The Shogun Machine Learning Toolbox

heiko.strathmann@gmail.com

13th London Machine Learning Meetup

May 27, 2014

Outline

Overview

Machine Learning Features

Technical Features

Community

A bit about me

- Heiko Strathmann - `http://herrstrathmann.de/`
- 2004-2008 - Jazz guitar
- 2008-2013 - BSc in CS, MSc in ML, Research assistant

- Phd student at UCL
- Kernels, Bayesian & MCMC, Bioinformatics, Open-Source
- Involved in Shogun since 2010

A bit about Shogun

- Made public in 2004
- Currently 8 core-developers + 20 regular contributors
- Academic background, originally SVM & Bioinformatics
- In Google Summer of Code since 2010 (29 projects!)
- Open-Source: GPL (changing to BSD)

Ohloh - Summary

In a Nutshell, SHOGUN...

... has had 25,187 commits made by 126 contributors
representing 588,445 lines of code

... is mostly written in C++
with a very low number of source code comments

... has a well established, mature codebase
maintained by a very large development team
with stable Y-O-Y commits

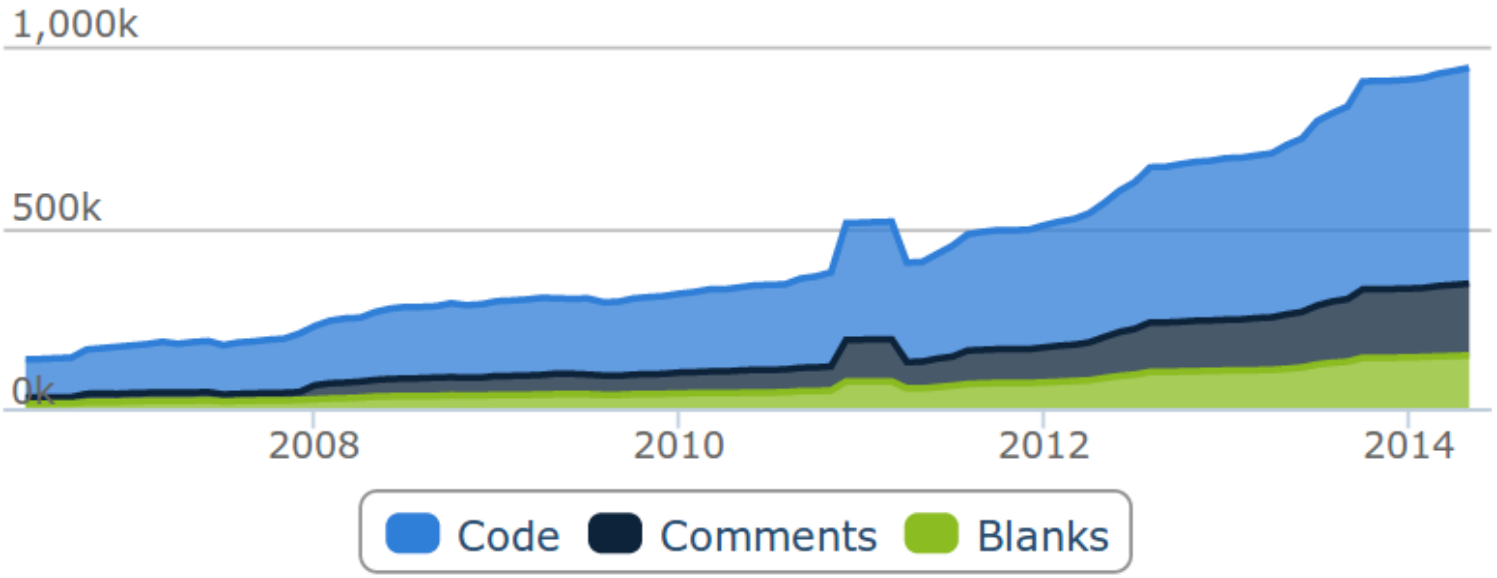
... took an estimated 162 years of effort (COCOMO model)
starting with its first commit in June, 2006
ending with its most recent commit about 3 hours ago

