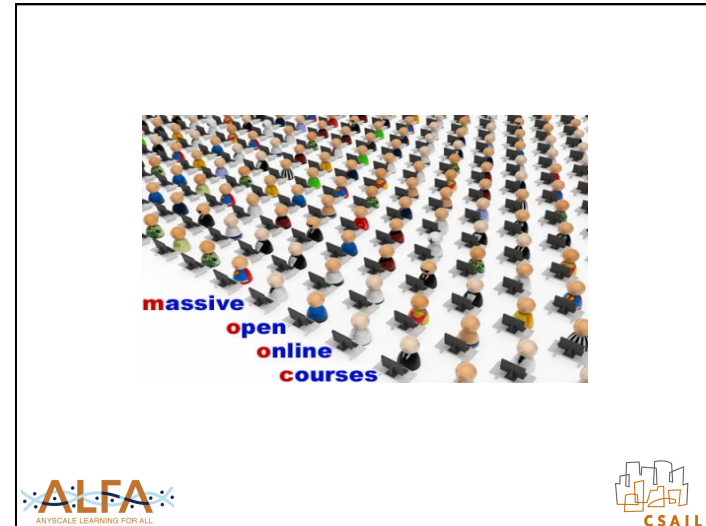
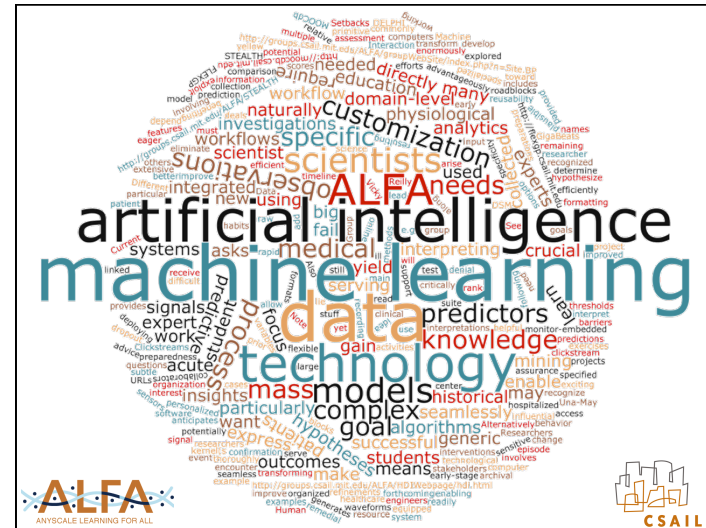


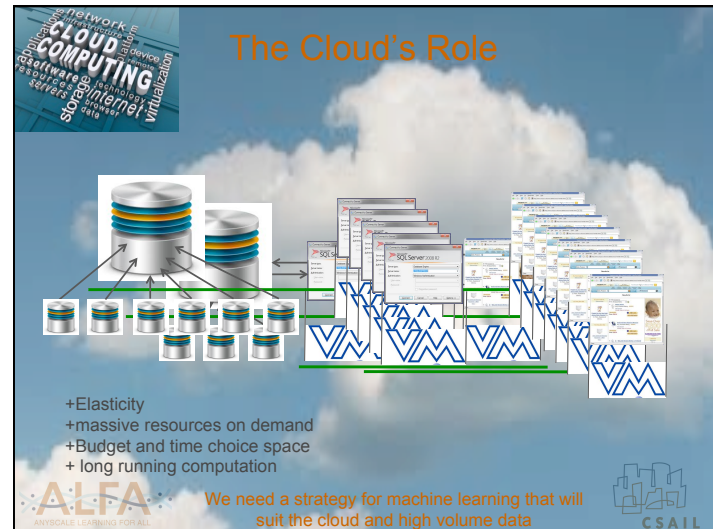
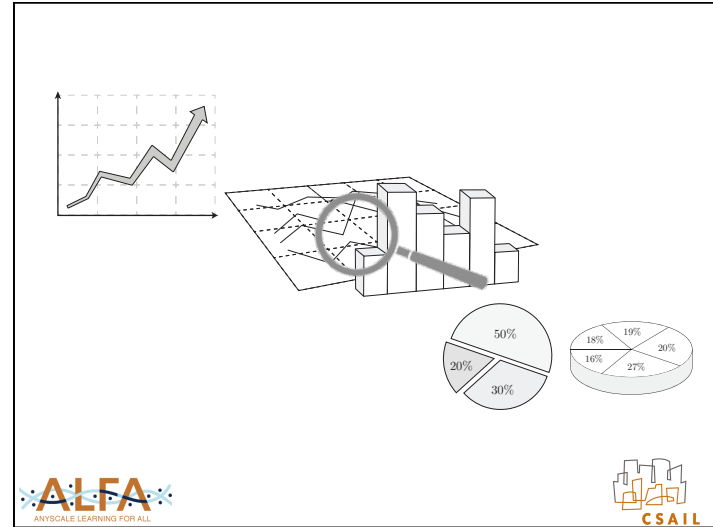
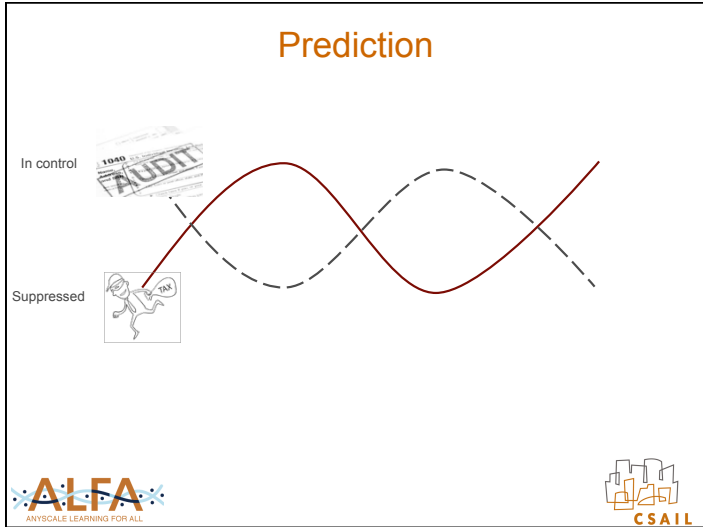
# Data-Driven Artificial Intelligence with Machine Learning

