

GURLS

Effective machine learning made easy

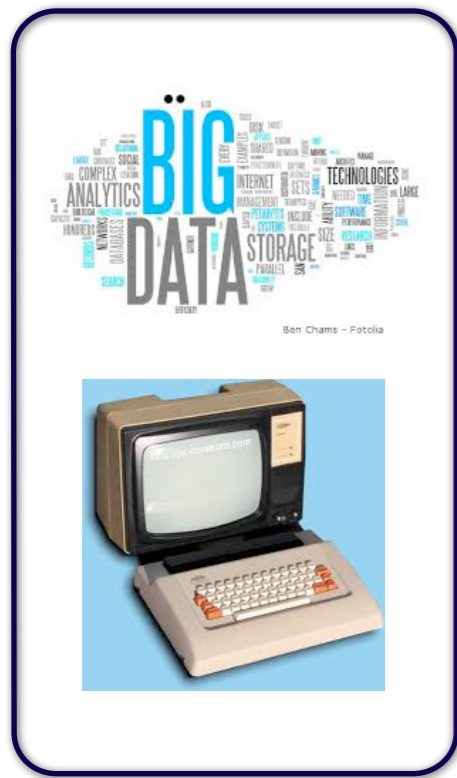
Alessandro Rudi

Carlo Ciliberto and Lorenzo Rosasco

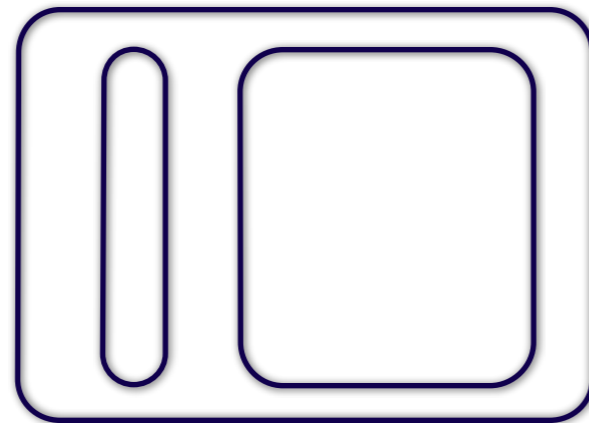
MLCC 2015

Machine Learning in a “Nutshell”

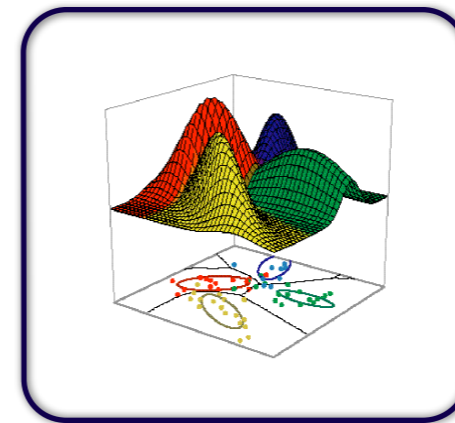
A share of
big data



“Favorite”
pre-processing



Statistical
modeling



+

+

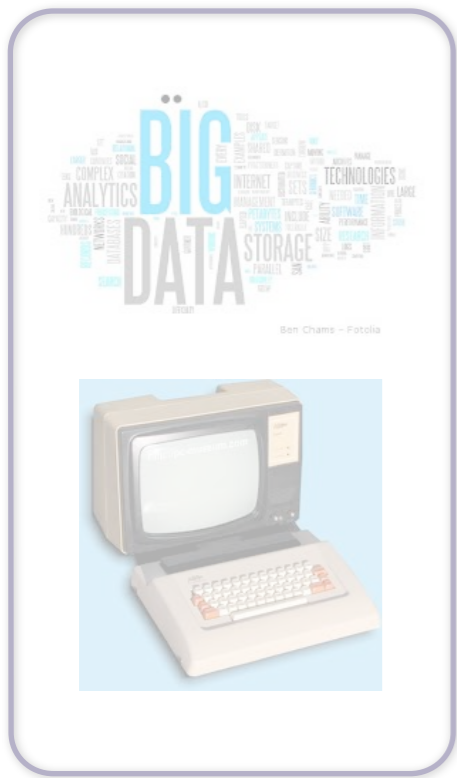
=

Data representation

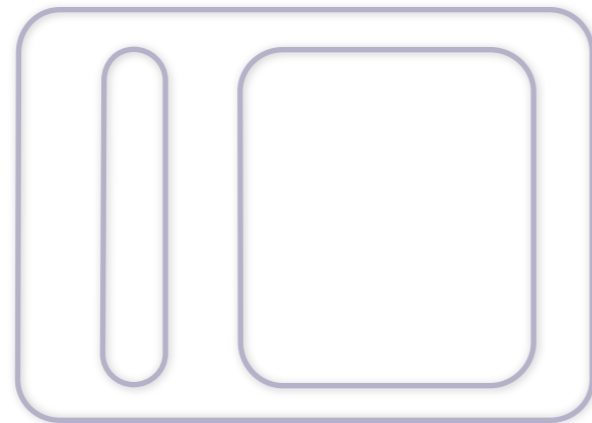


Machine Learning in a “Nutshell”

A share of
big data

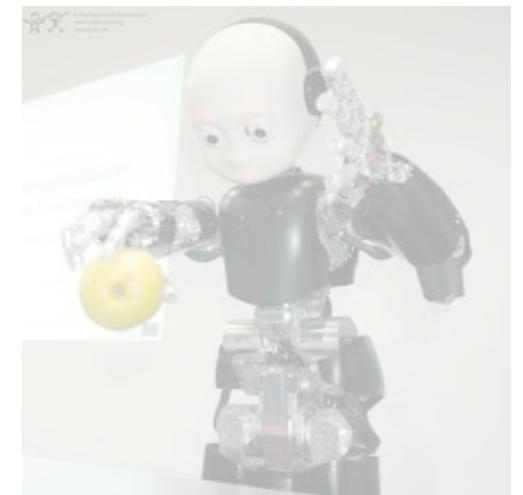
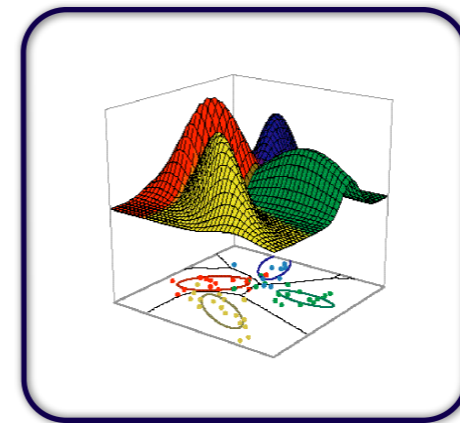


“Favorite”
pre-processing



Data representation

Statistical
modeling



Machine Learning in a “Nutshell”

Several Algorithms Available

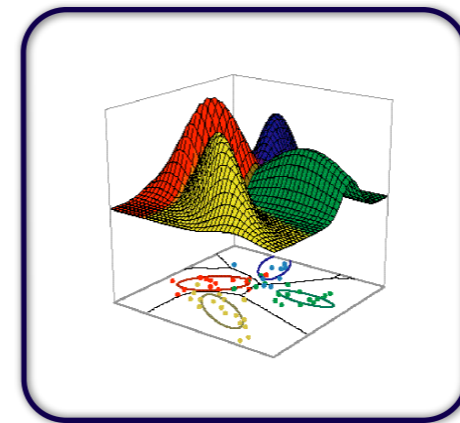


+



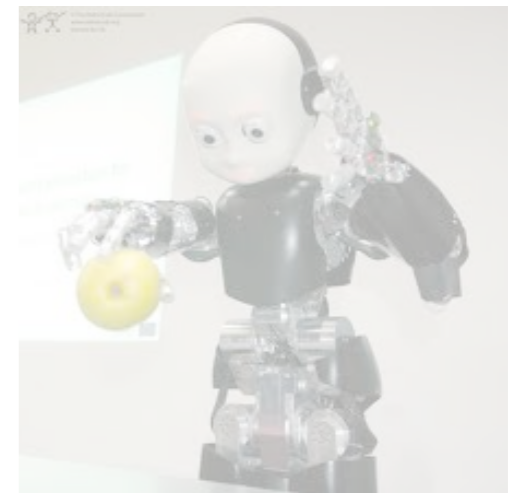
Data representation

+



Statistical modeling

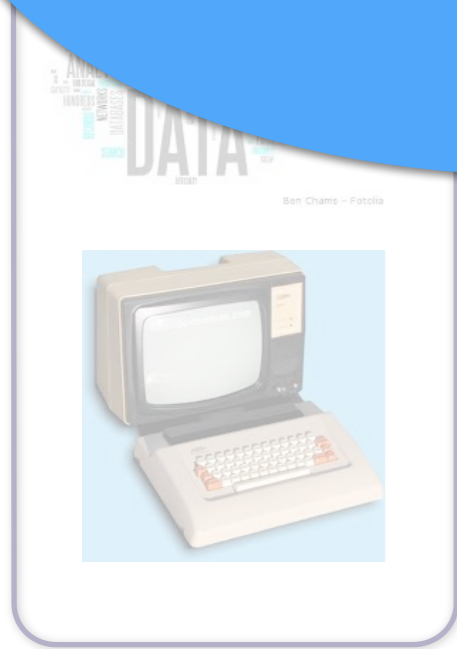
=



Machine Learning in a “Nutshell”