Ohloh - Code

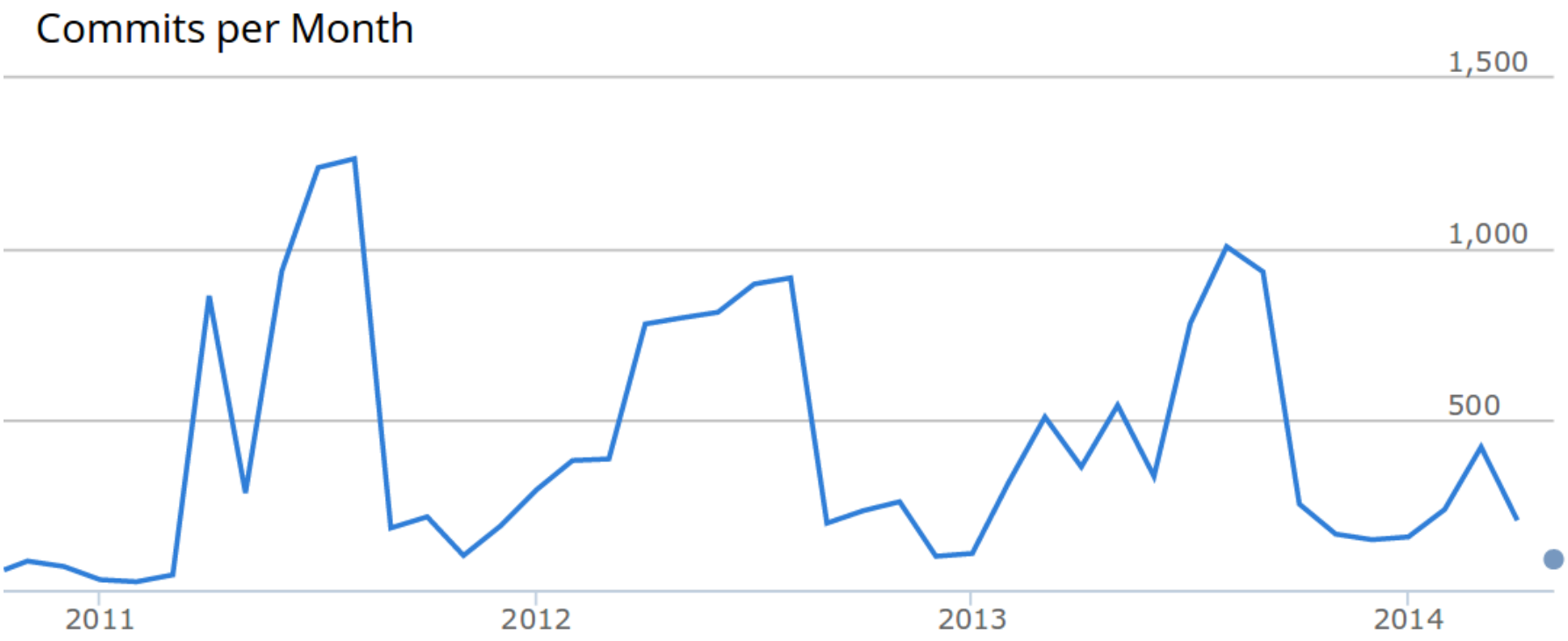
Languages

C++	50%	C	35%
Python	7%	12 Other	8%

Lines of Code



Ohloh - Commits



Outline

Overview

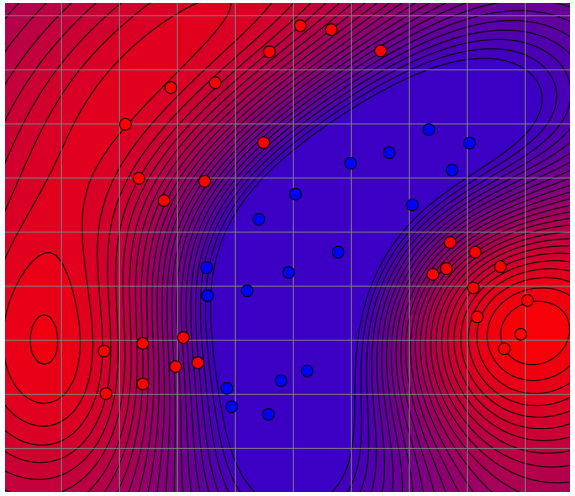
Machine Learning Features

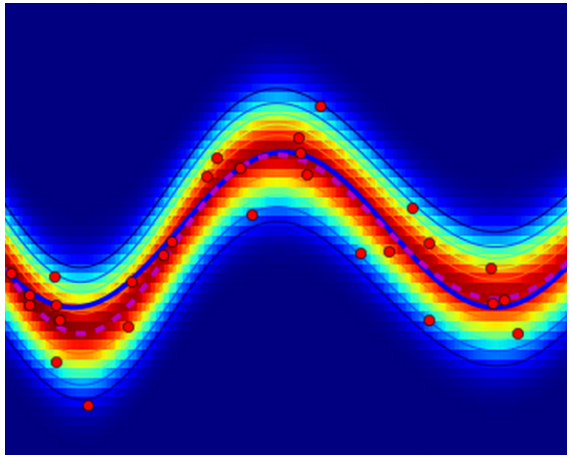
Technical Features

Community

Supervised Learning

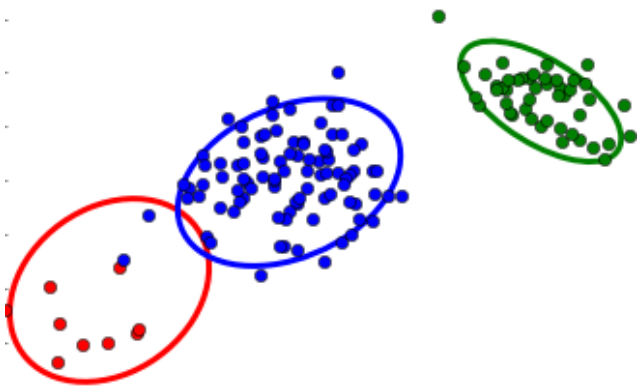
- Given: $\{(x_i, y_i)\}_{i=1}^n$, want: $y^*|x^*$
- Classification: y discrete
  - Support Vector Machine
  - Gaussian Processes
  - Logistic Regression
  - Decision Trees
  - Nearest Neighbours
  - Naive Bayes
- Regression: y continuous
  - Gaussian Processes
  - Support Vector Regression
  - (Kernel) Ridge Regression
  - (Group) LASSO

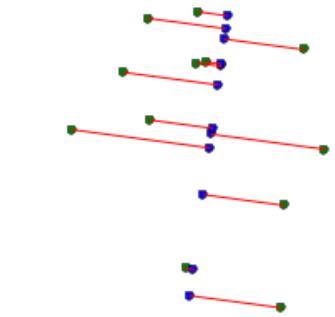