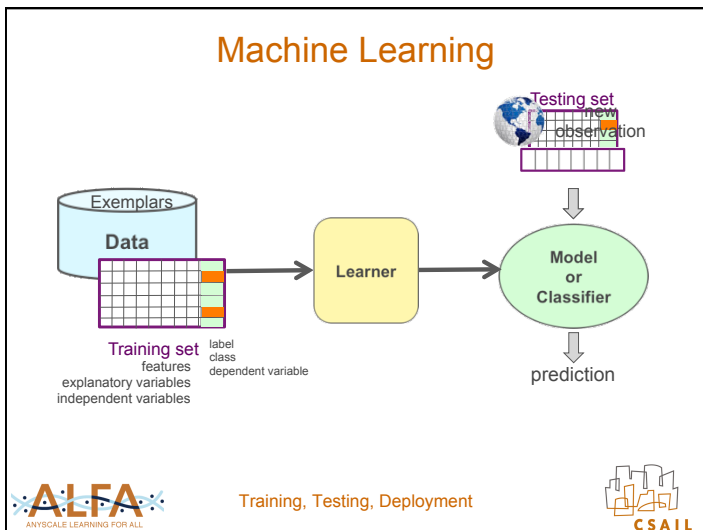
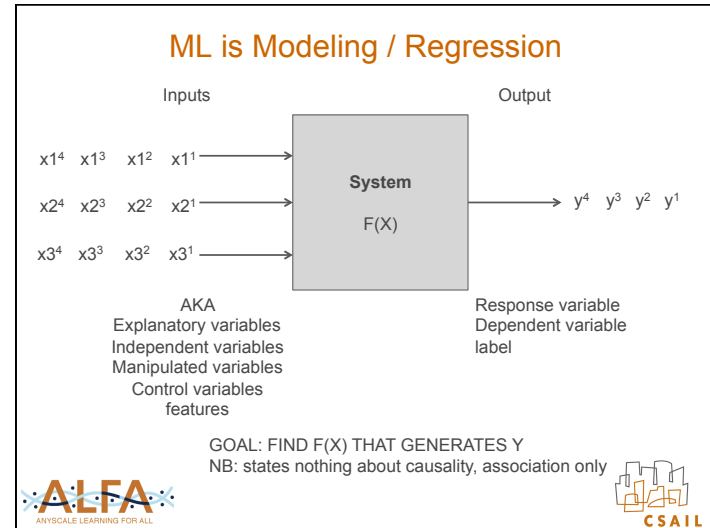
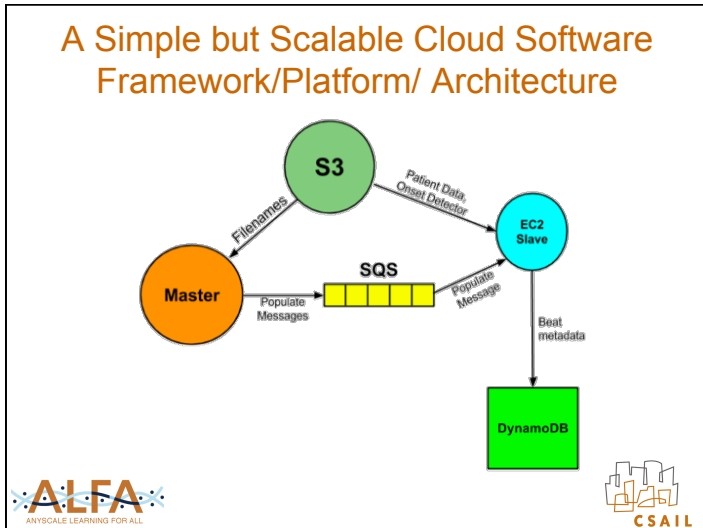
Una-May O'Reilly unamay@csail.mit.edu  
Leader: ANYSCALE LEARNING FOR ALL Group  
MIT Computer Science and Artificial Intelligence Lab