Several Algorithms Available

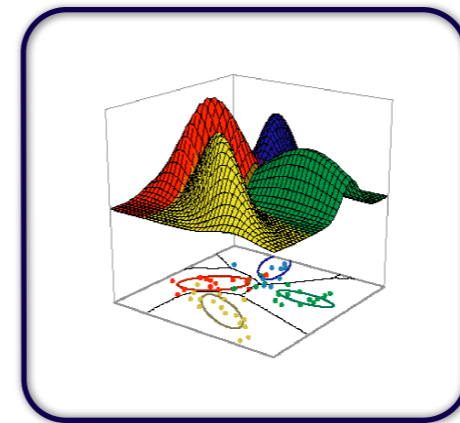
Statistical modeling



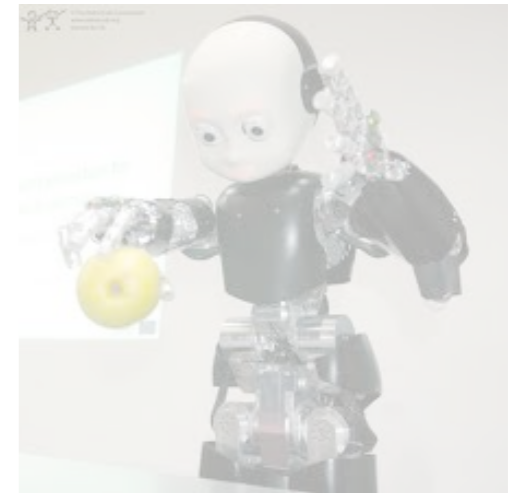
+



+



=



How to select a good hyperparameters range?

Machine Learning in a “Nutshell”

Several Algorithms Available

Statistical modeling

Lots of Boilerplate Code

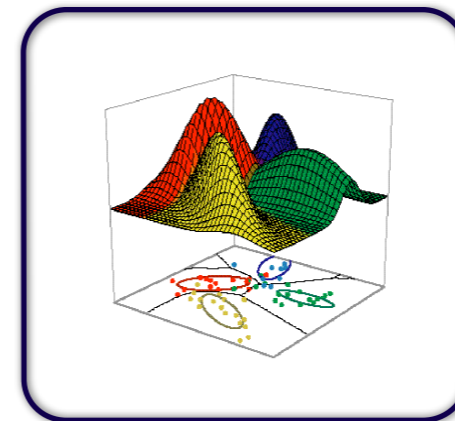
How to select a good hyperparameters range?



+



+



=



GURLS Automates this boring part!

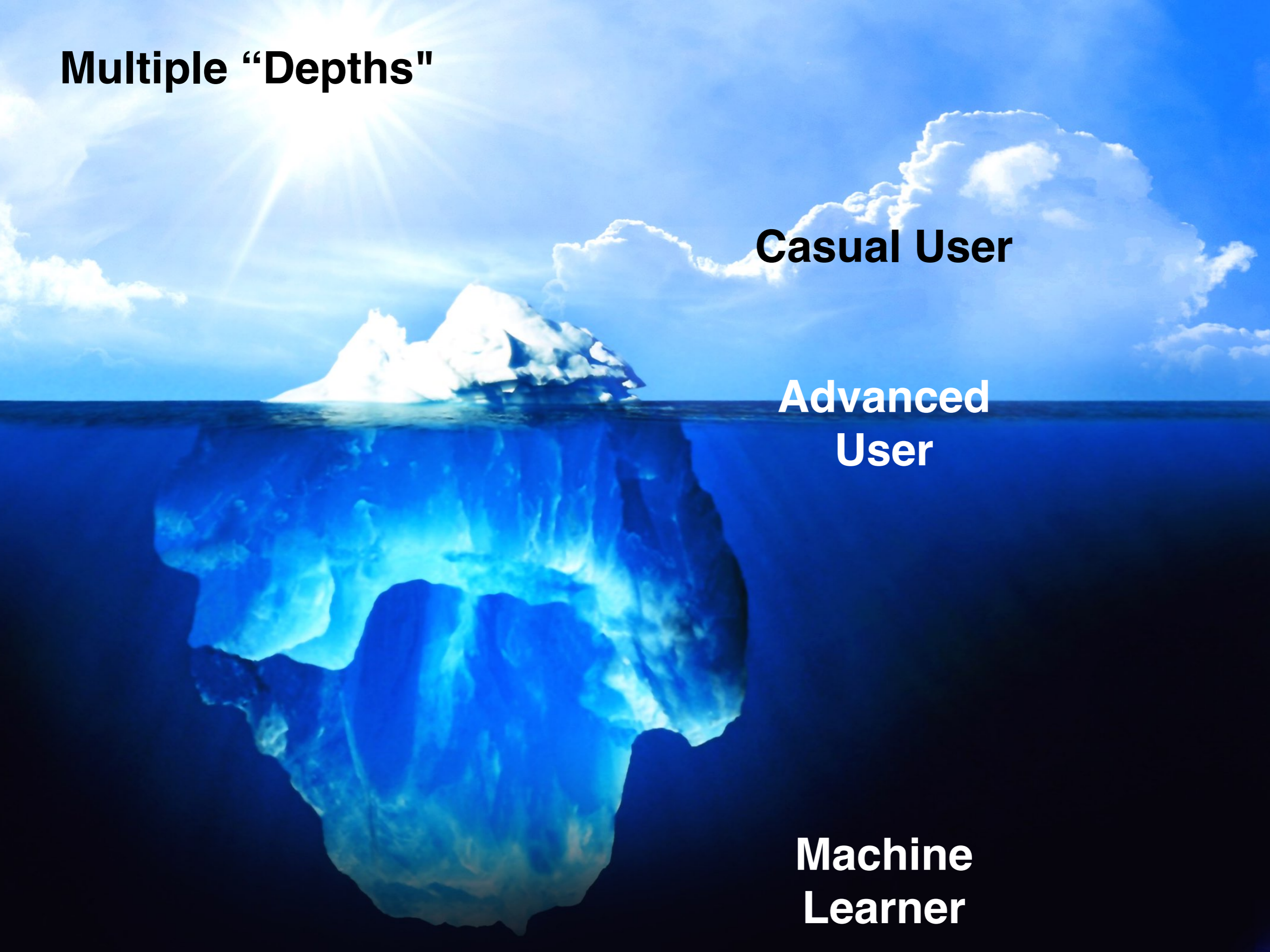


Multiple "Depths"

Casual User

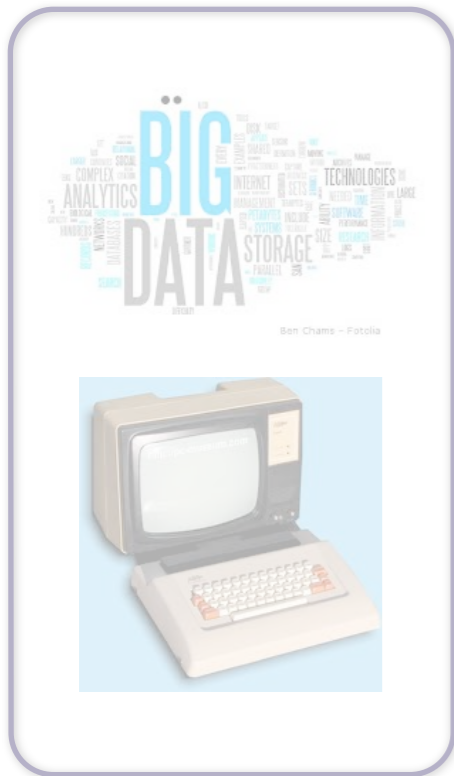
Advanced User

Machine Learner

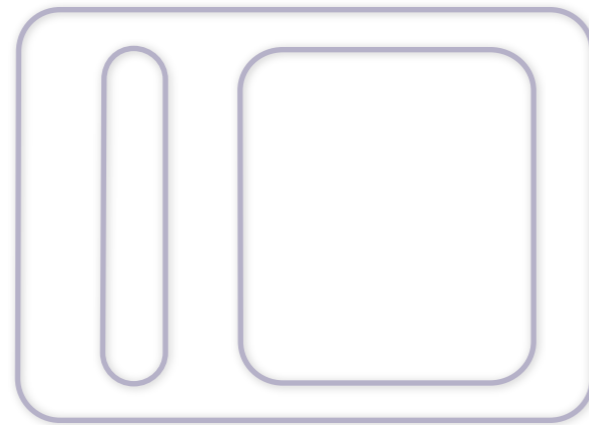


GURLS as a Casual User

A share of
big data

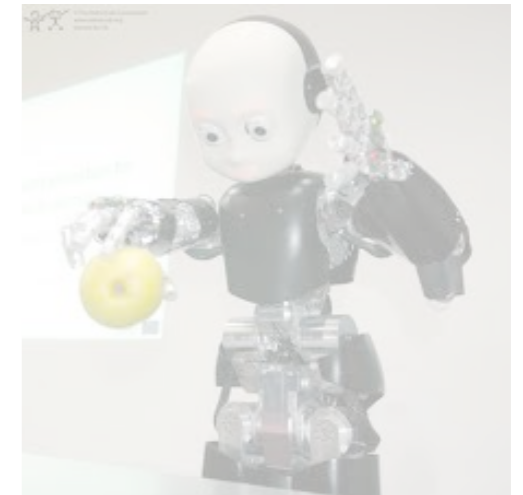
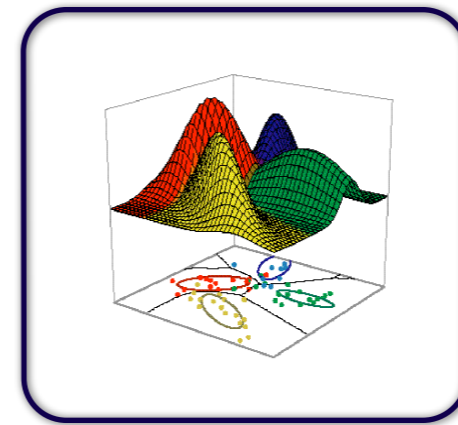