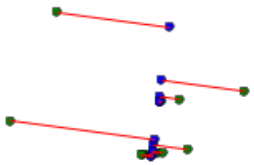


Unsupervised Learning

- Given: $\{x_i\}_{i=1}^n$, want notion of $p(x)$
- Clustering:
  - K-Means
  - (Gaussian) Mixture Models
  - Hierarchical clustering
- Latent Models
  - (K) PCA
  - Latent Discriminant Analysis
  - Independent Component Analysis
- Dimension reduction
  - (K) Locally Linear Embeddings
  - Many more...

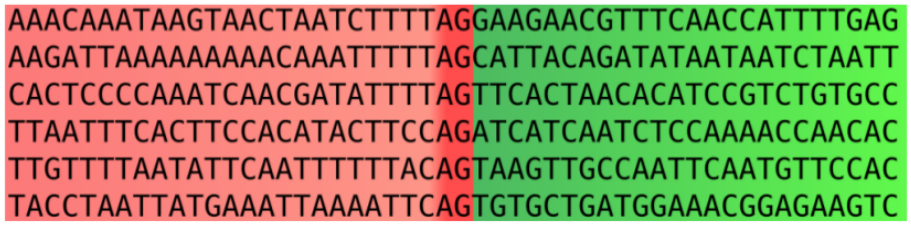






Large-Scale Linear Learning

- Given: $\{(x_i, y_i)\}_{i=1}^n$ really large, want: $y^* = w^T x^*$
- Linear SVMs
  - LibLinear
  - Coffin framework
  - Feature hashing & efficient data representation
- Applications
  - Splice Site prediction: 50m examples of 200m dimensions
  - Face recognition: 20k examples of 750k dimensions