NASA Workshop on  
Application of Machine Learning Technologies  
for  
Scientific and Engineering Domains

Aug 17, 2016



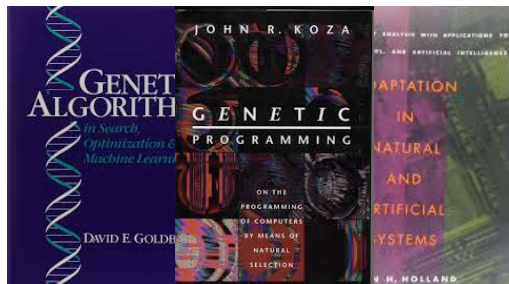
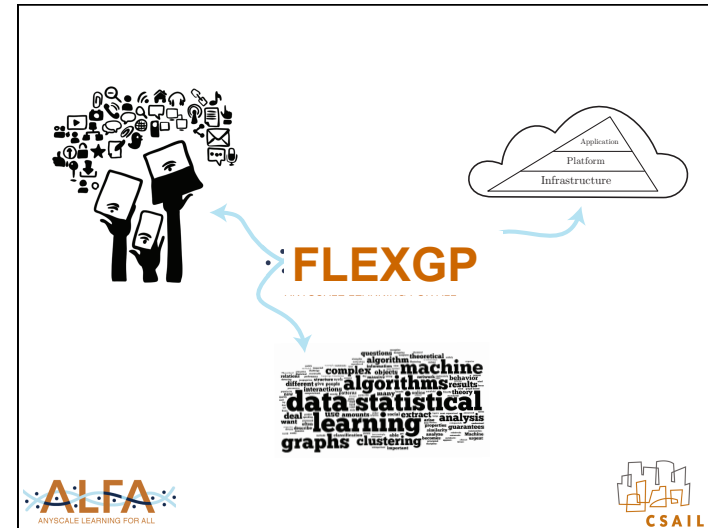




- ### Commonly Used ML Algorithms
- **Generalized Linear Models**
    - Ordinary Least Squares – Linear Regression
    - Ridge Regression: imposes a penalty on the size of coefficients
    - Lasso Regression: estimates sparse coefficients
    - Elastic Net
    - Stochastic gradient descent
    - Logistic Regression
      - » linear model for classification (vs regression)
      - » aka logit regression, maximum-entropy classification (MaxEnt) or the log-linear classifier
    - Polynomial regression: inear models trained on nonlinear functions of the data
  - **Non-Linear Models**
    - Support Vector Machines
    - Naïve Bayes
    - Decision Tree, Random Forest
    - Gaussian Processes
    - Neural Network
      - » deep learning
    - Genetic programming
- ALFA**  
ANYSCALE LEARNING FOR ALL
- Supervised Learning Algorithms
- CSAIL**

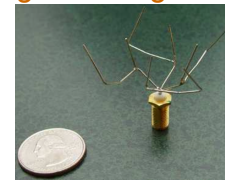
## ML Algorithm Competence Metrics

Method	Speed	Accuracy	Readable
GOAL			
LINEAR METHODS	✓	✓	✓
NON-LINEAR Methods		✓ ✓	

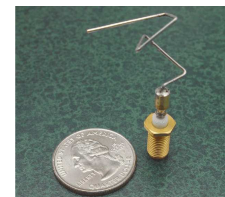


## Example of Genetic Programming for Design

- Evolving an antenna for NASA's Space Technology 5 mission
  - <https://ti.arc.nasa.gov/profile/hornby/>
  - » Computer-Automated Evolution of an X-Band Antenna for NASA's Space Technology 5 Mission. [Evolutionary Computation 19\(1\): 1-23 \(2011\)](#)





Computer Evolution Antenna Technology [Evolutionary Computation 19\(1\): 1-23 \(2011\)](#)







### Example of Genetic Programming for Modeling

- **Nonlinear Dynamical Systems Identification**
  - J. Bongard and H. Lipson, "Automated reverse engineering of nonlinear dynamical systems," Proceedings of the National Academy of Sciences, vol. 104, no. 24, pp. 9943–9948, 2007.
  - M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," Science, vol. 324, no. 5923, pp. 81–85, 2009.
  - Inferring biological networks by sparse identification of nonlinear dynamics, Niall M. Mangan, Steven L. Brunton, Joshua L. Proctor, J. Nathan Kutz, arXiv:1605.08368

### Example of Genetic Programming for Big Data ML



- *Data-driven methods in fluid dynamics: Sparse classification from experimental data*, in Whither Turbulence and Big Data in the 21st Century, A. Pollard et al. Eds. 281-301 (Springer 2016) [Bai, Brunton, Brunton, Kutz, Kaiser, Spohn, Noack].