“Favorite”
pre-processing



Data representation

Statistical
modeling

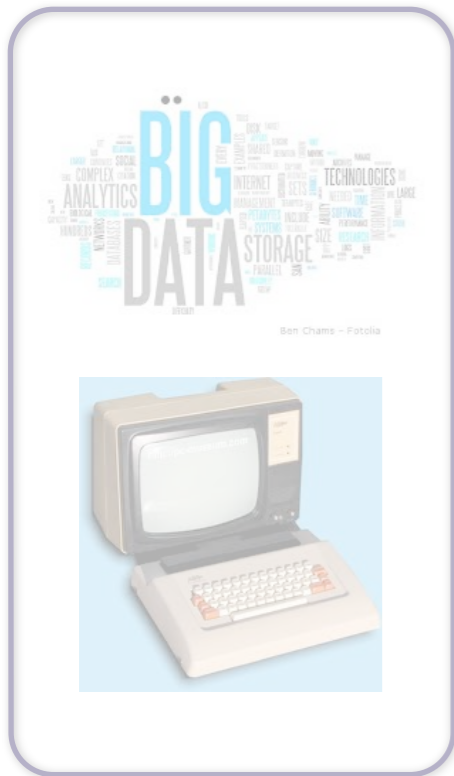


```
model = gurls_train(Xtr, ytr)
```

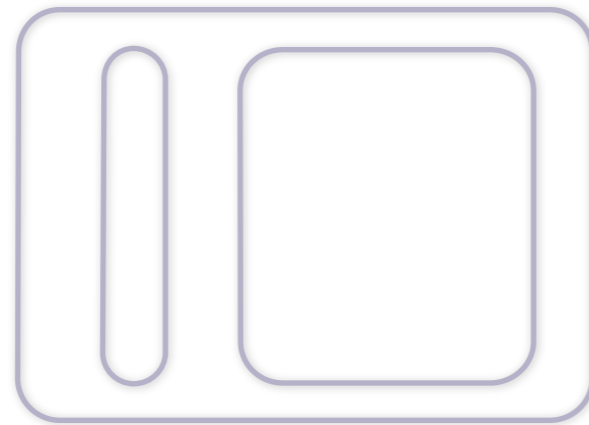
```
ypred = gurls_test(model, Xts)
```

What if we wanted to change... e.g. the kernel?

A share of
big data

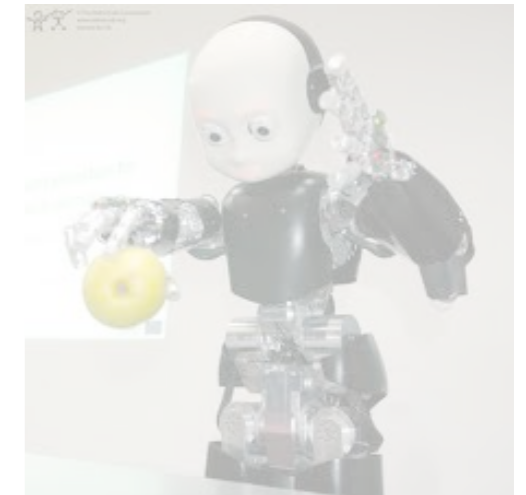
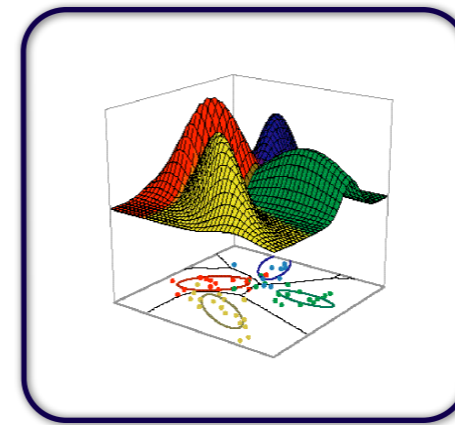


“Favorite”
pre-processing



Data representation

Statistical
modeling



```
model = gurls_train(Xtr, ytr,  
                  'kernel', 'rbf', 'kerpar', 0.1)
```

```
ypred = gurls_test(model, Xts)
```

General Parameters Feeding Syntax

```
model = gurls_train(Xtr, ytr,  
                   'par1', val1, 'par2', val2, ...)
```

```
ypred = gurls_test(model, Xts)
```

Features of the Library

Algorithms

- KRLS with
 - Tikhonov
 - Landweber
 - Nu-method
 - Truncated-SVD
 - Conjugate Gradient
- Kernel Logistic Regression
- Gaussian Processes
- Random Features

Kernels

- Linear
- Polynomial
- RBF
- Chisquared
- Quasi-periodic

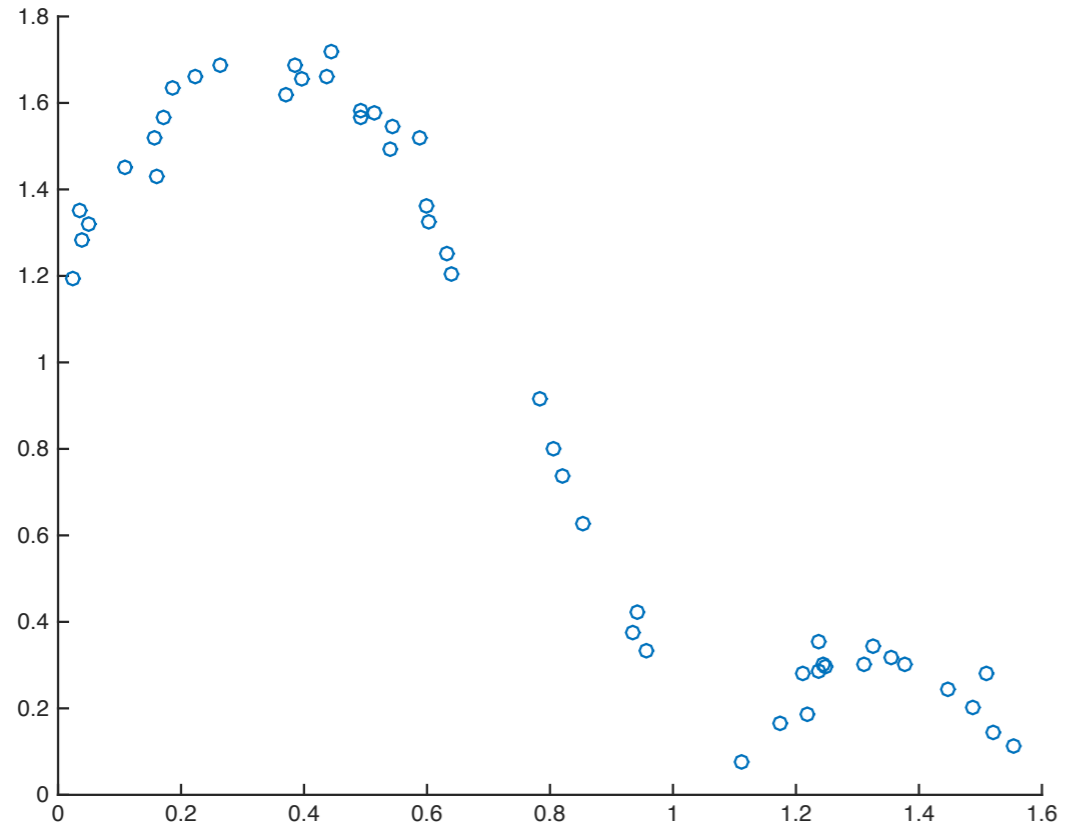
Model Selection

- Performance Measures:
 - RMSE
 - Macroavg
 - Precision/Recall
- Automatic Parameter Tuning:
 - Automatic split & randomization
 - Hold out
 - Leave-one out
 - k-fold
- Automatic Range for Hyperparameters:

