Help save net neutrality! A free, open internet is once again at stake—and we need your help. **Learn more**

Shark-ML / **Shark**

The Shark Machine Leaning Library. See more: http://shark-ml.org/

3,828 commits	**7** branches	**7** releases	**15** contributors	GPL-3.0

Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

christian-igel old file from MT — Latest commit `51f12b8` 15 days ago

Test	Redesigned MeanModel and RFClassifier. Should fix #206	17 days ago
doc	old file from MT	15 days ago
examples	Redesigned MeanModel and RFClassifier. Should fix #206	17 days ago
include	Redesigned MeanModel and RFClassifier. Should fix #206	17 days ago
src	rewrote Random Forest, removed CART and redid Benchmarks	25 days ago
.gitignore	removed Rng component and replaced by single header Core/Random.h	8 months ago
.travis.yml	Reintroduce the other Travis builds.	3 months ago
CMakeLists.txt	added SIMD to dthe default gemm and implemented benchmarks	10 months ago
COPYING	changed license	4 years ago
COPYING.LESSER	fixed legal thingy that prevented CPack from building	4 years ago
CTestConfig.cmake	oops	2 years ago
README.txt	Update README.txt	8 months ago
SharkConfig.cmake.in	oops	2 years ago
SharkConfigVersion.cmake.in	updated shark packaging for cmake	5 years ago
UseShark.cmake	fixed a few smaller isues with NDEBUG	2 years ago
appveyor.yml	removed MSVC12 frim appveyor and fixed cmake issues with backward sla…	7 months ago
cBlasCheck.cpp	added support for generic cblas libraries	2 years ago
cmake_uninstall.cmake.in	simple "make uninstall" added	4 years ago
update_remora.sh	added remora update script	10 months ago

README.txt

```
Shark is a fast, modular, general open-source C++ machine
learning library.

Shark is licensed under the GNU Lesser General Public License, please
see the files COPYING and COPYING.LESSER, or visit
http://www.gnu.org/licenses .

Any application of the SHARK code toward military research and use is
expressly against the wishes of the SHARK development team.


INSTALLATION / DOCUMENTATION
```

The entry point to the Shark library documentation is located at doc/index.html . For installation instructions, please click on "Getting started" on that page. Short version of installation guide: issue "ccmake ." in the main directory to select your build options, and afterwards issue "make" in the main directory -- you should be done (assuming Boost and CMake were installed). See the documentation for detailed instructions.

BUILDING THE DOCUMENTATION: To build the documentation yourself (e.g., if you need to read it locally in order to install it, i.e., because you don't have internet), see doc/README.txt

FILE STRUCTURE

README.txt	This file (residing in the root directory of the Shark library).
CMakeLists.txt	Definitions for the CMake build system.
include/	This directory and its sub-directories hold all include files of the library. Note that some functionality is implemented in lower-level Impl/ folders and inline .inl files.
lib/	The Shark library is placed in this directory. In the source code distribution this directory is initially empty, and the library is placed into the directory as the results of compilation. Binary distributions already contain the library, pre-built in release mode.
doc/	All documentation files are found in this sub-directory. In packaged versions of Shark the html documentation is pre-built; the repository provides the corresponding sources. The documentation contains technical reference documents for all classes and functions as well as a collection of introductory and advanced tutorials.
doc/index.html	Entry point to the Shark documentation.
examples/	The examples directory contains example use-cases of the most important algorithms implemented in Shark. Besides exemplifying powerful learning algorithms, these programs are intended as starting points for experimentation with the library. The executables corresponding to the C++ example programs are found in examples/bin/.
Test/	Shark comes with a large collection of unit tests, all of which reside inside the Test directory.

```
bin/                The binaries of the Shark unit tests are placed
                    here. Once the CMake build system is set up
                    (with the "ccmake" command or equivalent) the
                    whole test suite can be executed with the
                    command "make test", issued in the Shark root
                    directory.

src/                Source files of the Shark library. Note that
                    from Shark version 3 onwards large parts of the
                    library are templated and therefore header-only.

contrib/            The contrib directory contains (non-standard)
                    tools by third parties. Typically, there is no
                    need for users of Shark to deal with these
                    tools directly.

gpl-3.0.txt         GNU general public license, version 3.


Note:
Depending of the type of Shark distribution (binary or source
package, or current repository snapshot) not all of theses files
and directories are present.



PACKAGE STRUCTURE
-----------------

  >> Note for users of Shark 2: <<
  The internal structure of the Shark library has changed in the
  transition to version 3. The old infrastructure packages Array, Rng,
  and FileUtil, as well as parts of LinAlg, have been replaced with
  more modern solutions provided by Boost. The machine learning
  related components EALib, MOO-EALib, Mixture, ReClaM, and TimeSeries
  have been unified and organized into competely new interfaces.
  Therefore there is no one-to-one correspondance between files or
  even concepts in version 3 and in older versions of Shark. In fact,
  the lion's share of the library has been rewritten from scratch,
  and this is also reflected in a completely new structure. In
  particular, many of the rather independent sub-modules (such as
  Mixture and MOO-EALib) have been unified. They now share the same
  top-level interfaces and thus form a coherent learning architecture.

The organization of the include/ directory reflects the structure of
the Shark library. It consists of the following modules:


GENERAL INFRASTRUCTURE:

LinAlg              Data structures and algorithms for typical
                    linear algebra computations. For (dense and
                    sparse) vector and matrix classes Shark relies
                    on Boost uBLAS. Many higher level algorithms
                    (such as singular value decomposition) are
                    still implemented by the library itself.

Statistics          This component is new in Shark 3. It wraps the
```

```
                    capabilities of Boost accumulators, and it
                    provides tools that appear regularly in machine
                    learning, such as the Mann-Whitney U-test (also
                    known as the Wilcoxon rank-sum test).


LEARNING INFRASTRUCTURE:

Core                The core module is the central place for all
                    top-level interfaces. In addition it holds a
                    few infrastructure classes, such as exceptions.

Data                The data module hosts data containers that have
                    been specifically designed for the needs of
                    machine learning code. Also, data can be
                    imported and exported from and to different
                    standard machine learning data file formats.


MACHINE LEARNING:

Models              Models are adaptive systems, the architectures
                    on top of which (machine) learning happens.
                    Shark features a rich set of models, from simple
                    linear maps to (feed-forward and recurrent)
                    neural networks, support vector machines, and
                    different types of trees. Models can also be
                    concatenated with data format converters and
                    other models.

ObjectiveFunctions  This module collects different types of cost,
                    fitness, or objective functions for learning.
                    The bandwidth includes data-dependent error
                    functions based on simple loss functions,
                    cross-validation, area under the ROC curve, and
                    different objectives used for model selection.

Algorithms          All actual learning algorithms reside in this
                    module. There are two main groups of learning
                    algorithms, namely iterative optimizers and
                    more specialized model trainers. General
                    optimizers are organized into direct search
                    and gradient-based optimization. Specialized
                    algorithms for linear programming (a part of
                    GLPK, the GNU linear programming kit) and
                    quadratic programming for training of non-linear
                    support vector machines are included. Shark
                    also ships with algorithms for efficient
                    nearest neighbor search.

Fuzzy               The fuzzy module provides classes for the
                    representation of linguistic terms, variables,
                    operators and rules, as well as fuzzy logic
                    interference engines and controllers.

Unsupervised        This module contains the Shark implementation
                    of restricted Bolzmann machines (RBMs),
                    a recent experimental feature of Shark.
```