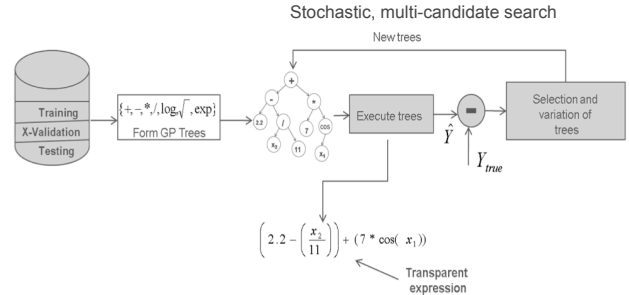


### How Genetic Programming Works

- **Goal: FIND F(X) THAT GENERATES Y**
  - Generalized Linear Modeling (GLM) optimizes model structural parameters/coefficients using (X,Y) examples
$$\hat{y}(w, x) = w_0 + w_1x_1 + \dots + w_px_p$$

$$\min ||Xw - y||_2^2$$
- **GP composes candidate models as program expression**
  - › eg:  $x_1 + x_2 * x_3 + (x_2 + x_4 / x_5)$ 
    - coefficients/parms tuned after model composition or within it
  - › composition uses arithmetic operators
    - +, -, \*, protected divide
    - square, square root, log, exponent, cos, sin etc...
  - › tuning through GLM or other means







### Genetic Programming

## FlexGP



- Introduction -> open source project
- Scaling -> bigger
  - FlexGP system
  - FCUBE
- Learners -> improving their ML competence



### Design Drivers

- As always
  - accuracy
  - interpretability
  - speed
- Design for **scalability**
- **IDENTIFY** and **BUILD** a niche for GP-ML
  - it's not NN, SVM, or generalized linear modeling
- Ease of design, use
- Reflective of the cloud's "new" way of supporting our work



Ignacio Arnaldo



Una-May O'Reilly



Kalyan Veeramachaneni



Owen Derby



Krzysztof Krawiec




**FLEXGP Project**

**FLEXGP (System)**

Scaling Prototype

**FUBE (Platform)**


Scaling for All


**GP LEARNERS**

Regression  
Classification


**HYBRID LEARNERS**

MRGP  
EFS  
Behavioral GP





李嘉誠基金會  
LI KA SHING FOUNDATION



The FlexGP Project

Home People Publications Source Workshop (2014)

**FlexGP**

**CSAIL**

### The FlexGP Project

In a nutshell, the FlexGP project goal is scalable machine learning using genetic programming (GP).

Genetic programming is a mature, robust multi-point search technique (inspired by evolution) which supports readable, and flexibly specified learning representations which can readily express linear or non-linear data relationships. It is well suited to parallelization and machine learning. It has a strong record in real world domains.

- **Evolutionary learners:** this layer provides access to the learners so that one could run them on their desktop. See description of the learners [here](#) and a tutorial to running them on multiple examples [here](#)
- **FlexGP:** a cloud based platform for generating transparent non-linear large scale regression problems
- **FCUBE:** A data parallel approach to building ensemble of classifiers
- **Feature learning:** Evolutionary Feature Synthesis (EFS) generates accurate, readable, nonlinear features for tabular data.

**ALFA**  
ANYSCALE LEARNING FOR ALL

**CSAIL**

The FlexGP Project

FCUBE GP-based learners EFS Examples

flexgp.github.io/gp-learners/

## GP-based Learners

We provide a tutorial for the core learners developed within the frame of the FlexGP project.

### Regression

1. Multiple Regression Genetic Programming
2. SR learner: Symbolic Regression

### Classification

1. GP function classification
2. Rule Tree classification

### Examples

To check examples and reports visit our blog: [FlexGP Blog](#)

**ALFA**  
ANYSCALE LEARNING FOR ALL

**CSAIL**

The FlexGP Project

FCUBE GP-based learners EFS Examples

flexgp.github.io/gp-learners/blog.html

## FlexGP Blog

In this blog we will provide examples of use of the learners developed within the FlexGP project. We will analyze the performance of the released learners for different datasets.

1. SR learner
2. Rule Tree classification
3. GP function classification
4. Multiple Regression Genetic Programming

### Symbolic Regression Learner: Predicting the quality of wine

The Wine Quality dataset is available at the [UCI Machine Learning repository website](#). This problem consists in modeling the quality (a grade from 1 to 10) of a given red or white wine given 11 features such as acidity, alcohol degree etc. Note that the first line of both datasets contains the labels of the different features and needs to be deleted. Additionally, the separators employed in the original dataset

Hosted on [GitHub Pages](#) using the [Dinky theme](#)

**ALFA**  
ANYSCALE LEARNING FOR ALL

**CSAIL**

flexgp - GitHub

GitHub, Inc. (US) https://github.com/flexgp

Personal Open source Business Explore Pricing Blog Support This organization

**flexgp**

Repositories All People

Filters Find a repository...