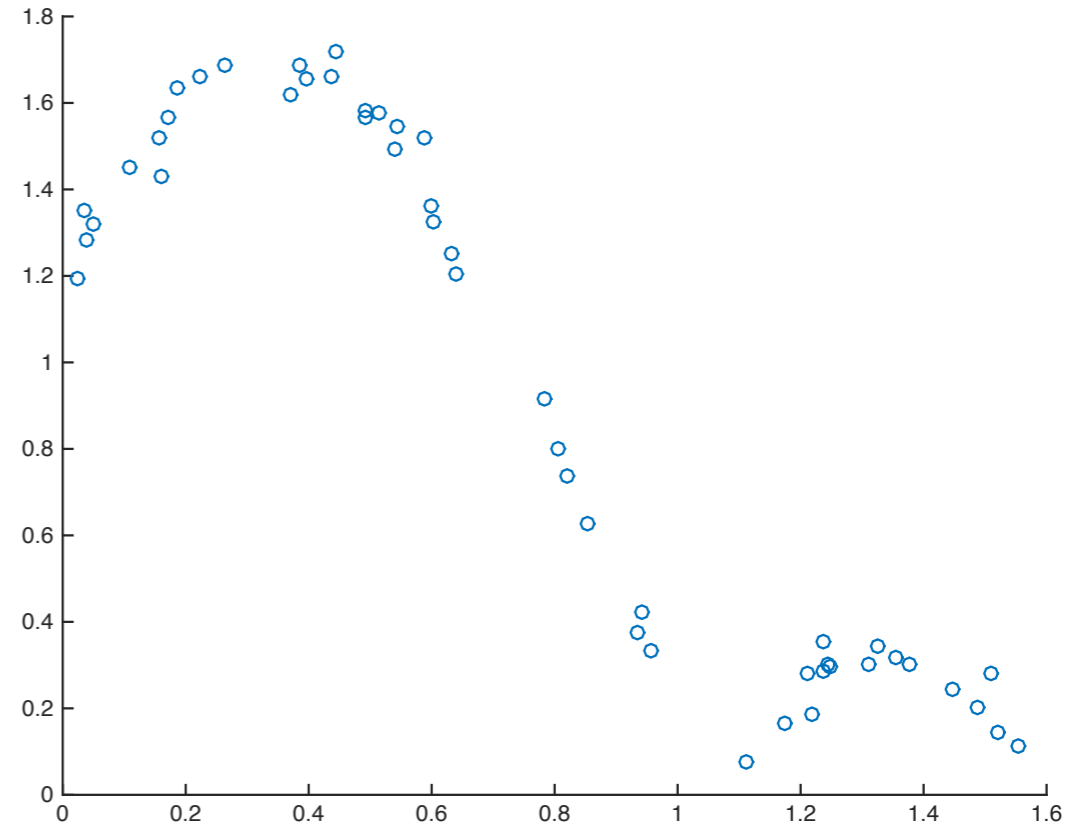
MATLAB & C++ Interfaces

Example

Linear RLS

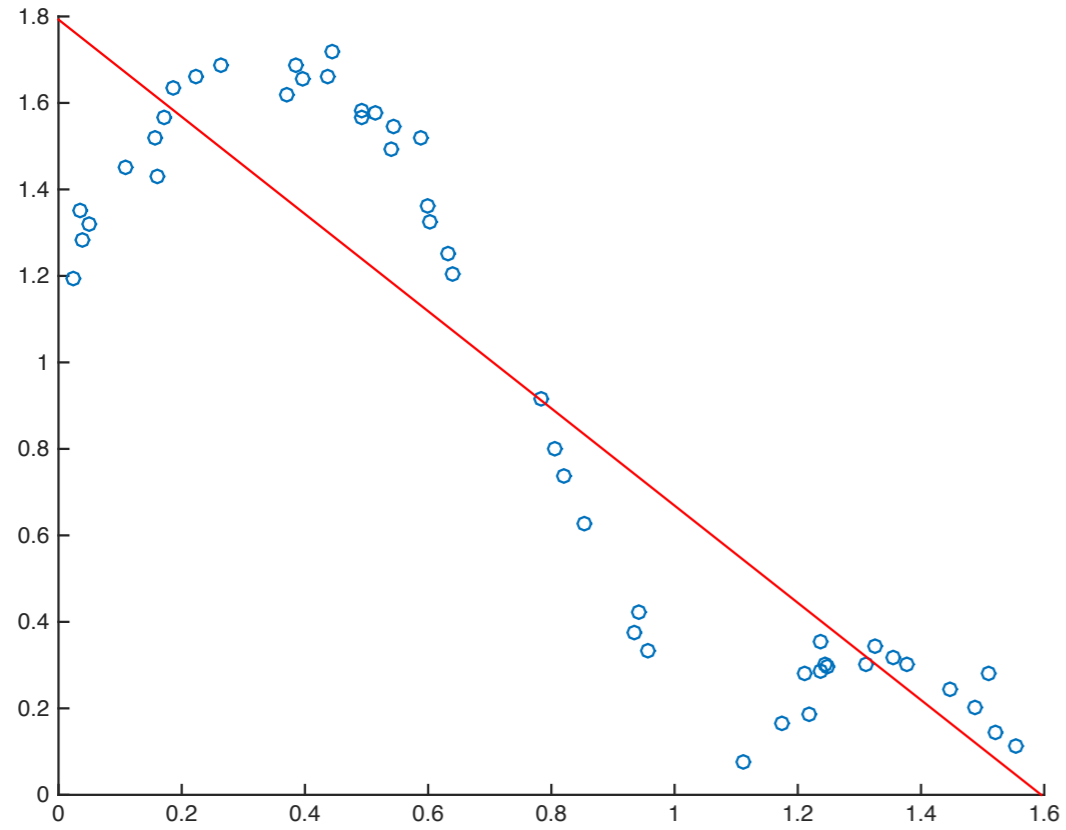


KRLS - RBF Kernel Landweber Filter

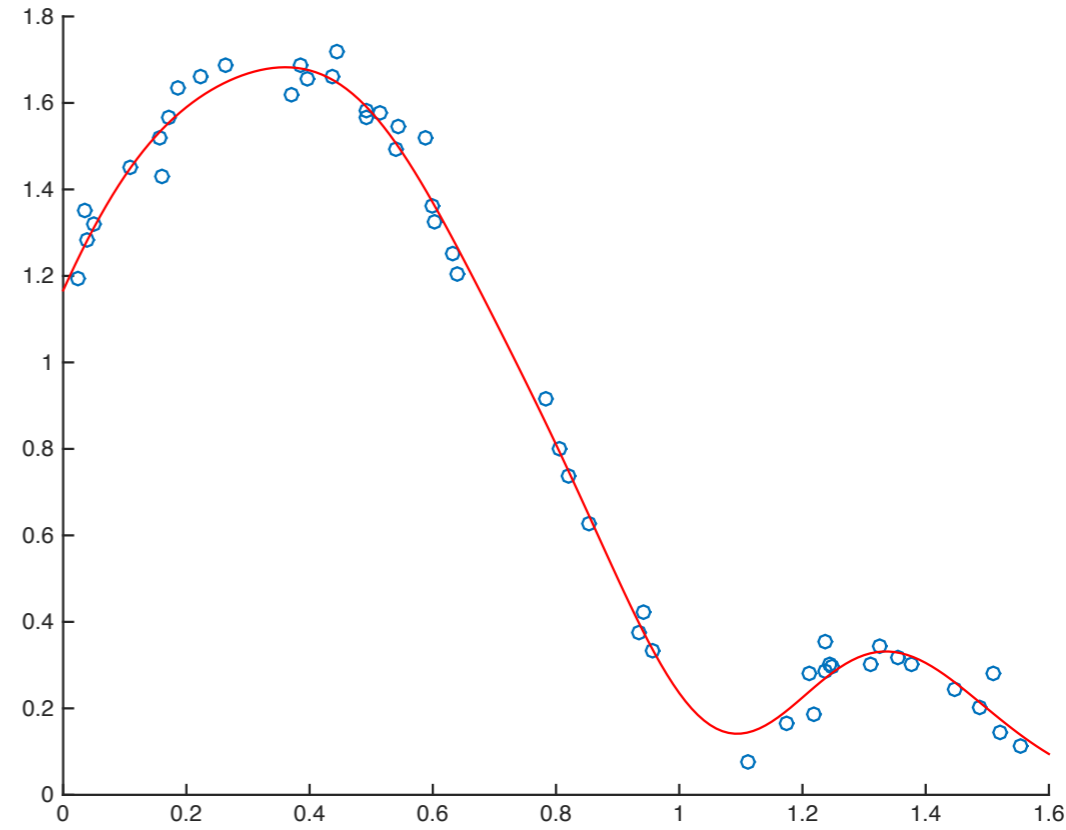


Example

Linear RLS

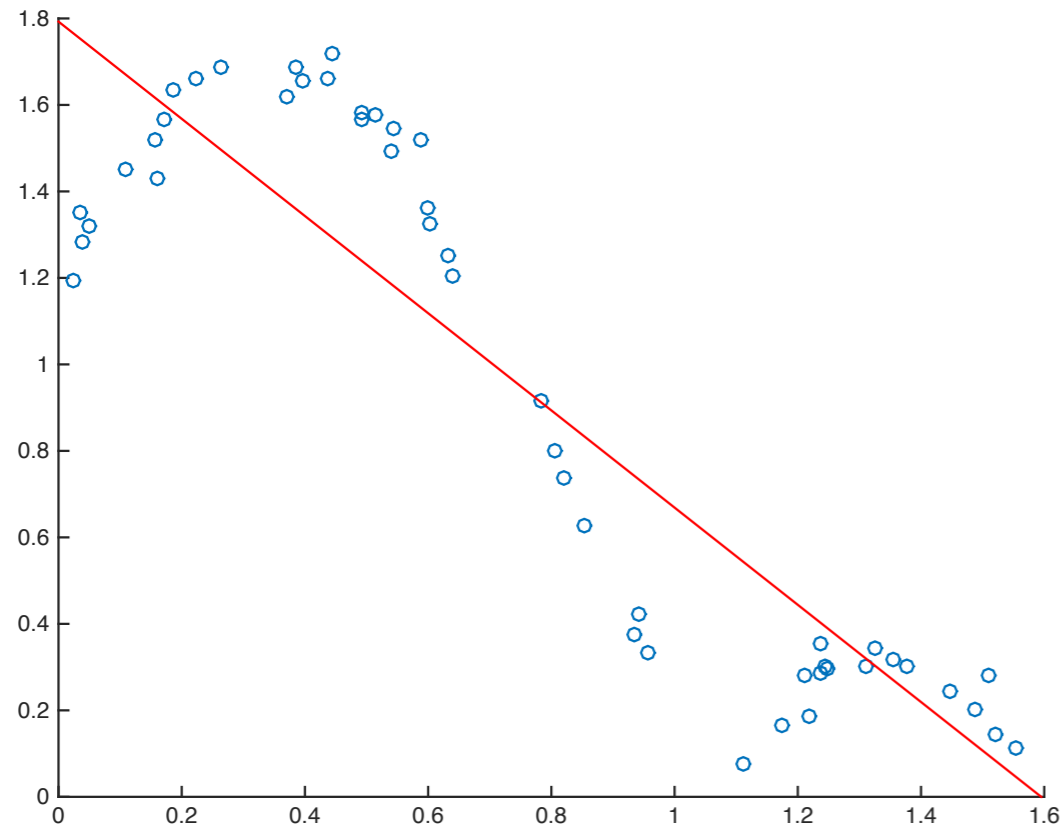


KRLS - RBF Kernel Landweber Filter



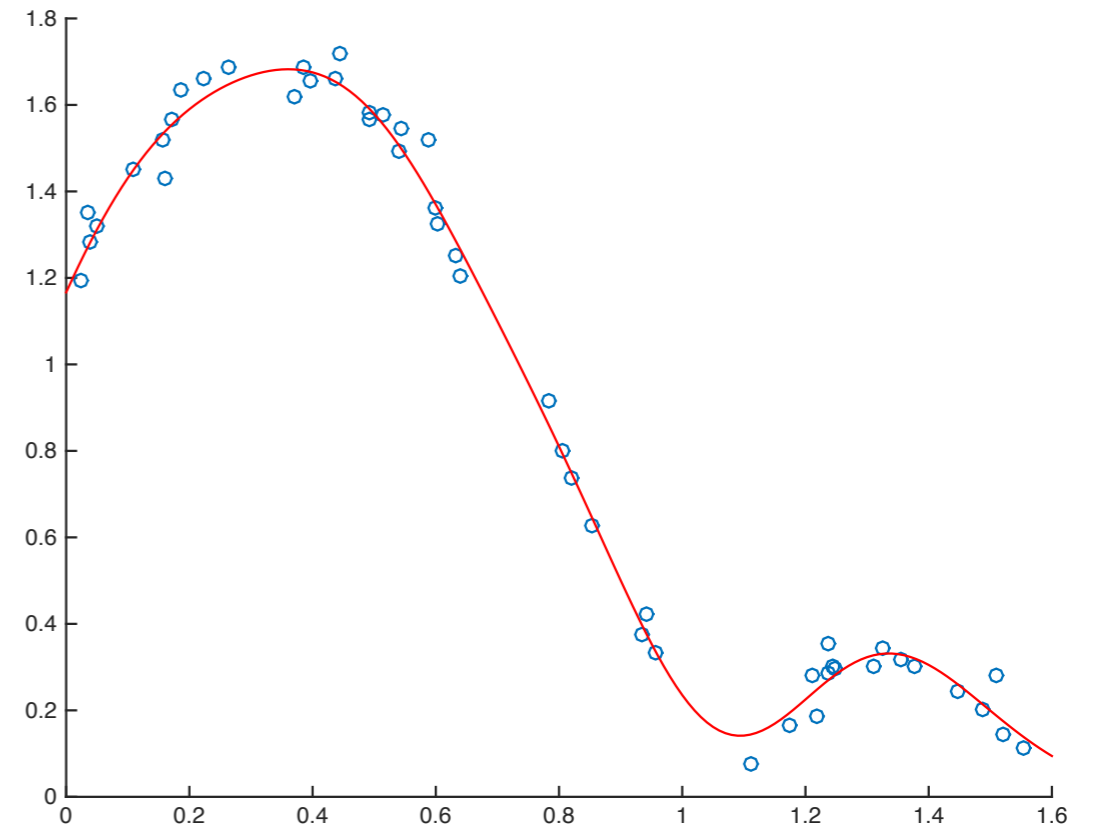
Example

Linear RLS



```
gurls_train(Xtr, ytr,  
            'algorithm', 'lrls')
```

KRLS - RBF Kernel Landweber Filter

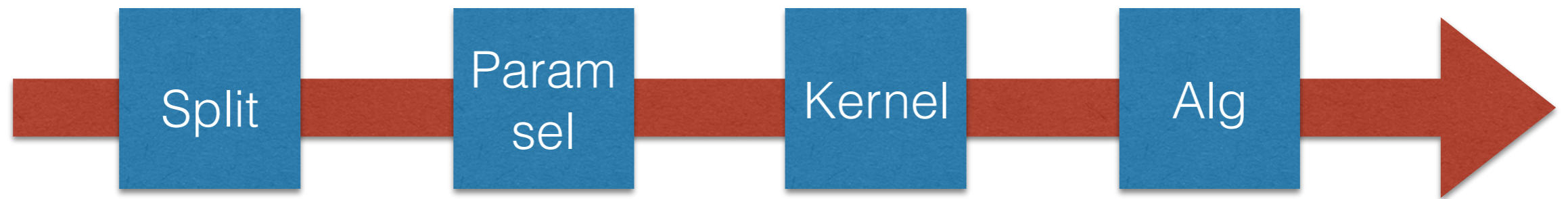


```