Kernel Frequentists

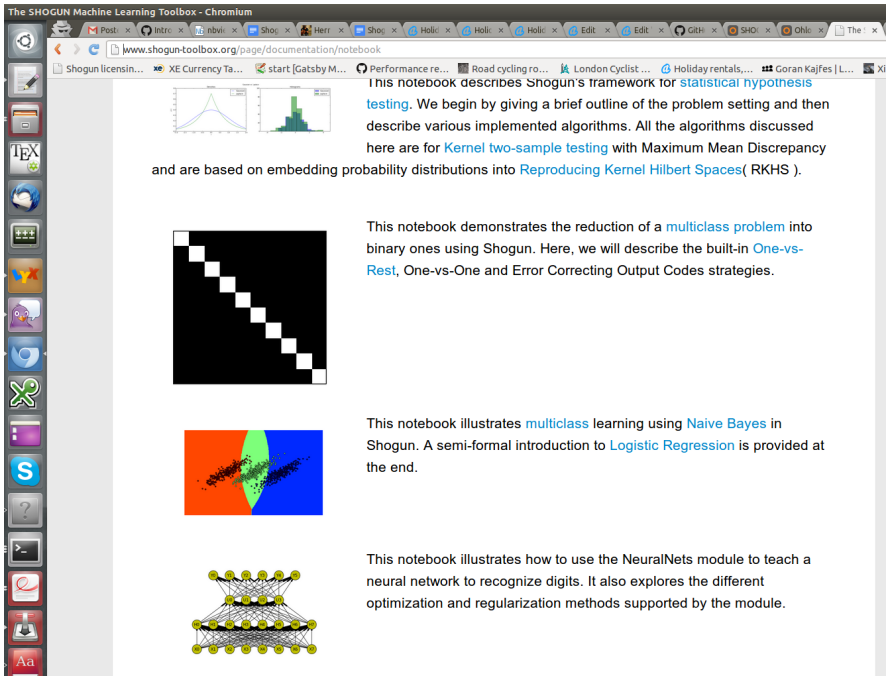
- Two-Sample Testing: Given $\{x_i\}_{i=1}^n, \{y_j\}_{j=1}^m$, is $p(x) = q(y)$?
- Independence Testing: Given $\{(x_i, y_i)\}_{i=1}^n$, is $p(x, y) = p(x)q(y)$?

. . . no doubt there is great pressure on gouvernements and municipal governments in relation to the issue of child care, but the services de garde, reality is that there have been no cuts to child care funding from the federal to the provinces. In fact, we have increased federal investments for early childhood development. . .

. . . il estévident que les ordres de gouvernements provinciaux et municipaux subissent de fortes pressions en ce qui concerne les services de garde, mais le gouvernement n'a pas réduit lefinancement qu'il verse aux provinces pour les services de garde. Au contraire, nous avons augmenté le financement fédéral pour le développement

And more

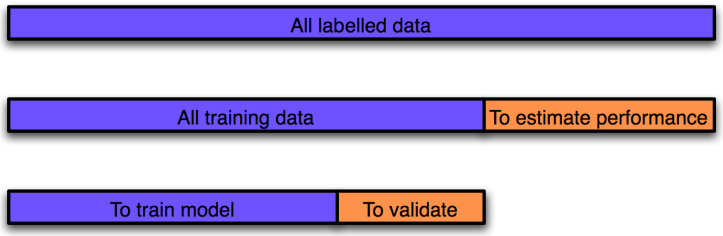
- Multiple Kernel Learning
- Structured Output
- Metric Learning
- Large-Scale log-determinants
- Gaussian Processes
- Deep Learning
- ...

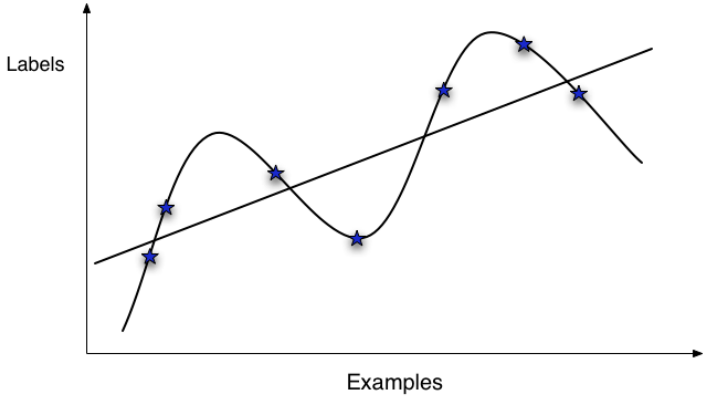


`http://www.shogun-toolbox.org/page/documentation/notebook`

ML in Practice

- Modular data represetation
  - Dense, Sparse, Strings, Streams, ...
  - Multiple types: 8-128 bit word size
  - Preprocessing tools
- Evaluation
  - Cross-Validation
  - Accuracy, ROC, MSE, ...
- Model Selection
  - Grid-Search
  - Gradient based
- Various native file formats, generic multiclass, etc





