MLbase

www.mlbase.org

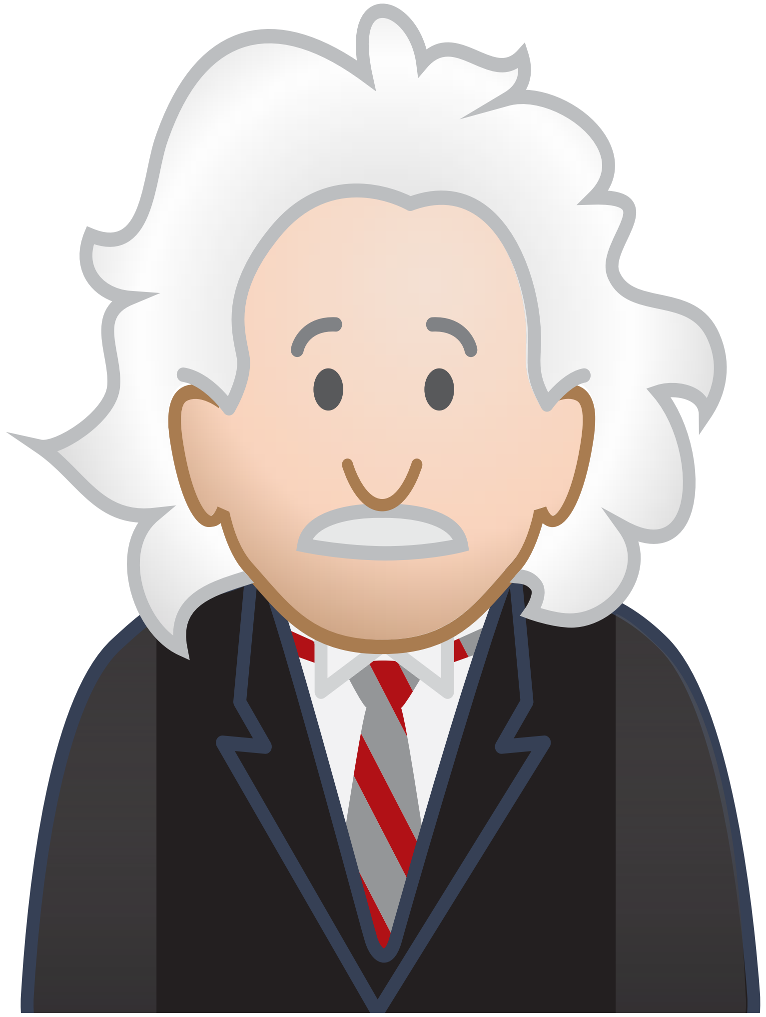
Evan Sparks and Ameet Talwalkar
UC Berkeley

Collaborators: Tim Kraska[2], Virginia Smith[1], Xinghao Pan[1], Shivaram Venkataraman[1], Matei Zaharia[1], Rean Griffith[3], John Duchi[1], Joseph Gonzalez[1], Michael Franklin[1], Michael I. Jordan[1]