gurls_train(Xtr, ytr,  
            'algorithm', 'krls',  
            'kernelfun', 'rbf',  
            'filter', 'land')
```

UNDER THE HOOD

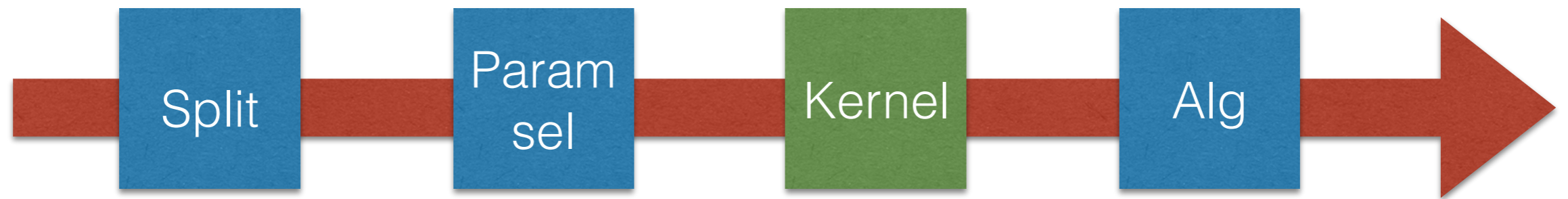
The Pipeline

Under the hood - The Pipeline



highly modular, i.e. reuse of code components

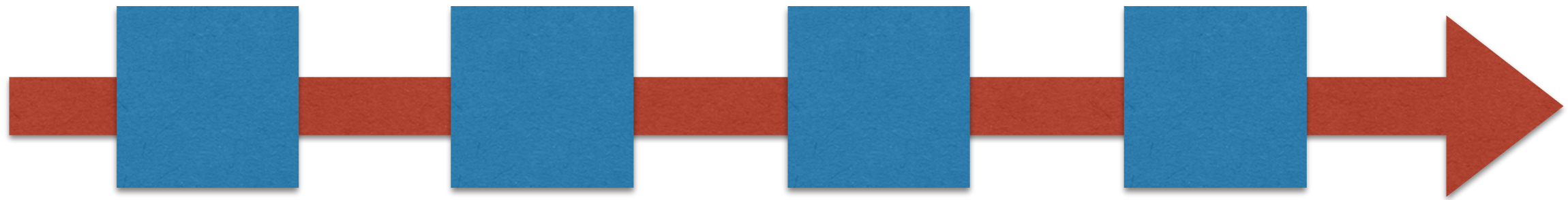
Under the hood - The Pipeline



highly modular, i.e. reuse of code components

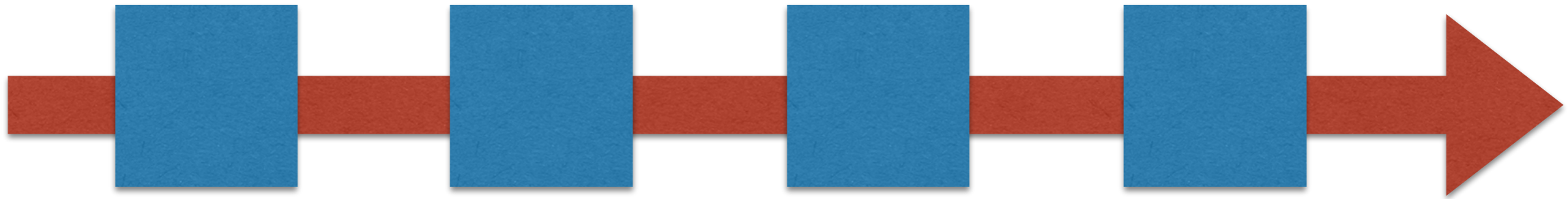
Example: from gurls_train to the pipeline

```
gurls_train(Xtr, ytr, 'algorithm', 'lrls')
```