Outline

Overview

Machine Learning Features

Technical Features

Community

Geeky details

- Written in (proper) C/C++
- Modular, fast, memory efficient
- Unified interface for Machine Learning
- Linear algebra & co: Eigen3, Lapack, Arpack, pthreads, OpenMP, recently GPUs

Class list:

`http://www.shogun-toolbox.org/doc/en/latest/namespaceshogun.html`

Modular language interfaces

- SWIG - `http://www.swig.org/`
- We write:
  - C/C++ classes
  - Typemaps (i.e. 2D `C++` matrix $\Leftrightarrow$ 2D `numpy` array)
  - List of classes of expose
- SWIG generates:
  - Wrapper classes
  - Interface files
- Automagically happens at compile time
- Identical interface for all modular languages:
  - C++, Python, Octave, Java, R, Ruby, Lua, C#
- We are in Debian/Ubuntu, but also Mac, Win, Unix

C/C++

```
#include <shogun/base/init.h>
#include <shogun/kernel/GaussianKernel.h>
#include <shogun/labels/BinaryLabels.h>
#include <shogun/features/DenseFeatures.h>
#include <shogun/classifier/svm/LibSVM.h>

using namespace shogun;

int main()
{
init_shogun_with_defaults();
...
exit_shogun();
return 0;
}
```

C/C++

```
DenseFeatures<float64_t>* train=new
    DenseFeatures<float64_t>(...);
DenseFeatures<float64_t>* =new DenseFeatures<
    float64_t>(...);
BinaryLabels* labels=new BinaryLabels(...);

GaussianKernel* kernel=new GaussianKernel(
    cache_size, width);
svm=new LibSVM(C, kernel, labels);
svm->train(train);

CBinaryLabels* predictions=CLabelsFactory::
    to_binary(svm->apply(test));
predictions->display_vector();

SG_UNREF(svm);
SG_UNREF(predictions);
```

Python

```
from modshogun import *

train=RealFeatures(numpy_2d_array_train)
test=RealFeatures(numpy_2d_array_test)
labels=BinaryLabels(numpy_1d_array_label)

kernel=GaussianKernel(cache_size, width)
svm=LibSVM(C, kernel, labels)
svm.train(train)

predictions=svm.apply(test)

# print first prediction
print predictions.get_labels()[0]
```

Octave

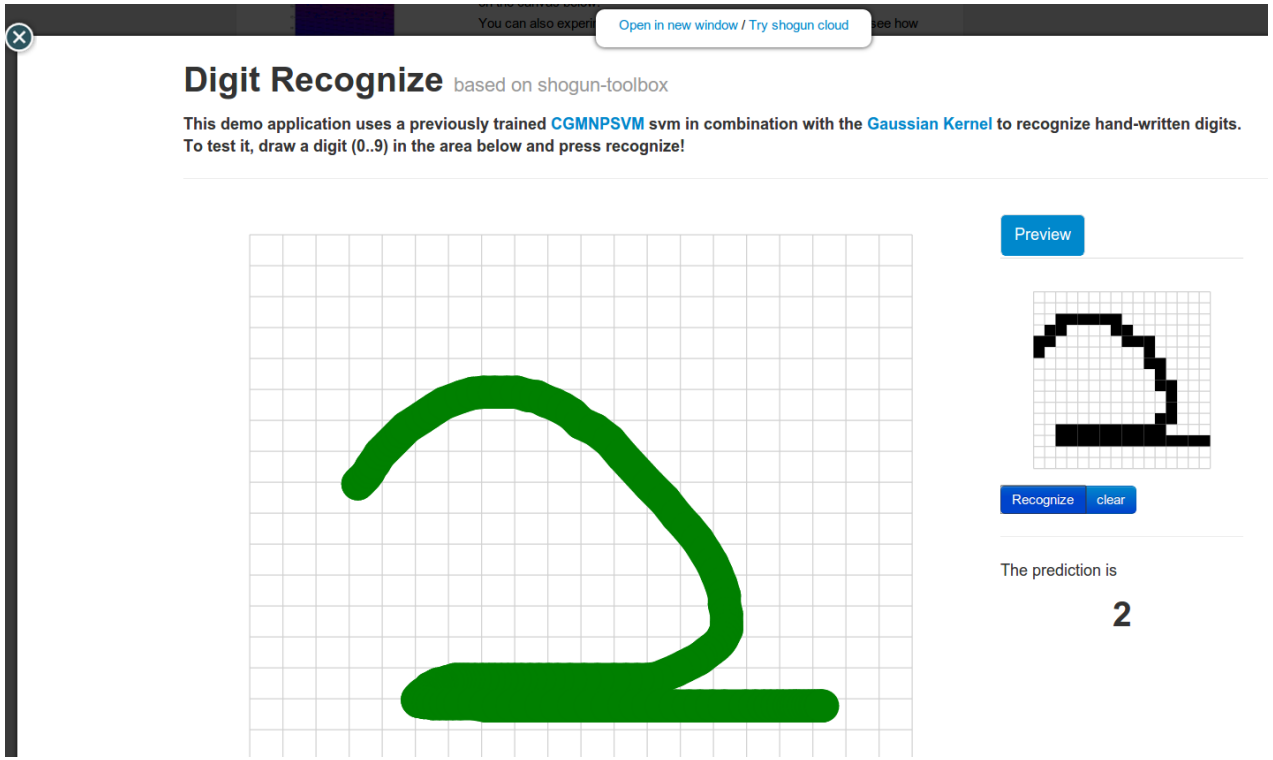
```
modshogun

train=RealFeatures(octave_matrix_train);
test=RealFeatures(octave_matrix_train);
labels=BinaryLabels(octave_labels_train);

kernel=GaussianKernel(cache_size, width);
svm=LibSVM(C, kernel, labels);
svm.train(train);

predictions=svm.apply(test);

% print first prediction
disp(predictions.get_labels()[1])
```