<b>efs</b>	Evolutionary feature synthesis	Java	★ 3	3/2
<b>gp-learners</b>	GP-based learners	Java	★ 0	3/1
<b>FCUBE</b>	FCUBE: a platform for collaborative learning	Java	★ 3	3/1
<b>flexgp</b>	FlexGP: Flexible ML with Genetic Programming	Java	★ 4	3/1

**ALFA**  
ANYSCALE

**CSAIL**

The FlexGP Project Home People Publications Source Workshop (2014)

- **Building predictive models via feature synthesis.** Ignacio Arnaudo, Kaiyan Veeramachaneni, Una-May O'Reilly, GECCO '15, pp. TBD.
- **Bring Your Own Learner!** A cloud-based, data-parallel commons for machine learning. Ignacio Arnaudo, Kaiyan Veeramachaneni, Andrew Song, Una-May O'Reilly. To appear in IEEE Computational Intelligence Magazine: Special Issue on Computational Intelligence for Cloud Computing (Feb. 2015).
- **FlexGP:** Cloud-Based Ensemble Learning with Genetic Programming for Large Regression Problems. Kaiyan Veeramachaneni, Ignacio Arnaudo, Owen Derby, Una-May O'Reilly. Journal Of Grid Computing, Nov 2014.
- **Flash:** A GP-GPU Ensemble Learning System for handling Large Datasets. Ignacio Arnaudo, Kaiyan Veeramachaneni and Una-May O'Reilly, 17th European Conference on Genetic Programming, Springer LNCS 8569, pp 13-24.
- **Multiple regression genetic programming.** Ignacio Arnaudo, Krzysztof Krawiec, Una-May O'Reilly, GECCO '14, pp 879-886. Available free from the ACM Digital Library.
- **Cloud Scale Distributed Evolutionary Strategies for High Dimensional Problems.** Dennis Wilson, Kaiyan Veeramachaneni, and Una-May O'Reilly, EVOPAR track, Applications of Evolutionary Computation, Lecture Notes in Computer Science Volume 7835, 2013, pp 519-528.
- **Cloud Driven Design of a Distributed Genetic Programming Platform.** Owen Derby, Kaiyan Veeramachaneni, and Una-May O'Reilly, EVOPAR track, Applications of Evolutionary Computation, Lecture Notes in Computer Science Volume 7835, 2013, pp 509-518.
- **Learning regression ensembles with genetic programming at scale.** Kaiyan Veeramachaneni, Owen Derby, Dylan Sherry, Una-May O'Reilly, GECCO '13, Proceeding of the thirteenth annual conference on Genetic and evolutionary computation conference, 2013
- **Building MultiClass Nonlinear Classifiers with GPUs.** Ignacio Arnaudo, Kaiyan Veeramachaneni and Una-May O'Reilly, 2014 NIPS Workshop on Big Learning, Workshop papers.
- D. Sherry, K. Veeramachaneni, J. McDermott, and U.M. O'Reilly. **Flex-GP: Genetic programming on the cloud.** In Evolutionary Applications 2012, LNCS, Vol. 7248, pp. 477-486, Springer-Verlag, 11-13 April 2012. (It was awarded Best Paper)

ALFA ANYSCALE LEARNING FOR ALL CSAIL

FLEXGP Project

Scaling

FLEXGP (System) FUSE (Platform) GENETIC LEARNER HYBRID LEARNERS

Regression Classification MRGP EFS

WORK In Progress

ALFA ANYSCALE LEARNING FOR ALL GE 李嘉誠基金會 LI KA SHING FOUNDATION CSAIL

A data parallel system for large scale machine learning

Yields hundreds to thousands of models per run

ALFA ANYSCALE LEARNING FOR ALL <http://flexgp.github.io> CSAIL

F<sup>3</sup>: Handling Big Data

F<sup>3</sup> = Factor the data across learners replicated on the cloud  
Filter the resulting models  
Fuse the filtered models into a Metal-Model

ALFA ANYSCALE LEARNING FOR ALL CSAIL

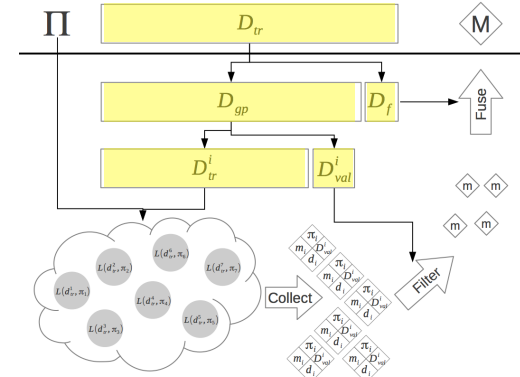
## Parameter and Data Factoring

- $\{\Pi, D_{tr}^i\}$
- Distribution over each of these parameters probabilistically biases how each instance chooses a value for that parameter when it starts the local GP.
  - Eg.  $\{W, \text{Norm2}, \{d1...3000\}, \{x1, x2\}\}$
  - Eg.  $\{WUX, \text{Norm2}, \{d2...6000\}, \{x1, x4\}\}$ .