Example: from gurls_train to the pipeline

```
gurls_train(Xtr, ytr, 'algorithm', 'lr1s')
```

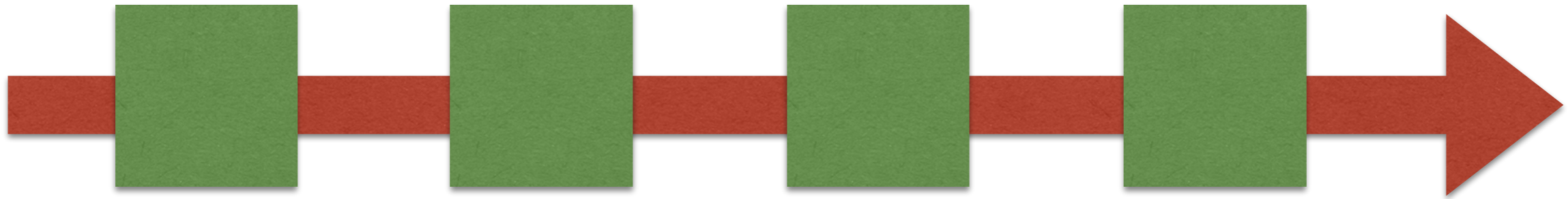


With the pipeline...

```
opt = gurls_defopt('');  
opt.seq = {'split:ho', 'paramsel:hoprimal', 'rls:primal'};  
opt.process{1} = [2, 2, 2];  
  
model = gurls(Xtr, ytr, opt, 1);
```

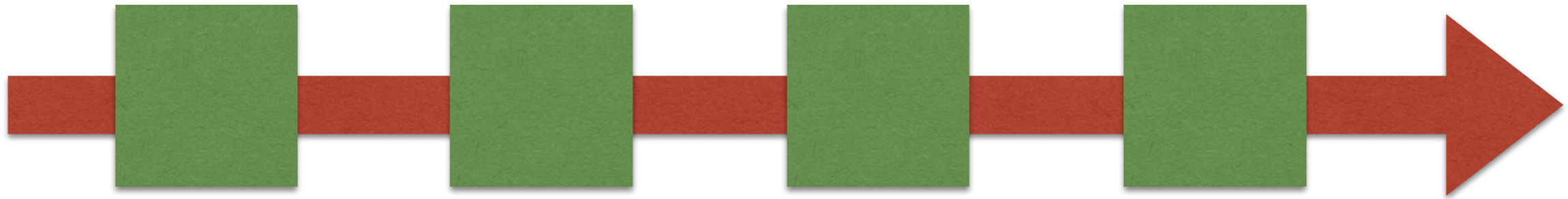
Examples with what we have seen

```
gurls_train(Xtr, ytr, 'algorithm', 'krls', 'kernelfun', 'rbf', 'filter', 'land')
```



Examples with what we have seen

```
gurl_s_train(Xtr, ytr, 'algorithm', 'krls', 'kernelfun', 'rbf', 'filter', 'land')
```

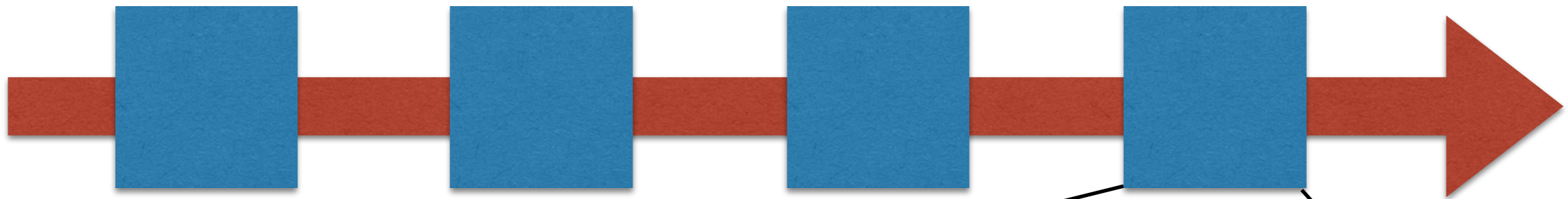


With the pipeline...

```
opt = gurl_s_defopt('');  
opt.newprop('paramsel.guesses', 100);  
opt.paramsel.optimizer = str2func('rls_landweberdual');  
opt.kernel.func = str2func('kernel_rbf');  
opt.seq = {'split:ho', 'paramsel:hokrls', 'kernel:rbf', 'rls:landweberdual'};  
opt.process{1} = [2, 2, 2, 2];  
  
model = gurl_s(Xtr, ytr, opt, 1);
```

Structure of a stage

```
gurls_train(Xtr, ytr, 'algorithm', 'lr1s')
```



```
function [cfr] = rls_primal (X, y, opt)
```

```
% rls_primal(X,y,opt)
```

```
% computes a classifier for the primal formulation of RLS.
```

```
% The regularization parameter is set to the one found in opt.paramsel.
```

```
% In case of multiclass problems, the regularizers need to be combined with the opt.singlelambda function.
```

```
%
```

```
% INPUTS:
```

```
% -OPT: struct of options with the following fields:
```

```
%   fields that need to be set through previous gurls tasks:
```

```
%     - paramsel.lambdas (set by the paramsel_* routines)
```

```
%   fields with default values set through the defopt function:
```

```
%     - singlelambda
```

```
%
```

```
%   For more information on standard OPT fields
```

```
%   see also defopt
```

```
%
```

```
% OUTPUT: struct with the following fields:
```

All stages have same interface

`<stage>_<funcname>(X, y, opt)`

All blocks have same interface

<stage>_<funcname>(X, y, opt)

```
function [kernel] = kernel_rbf(X, y, opt)
```

```
% kernel_rbf(opt)
% Computes the kernel matrix for a Gaussian kernel.
% INPUTS:
% -OPT: struct with the
```

```
function [p] = perf_precrec(X,y, opt)
```

```
% perf_precrec(opt)
% Computes the average precision per class
%
% INPUTS:
% -OPT: structure of options with the following fields:
% -y: labels matrix
% fields that need to be set through previous gurls tasks:
% -pred (set by the pred_* routines)
%
% OUTPUT: struct with the following fields:
% -acc: array of prediction accuracy for each class
% -forho: ""
% -forplot: ""
```

```
% Code adapted from PASCAL VOC 2007 code
```

```
if isstruct(opt, pred)
```

```
function [cfr] = rls_primal (X, y, opt)
```

```
% rls_primal(X,y,opt)
% computes a classifier for the primal formulation of RLS.
% The regularization parameter is set to the one found in opt.params
% In case of multiclass problems, the regularizers need to be computed
singlelambda function.
```