Java

```
import org.shogun.*;
import org.jblas.*;
import static org.shogun.LabelsFactory.to_binary;

public class classifier_libsvm_modular {
static {
System.loadLibrary("modshogun");
}

public static void main(String argv[]) {
modshogun.init_shogun_with_defaults();

RealFeatures train=new RealFeatures(new CSVFile(train_file));
RealFeatures test=new RealFeatures(new CSVFile(test_file));
BinaryLabels labels=new BinaryLabels(new CSVFile(label_fname));
GaussianKernel=new GaussianKernel(cache_size, width);
LibS svm=new LibSVM(C, kernel, labels);
svm.train(train);

// print predictions
DoubleMatrix predictions=to_binary(svm.apply(test)).get_labels();
System.out.println(predictions.toString());
}
}
```

Shogun in the Cloud

- We love IPython notebooks for documentation
- IPython notebook server: try Shogun without installation
- Interactive web-demos

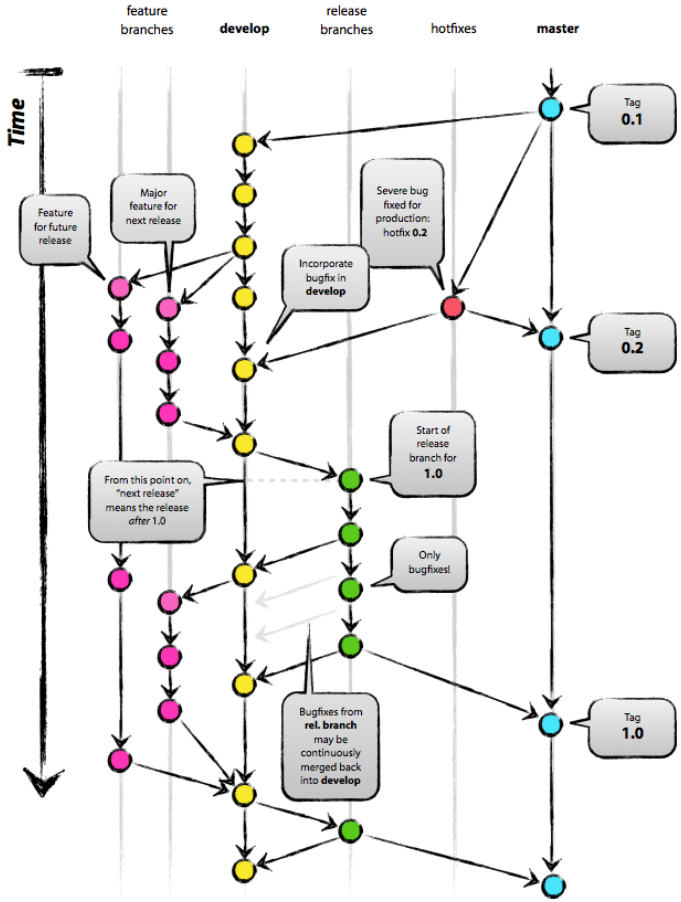


http://www.shogun-toolbox.org/page/documentation/notebook

http://www.shogun-toolbox.org/page/documentation/demo

Development model

- Github
  - Pull requests
  - Intense discussions
- Git flow
  - Feature branches
  - Dev & stable branch
- Extensive unit & integration tests
- Travis: Tests pass before merge
- Buildbot: All OS, nightly binaries, memory checks, notebooks, documentation



http://buildbot.shogun-toolbox.org/

shogun-toolbox	a4e7ce28b8dd... in develop	eb1f080e75db... in develop	67d55553fa37... in develop	d923cb10ed5e... in develop	4bfa55e61686... in develop
FC19 - libshogun	OK	OK		OK	OK
FCRH - libshogun	OK	OK		OK	OK
bsd1 - libshogun	OK	OK		OK	OK
clang34 - static analysis	OK	OK			OK
clang34 - thread analysis	warnings test	warnings test			warnings test
clang34 - undefined behaviour analysis	warnings test	warnings test	warnings test		warnings test
coverity analysis (waiting)					
cyg1 - libshogun (offline, plus 18)					
deb1 - libshogun	OK	OK		OK	OK
deb2 - static_interfaces	OK	OK			OK
deb3 - modular_interfaces	OK	OK			OK
deb4 - python3	OK	OK	OK	OK	OK
debian wheezy - memcheck	failed memory check	failed memory check			failed memory check
nightly_all (waiting)		OK			
nightly_default (waiting)		failed notebooks			
nightly_fedora (waiting)		failed git			
nightly_none (waiting)		OK			
osx1 - libshogun (offline, plus 48)					
osx2 - python (offline)					
precise - libshogun	OK	OK		OK	OK
rpm1 - libshogun	OK	OK		OK	OK