Parameter	Value	Definition		
II	Operator Set ( $L$ )	W X Y Z	$\{+, -, /, *\}$ $\{exp, ln\}$ $\{sqrt, x^2, x^3, x^4\}$ $\{sin, cos\}$	
	Objective Function ( $O$ )	Norm Norm-2 Norm-inf	Mean absolute error Mean squared error Max error	
	D	Training Cases ( $D_{tr}^i$ )	$n$	Subset of $D_{gp}$ , of size $n$
		Feature Set ( $F$ )	$m$	Subset of features, of size $m$



## Splitting the Data for ML



## Fusion: Model Combination Methods

- Model Selection
  - Average Model Prediction (AMP)
    - » Average performance of every model in ensemble on  $D_{test}$
  - Best Apriori Model (BAM)
    - » Select best model based on MSE based validation data
- Model Fusion
  - Average Ensemble Prediction (AVE)
    - » Report average of predictions
  - Median Average Model (MAD)
    - » Average of median plus 2 neighbours
  - Adaptive Regression Mixing (ARM)
    - » Yang, Y.: Adaptive regression by mixing. J. Am. Stat. Assoc. 96(454), 574-588 (2001)
- Probabilistic Fusion (impractical)



## Adaptive Regression Mixing

- Concept: report a weighted average of model predictions
    - Weights obtained by using  $D_f$
    - Assumes model errors are normally distributed
    - Uses variance in errors to identify weights
    - Split up  $D_f$  into  $D^{(1)}$  and  $D^{(2)}$  and  $r$  is size of  $D_f$
  - Provides a substitution for  $r$  in case of underflow
- Step 1: Evaluate  $\sigma_m^2$ , which is the maximum likelihood estimate of the variance of the errors,  $\epsilon_m = \{\hat{y}_{m,j} - z_j | x_j, z_j \in D^{(1)}\}$ . Compute the sum of squared errors on  $D^{(2)}$ ,  $\beta_m = \sum_{j=\frac{r}{2}+1}^r (\hat{y}_{m,j} - z_j)^2$ .
- Step 2: Estimate the weights using:
- $$W_m = \frac{(\sigma_m)^{-r/2} \exp(-\sigma_m^{-2} \beta_m / 2)}{\sum_{j=1}^M (\sigma_j)^{-r/2} \exp(-\sigma_j^{-2} \beta_j / 2)} \quad (1)$$
- Step 3: Redraw subsets  $D^{(1)}$  and  $D^{(2)}$  and repeat steps 1 and 2. Continue this process for a fixed number of times<sup>2</sup>. Average the weights to get the final weights for the models.
- Given a test point  $\bar{x}_j$ , predict  $\hat{z}_j$  as the weighted average of model predictions:  $\hat{z}_j = \sum_{m=1}^M W_m \hat{y}_{m,j}$ .



Combination Methods



## NOX and Million Song DataSets

Dataset	$ D_{sp} $	$ D_f $	$ D_t $	Total	$ S_j $	Range of $z$
NOx	4,017	310	900	5,227	18	[0.270 0.654]

	$D_{tr}$	$D_f$	$D_{te}$	Total
Exemplars	362K	51K	102K	515K
Features	70%	10%	20%	100%
use single-desktop	training		testing	-
use FlexGP	training	fusion train	testing	-

Table 3: MSD splits



## Testing the FlexGP Premise

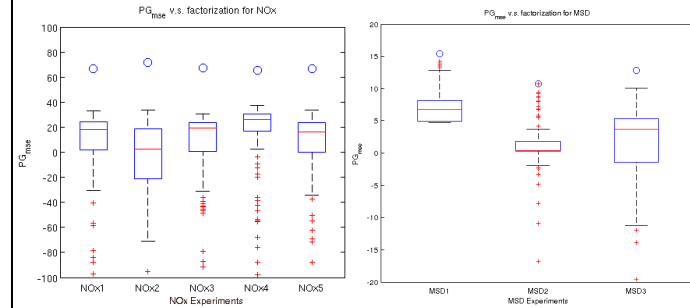


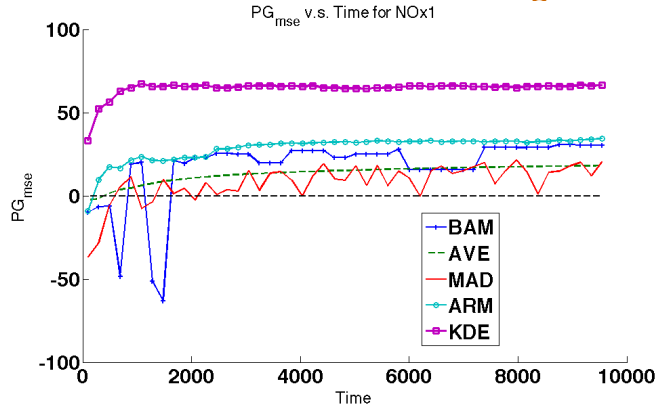
Figure 1: The quartile distribution of  $PG_{mse}$  of models used for fusion in each experiment. The circles represent the best  $PG_{mse}$  from fusion. *Left* Results for NOx experiments; KDE was the best fusion method. *Right* Results for MSD experiments; ARM was the best fusion method.