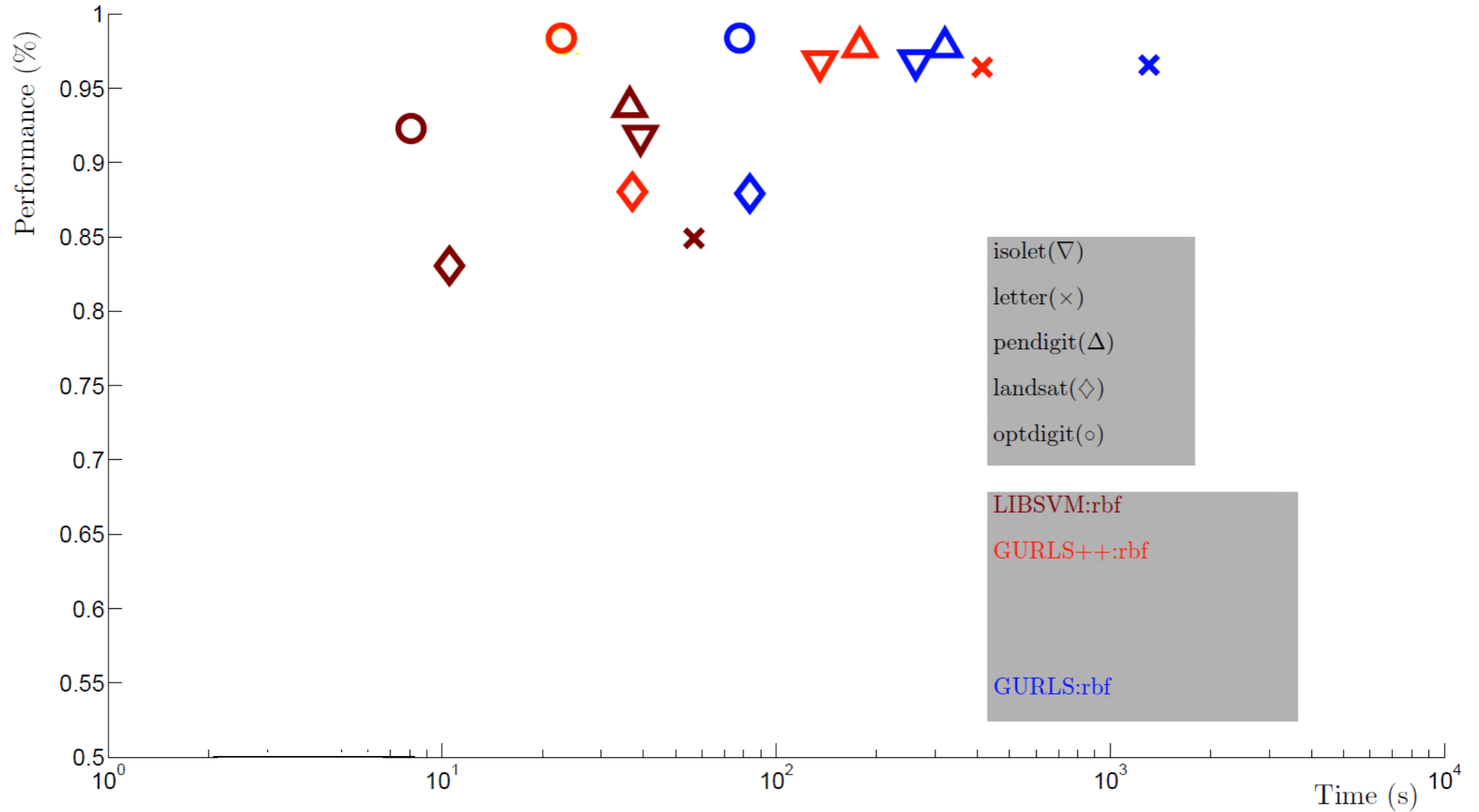
```
% INPUTS:
% -OPT: struct of options with the following fields:
% fields that need to be set through previous gurls tasks:
% - paramsel.lambdas (set by the paramsel_* routines)
% fields with default values set through the defopt function:
% - singlelambda
```

```
% For more information on standard OPT fields
% see also defopt
```

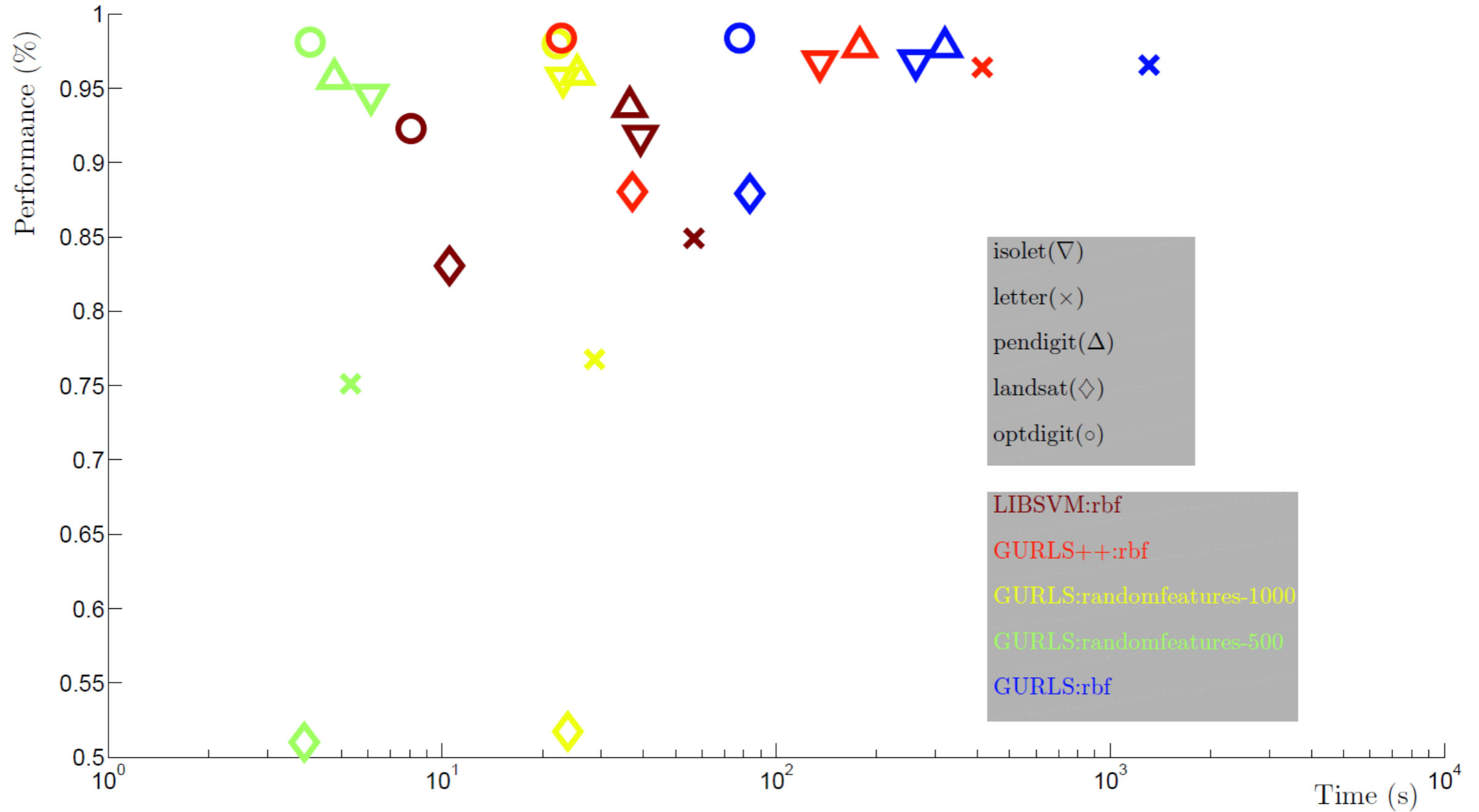
```
% OUTPUT: struct with the following fields:
```

```
% -W: matrix of coefficient vectors of rls estimator for each class
% -C: empty matrix
```