[1]UC Berkeley [2]Brown [3]VMware

amplab UC Berkeley

Problem: Scalable implementations *difficult* for ML Developers…



Problem: Scalable implementations *difficult* for ML Developers…

MATLAB
The Language of Technical Computing

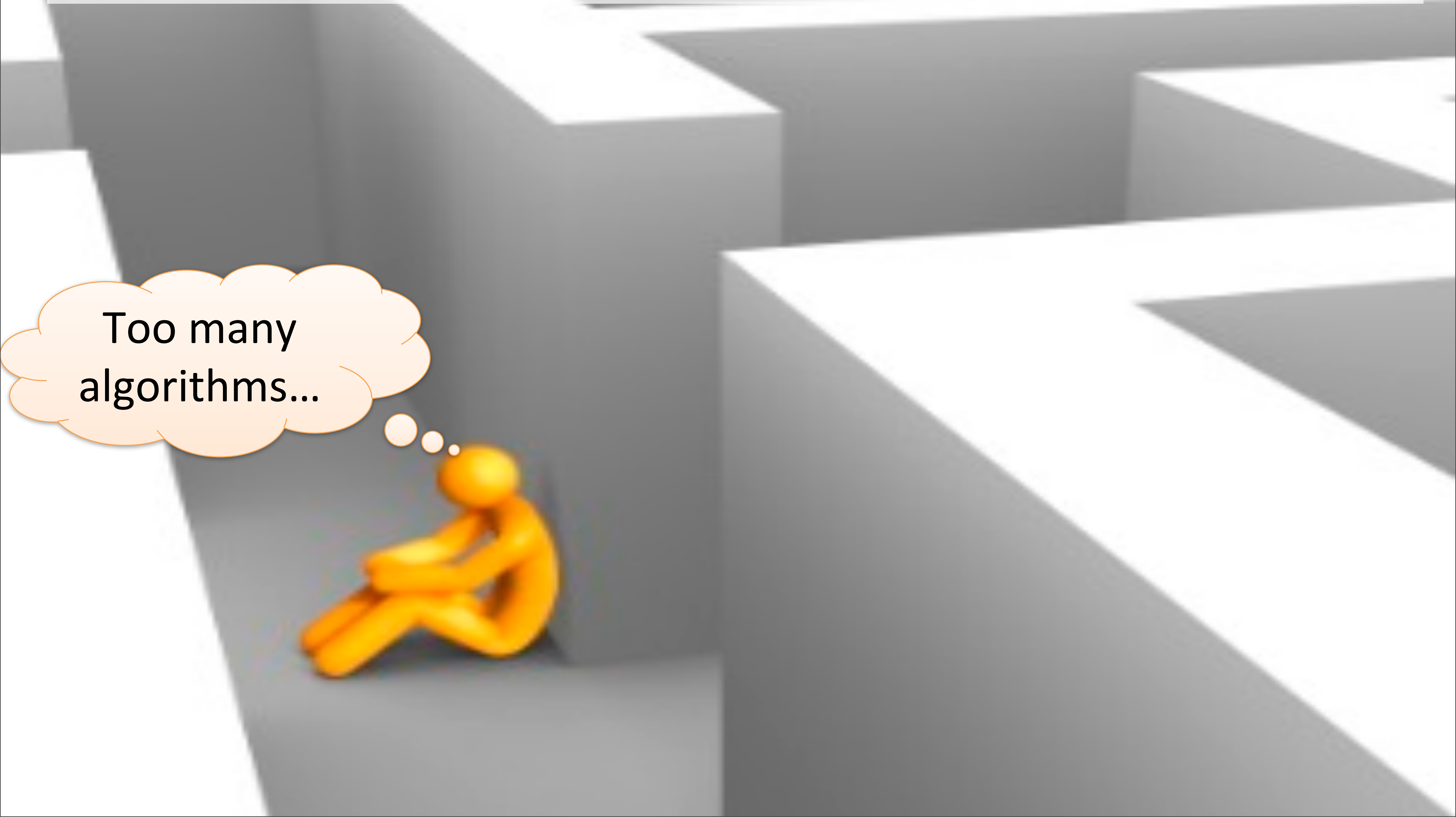
Problem: Scalable implementations *difficult* for ML Developers…

