi-Many-Core Devices. *Knowledge-Based Systems*, 2017



Woody (Python package)



Take me home

- We are further developing the Shark library – have a look if you need highly efficient machine learning on commodity hardware!

  Recent highlights:

  - Highly efficient linear algebra
  - True multi-class SVMs
  - New multi-objective algorithms

- Try buffer k-d trees for large-scale nearest neighbor queries using GPUs!
- Stay tuned: Woody, random forests (w/ large trees) with millions of training data points on your desktop!
- There is great, professional open source machine learning software maintained next door!