Results - Classification Benchmarks



Results - Classification Benchmarks



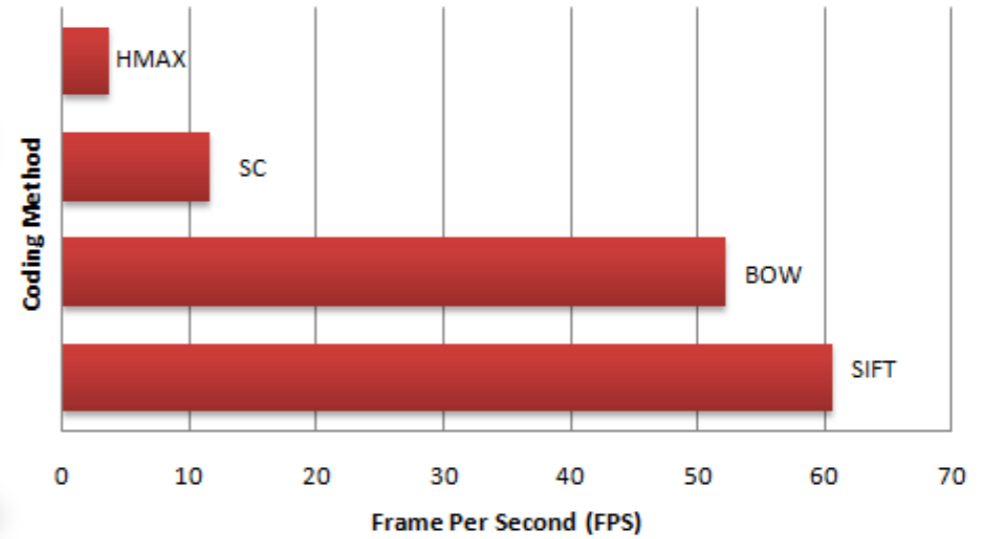
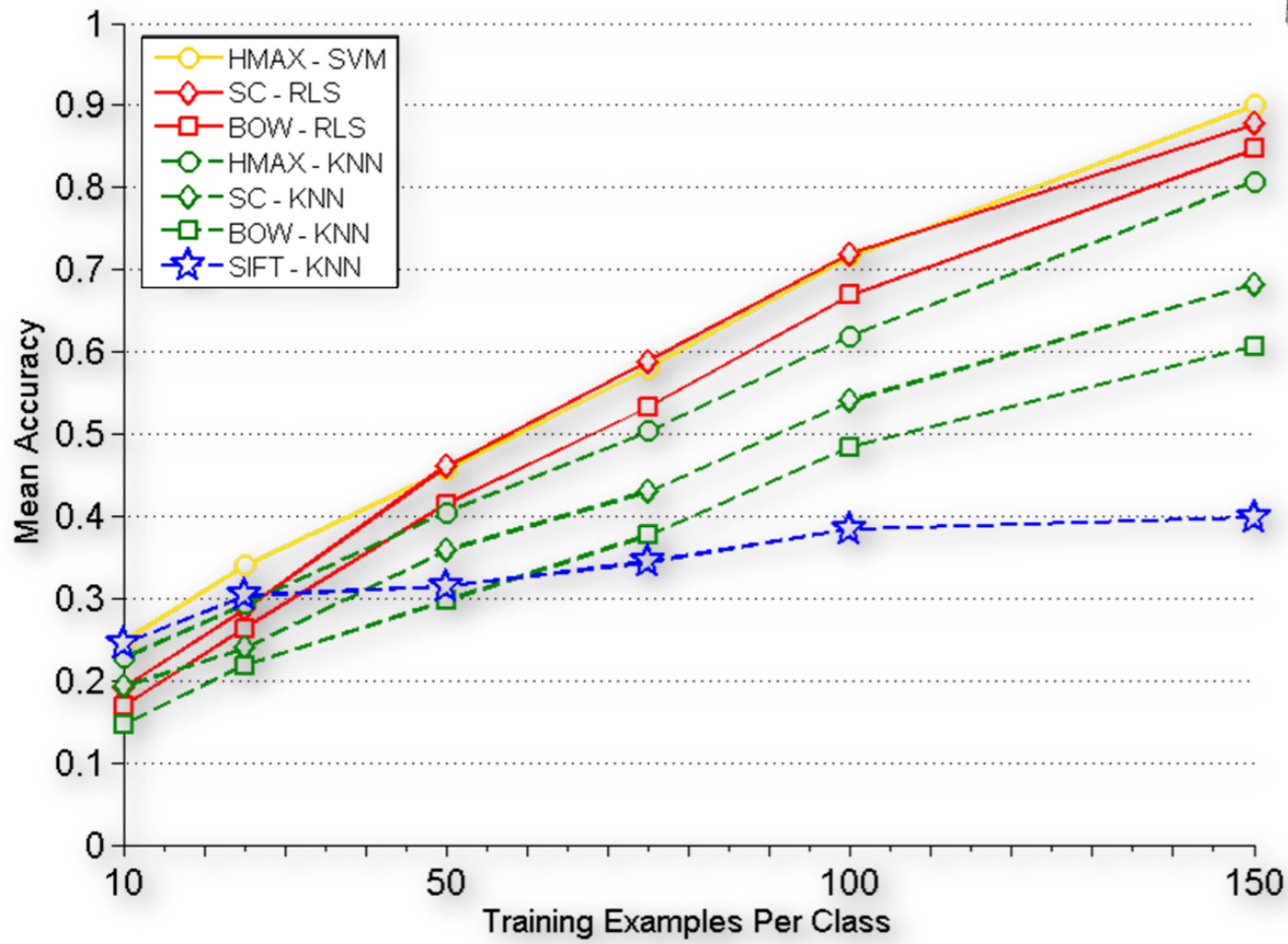
Results - Comparison with the state of the art

PubFig83

<http://www.eecs.harvard.edu/zak/pubfig83/>

<i>Package</i>	<i>Kernel</i>	<i>Accuracy</i>	<i>Time</i>
GURLS	Linear	87%	0h13m
LIBSVM	Linear	76%	5h20m
GURLS	RBF with selection	88%	5h51m
GURLS	RBF = 25th PCT of DST	87%	0h14m
LIBSVM	RBF = 25th PCT of DST	76%	4h18m

Results - Object Recognition in Robotics



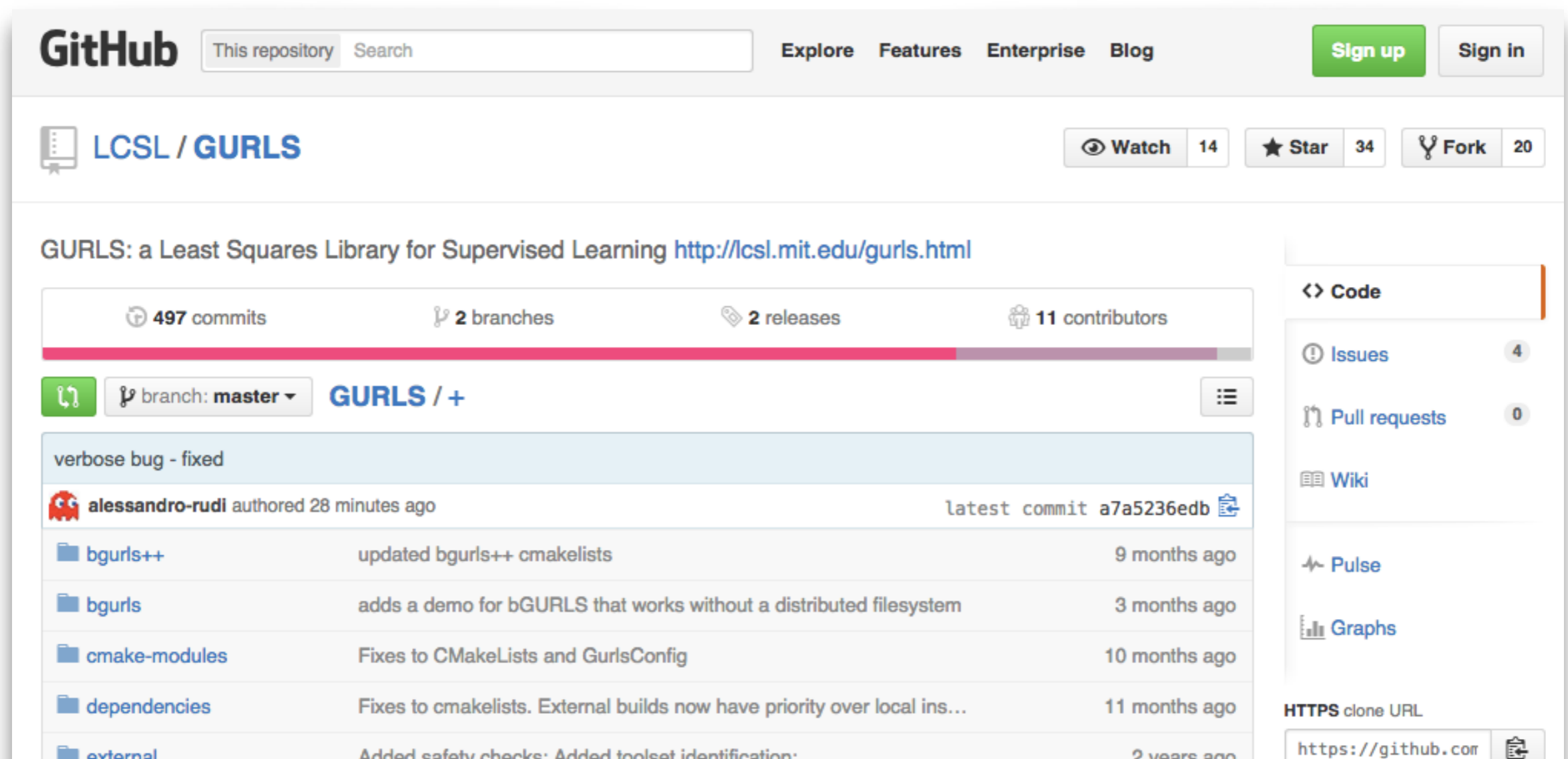
Download or Pull from Github: <https://github.com/LCSL/GURLS>

Reference Paper:

GURLS: a Least Squares Library for Supervised Learning.

A. Tacchetti, P. K. Mallapragada, M. Santoro and L. Rosasco

Journal of Machine Learning Research



The screenshot shows the GitHub repository page for **LCSL / GURLS**. The repository is titled "GURLS: a Least Squares Library for Supervised Learning" with a link to <http://lcs1.mit.edu/gurls.html>. The repository statistics are: 497 commits, 2 branches, 2 releases, and 11 contributors. The current branch is **master**. The repository is watched by 14 people, has 34 stars, and 20 forks. The commit history shows a recent commit by **alessandro-rudi** titled "verbose bug - fixed" 28 minutes ago. The commit message is "latest commit a7a5236edb". The repository structure includes folders: **bgurls++** (updated bgurls++ cmakeLists, 9 months ago), **bgurls** (adds a demo for bGURLS that works without a distributed filesystem, 3 months ago), **cmake-modules** (Fixes to CMakeLists and GurlsConfig, 10 months ago), **dependencies** (Fixes to cmakeLists. External builds now have priority over local ins..., 11 months ago), and **external** (Added safety checks: Added toolset identification, 2 years ago).

Folder	Description	Time
bgurls++	updated bgurls++ cmakeLists	9 months ago
bgurls	adds a demo for bGURLS that works without a distributed filesystem	3 months ago
cmake-modules	Fixes to CMakeLists and GurlsConfig	10 months ago
dependencies	Fixes to cmakeLists. External builds now have priority over local ins...	11 months ago
external	Added safety checks: Added toolset identification:	2 years ago

Right sidebar options: **Code**, **Issues** (4), **Pull requests** (0), **Wiki**, **Pulse**, **Graphs**. **HTTPS clone URL**: <https://github.com>

This afternoon at **2PM**:
a Lab to get acquainted with GURLS

+

A small **machine learning challenge**
(just for fun)