Results



## FlexGP cloud context: harvesting online



## FlexGP: Your Mileage May Vary

- your mileage may vary, why?
  - your problem's inherent/intrinsic structure and your data as domain's observations
  - your engineering
    - » feature definitions
    - » problem definition
  - your learner



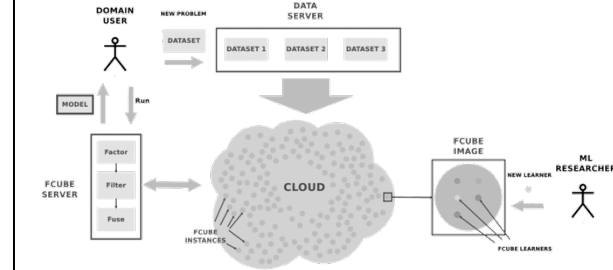


### Moving Forward

- FlexGP indicates how to crisply isolate algorithm from scaling framework
- ALFA and rest-of-EC will always be developing new learners
  - but that requires easy scaling
- We also need to
  - compare multiple learners
- bonus: how to facilitate collaboration with different learners
  - collaboration that might help us strengthen our niche
- for ALFA:
  - to get this boost in power, leave our implementations that hybridized popular design features
    - » eg tree complexity, NSGA2, usual parms
  - start to work on what had been put aside for the F3 phase
- For this we developed FCUBE platform



### FCUBE: simple cloud scaling and algorithm comparison and combination



Our goal is to support and unite developers of interesting classifier algorithms to solve relevant problems of public domain. FCUBE allows to:

- Execute classification algorithms with large training data with a preset computational budget on Amazon EC2
- Retrieve the solutions from the cloud nodes, build a fused model, and compute the testing predictions
- Easily upload datasets
- Easily contribute your standalone classifier in executable format (Java, python) or as source code (must compile in Linux, C, C++ etc)



"Bring your own learner! Opening up cloud-based massively data parallel frameworks" in IEEE Computation Intelligence Magazine, Feb 2015.  
<http://flexgp.github.io/#fcube>

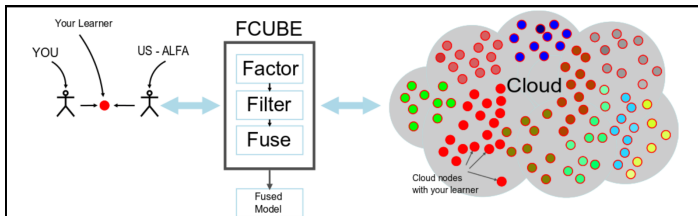


Fig. 4: Collaborative Big Learning Activity taking place within the first edition of the EC for Big Data and Big Learning workshop, GECCO 2014. Participants simply provide stand-alone executables of their learners. The FCUBE team integrates them in the framework, performs all the factor, filter, fuse process and provides performance metrics.

	$D_{tr_1} \dots D_{tr_g}$	$D_f$	$D_{te}$	Total
Exemplars	1,050,000	1,050,000	500,000	11,000,000
Features	28	28	28	28
negative exemplars	47%	47%	47%	47%
positive exemplars	53%	53%	53%	53%
function	training	fusion train	testing	-
accessed by	FCUBE instances	FCUBE Server	-	-

TABLE II: Characteristics of the Higgs dataset and of the generated splits.



### State of Art with GP

- What's "open source" ready from ALFA?
  - MRGP – show competence and central idea
  - EFS – rev 0
- what's underway:
  - EFS ongoing work looking for a client to drive the research's next steps
  - BGP – many objective, better info



### Competency of Regression Algorithms

Method	Speed	Nonlinearities	Fine Tuned Accuracy	Feature Sel.	Readable
GP					
➔ MRGP	?		TARGET		?
NN					
Linear Regression					
LASSO					

COMPETENCY COLOR LEGEND

High	Medium	Low
------	--------	-----

### Multiple Regression Genetic Programming MRGP

Ignacio Araldo  
 Krzysztof Krawiec  
 Una-May O'Reilly

ALFA Group, CSAIL, MIT  
Poznan University of Technology

### MRGP

- improves model accuracy
- how?
  - adds the power of a GLR algorithm to GP
    - » Least Angle Regression (LARS) algorithm
      - Efron et al. The Annals of statistics, 32(2):407-499, 2004.
  - maps model sub-expressions to linear combination with GLR-optimized coefficients