Outline

Overview

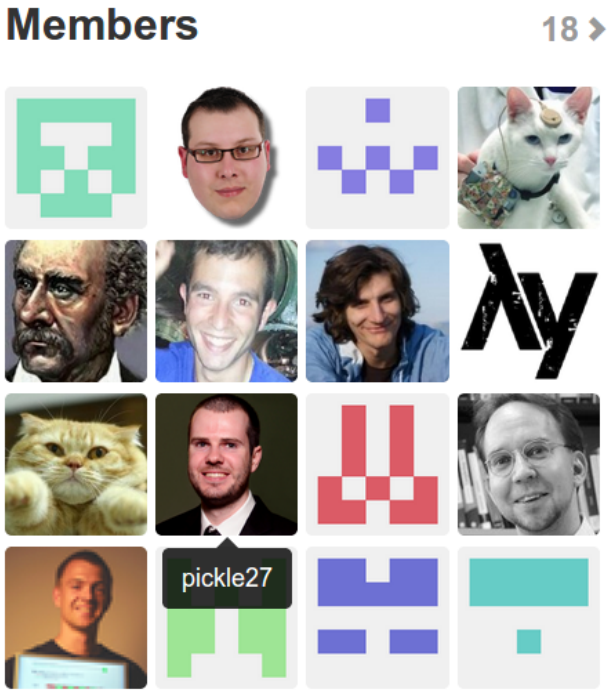
Machine Learning Features

Technical Features

Community

Strong vibrations

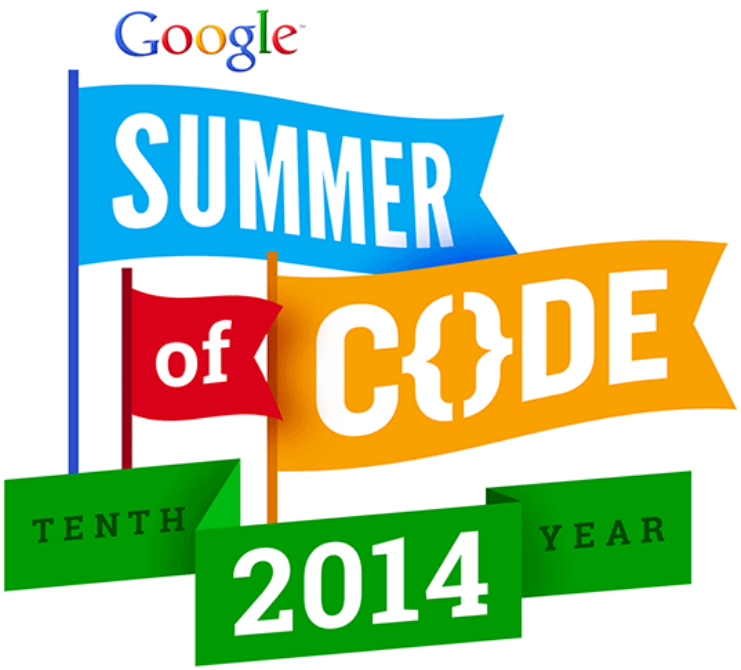
- Active mailing list
- Populated IRC (come say hello)
- Cool team & backgrounds



http://www.shogun-toolbox.org/page/contact/contacts

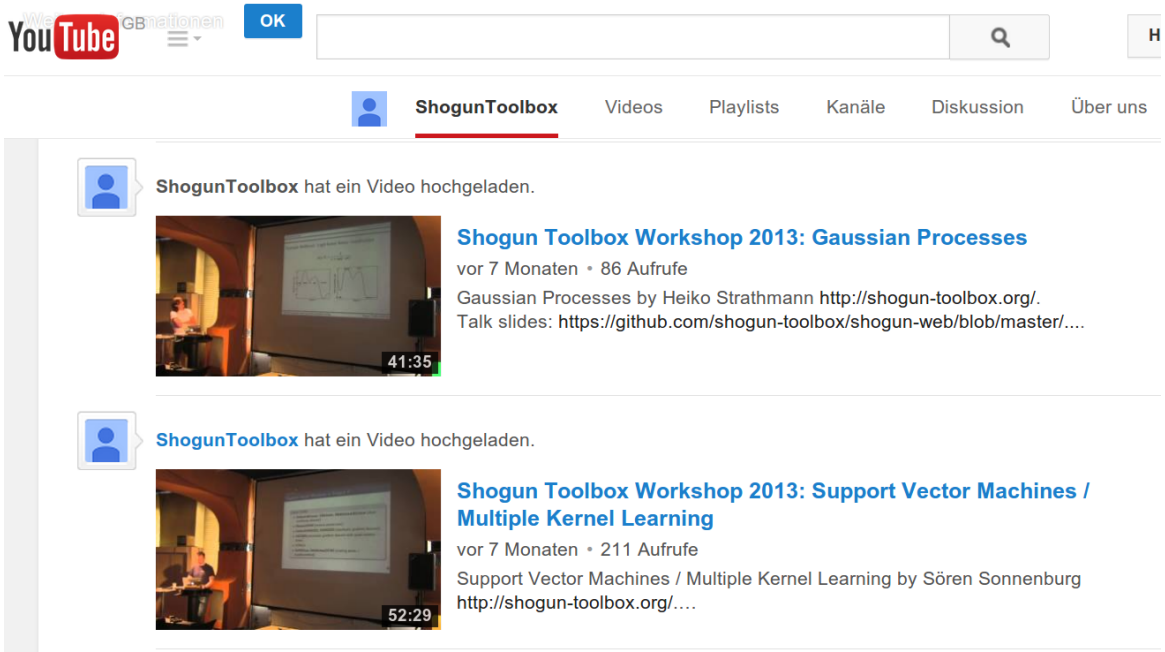
Google Summer of Code

- Student works full time during the summer
- Receives $5000 stipend
- Work remains open-source
- At this moment!
- 29 x 3 months (we have lots of impact)



Future Ideas

- We just founded a non-profit association
- Goal: Take donations and hire a full-time developer
- GPL $\rightarrow$ BSD (industry friendly)
- Shogun in education, fundamental ML
- We organise workshops (next July 28, Berlin)



Help!

- We don't sleep.
- You could:
  - Use Shogun and give us feedback
  - Fix bugs (see github), help us with framework design
  - Write examples and notebooks
  - Write documentation and update our website
  - Implement Super-parametric Massively Parallel Manifold Tree Classification Samplers (tm)
  - Mentor GSoC projects, or join as a student
  - Come to this year's workshop
  - Give us your money (we expose your logo to a large community)

Thanks!

Questions?