STANDARD GP:  $f(X) = \text{ROOT}(X)$

MRGP:  $f(X) =$

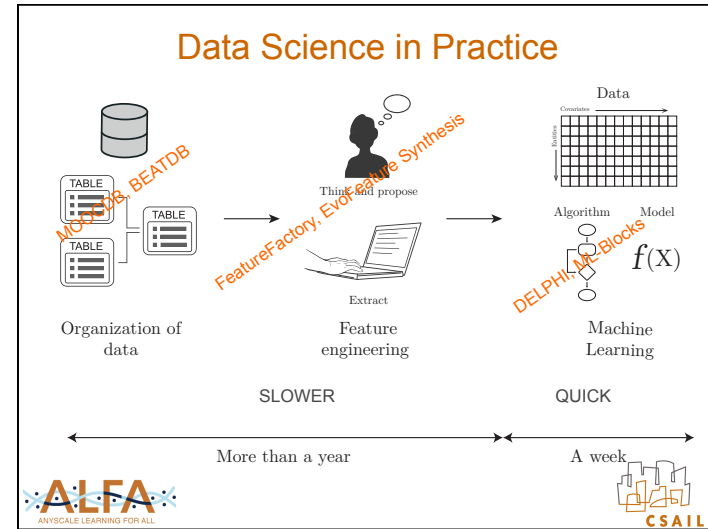
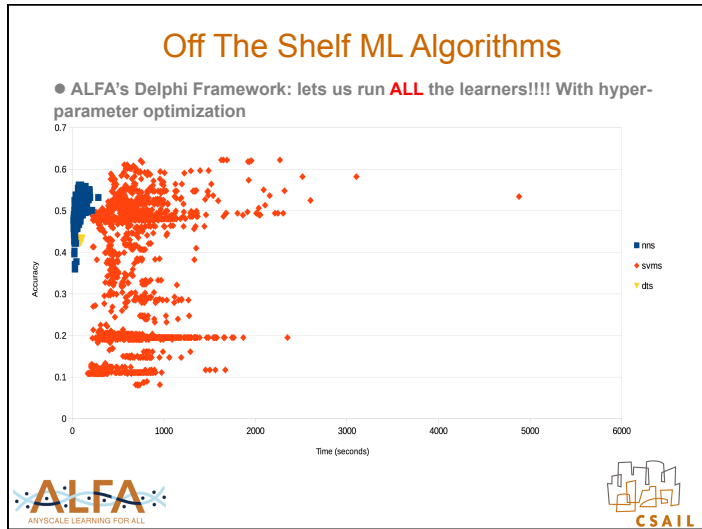
$$\begin{aligned}
 & b_1 X_1 + b_2 X_2 + b_3 X_3 + b_4 X_4 + b_5 X_5 \\
 & + b_6 S_1(X_4) + b_7 S(X_5)^2 + b_8 S_3(X_1, X_5) + b_9 S_4(X_1, X_5) \\
 & + b_{10} \text{ROOT}(X)
 \end{aligned}$$

### MRGP Competency of Regression Algorithms

Method	Speed	Nonlinearities	Fine Tuned Accuracy	Feature Sel.	Readable
GP					
➔ MRGP	✓		TARGET ✓✓		X
NN	CHECK				
Linear Regression					
LASSO					

COMPETENCY COLOR LEGEND

High	Medium	Low
------	--------	-----



### Feature Selection as a Goal

Method	Speed	Nonlinearities	Fine Tuned Accuracy	Feature Sel.	Readable
GP	Low	High	Low	Medium	Low
MRGP	Medium	High	Low	Low	Low
→ EFS	TARGET	High	Low	TARGET	TARGET
NN	Medium	High	Low	Low	Low
Linear Regression	High	Low	High	Medium	High
LASSO	High	Low	High	High	High

COMPETENCY COLOR LEGEND

High Medium Low

**ALFA**  
ANYSCALE LEARNING FOR ALL

CSAIL

### Building predictive models via Feature Synthesis EFS

Ignacio Arnaldo  
Una-May O'Reilly  
Kalyan Veeramachaneni  
ALFA Group, CSAIL, MIT

Project website:  
<http://flexgp.github.io/efs/>

**ALFA**  
ANYSCALE LEARNING FOR ALL

CSAIL

### Tree-based GP vs. EFS

OLD/MRGP	NEW/EFS
Search for <i>models</i>	search for coadapted <i>features</i>
Performance Metric: <i>model error</i>	Performance metric: <i>feature importance</i>
Fitness is independent of the population	Fitness depends on the population
Symbolic representation	No symbolic representation ( <i>faster</i> )



### Feature Selection as a Goal

Method	Speed	Nonlinearities	Weight tuning	Feature Sel.	Readable
GP	Low	High	Low	Medium	Medium
MRGP	Medium	High	High	Medium	Low
→ EFS	High	High	High	TARGET	TARGET
NN	Medium	High	High	Low	Low
Linear Regression	High	Low	High	Medium	High
LASSO	High	Low	High	High	High

COMPETENCY COLOR LEGEND  
High    Medium    Low



### Summary

- Scalable, competent (evolutionary) machine learning and data science
- **Cloud computing** as a resource
- FlexGP system:
  - data parallelism: **factoring**
  - ensemble based modeling through **filtering** and **fusion**
- Improving the Competence of Learners
  - MRGP
    - » **fine tuning model sub-expressions with ML/linear regression**
    - » improved accuracy and time-to-find-solution
  - EFS
    - » **deep learning of features and model**
    - » improved time-to-find-solution and readability



### Reflections and Ongoing Directions

- Filtering and fusion need more work
- FCUBE needs refinements and use cases
- Lots more scope for better learners