VOWPAL WABBIT

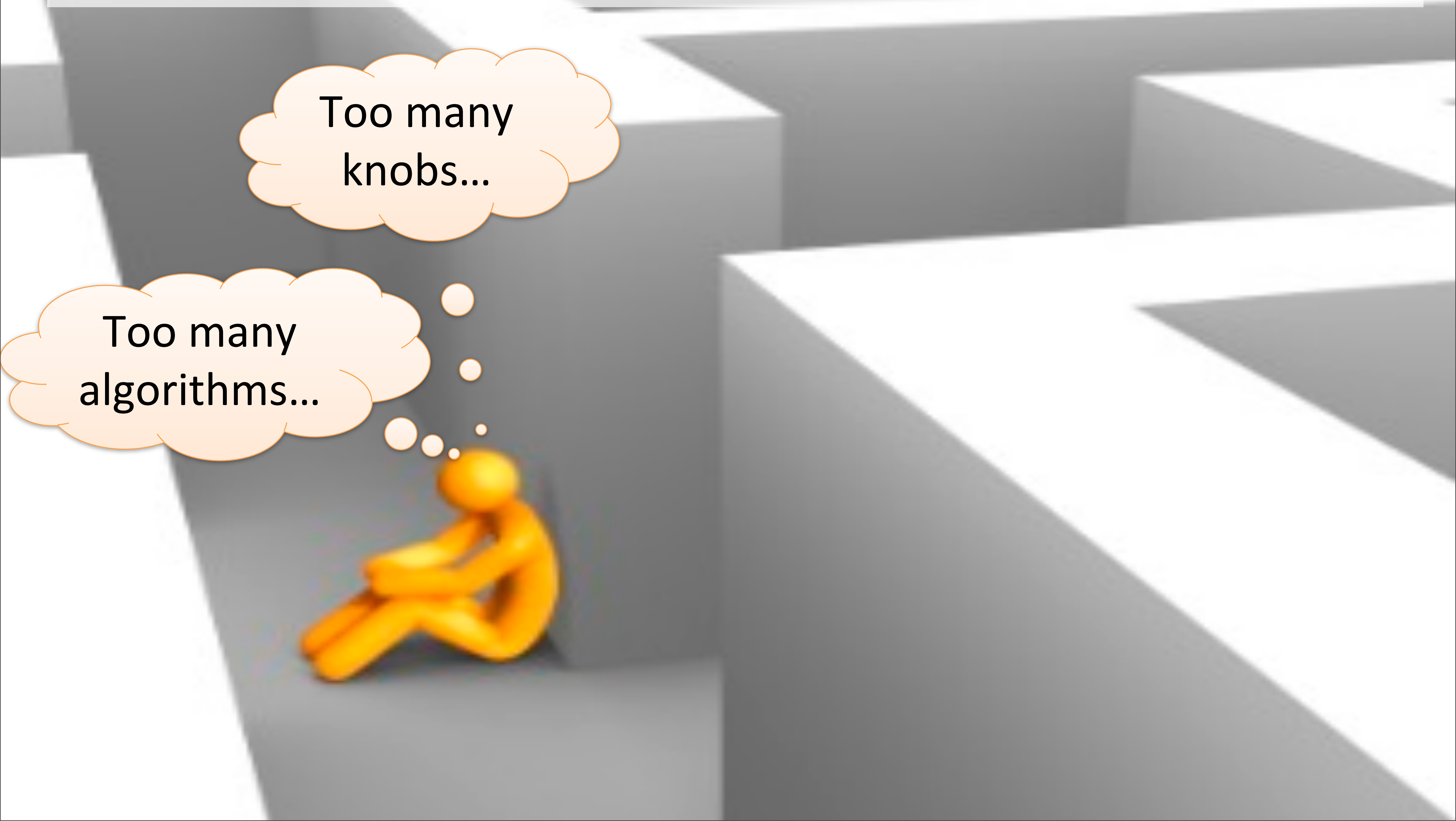
mahout

GraphLab

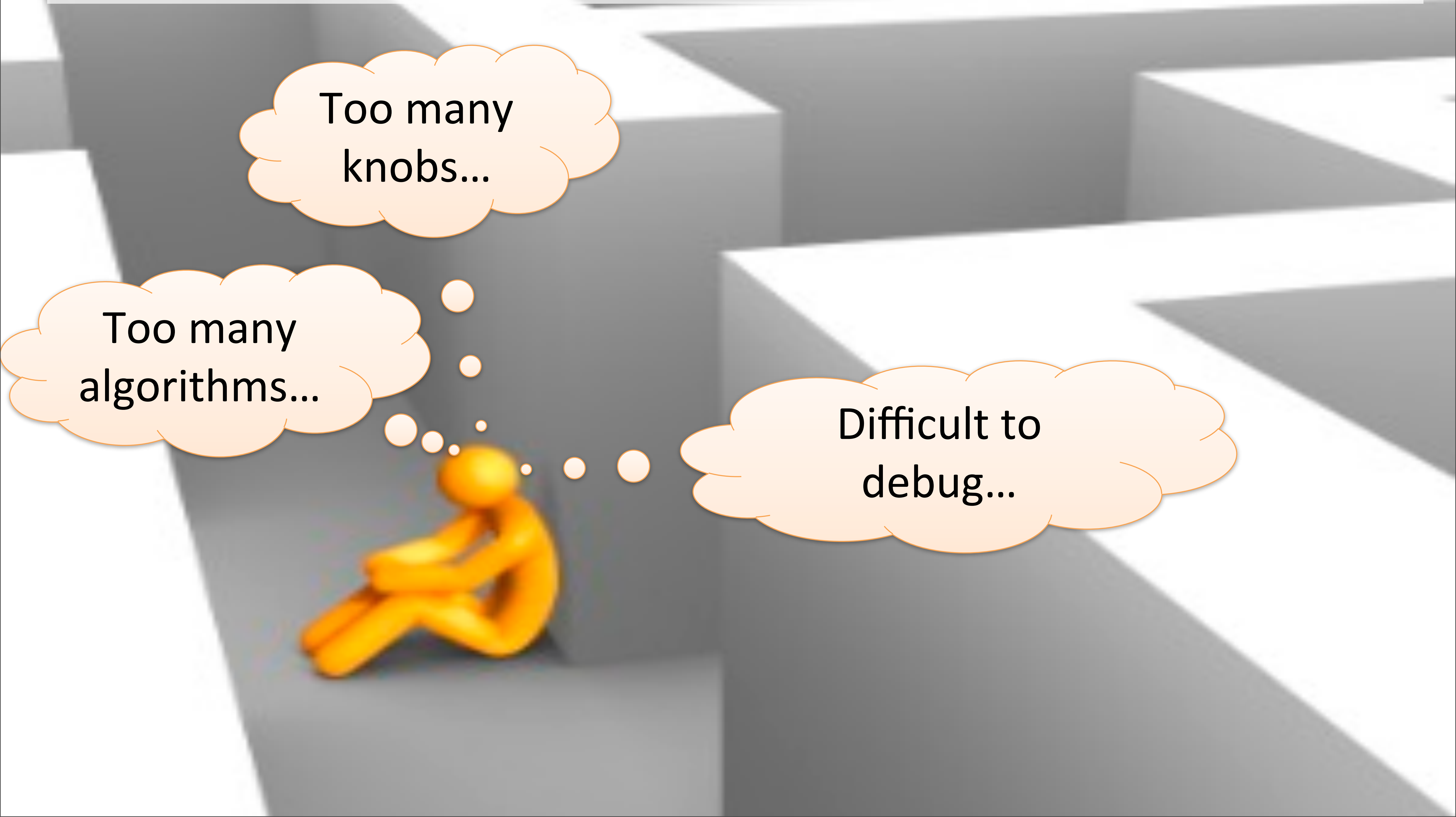
Problem: ML is *difficult* for End Users…



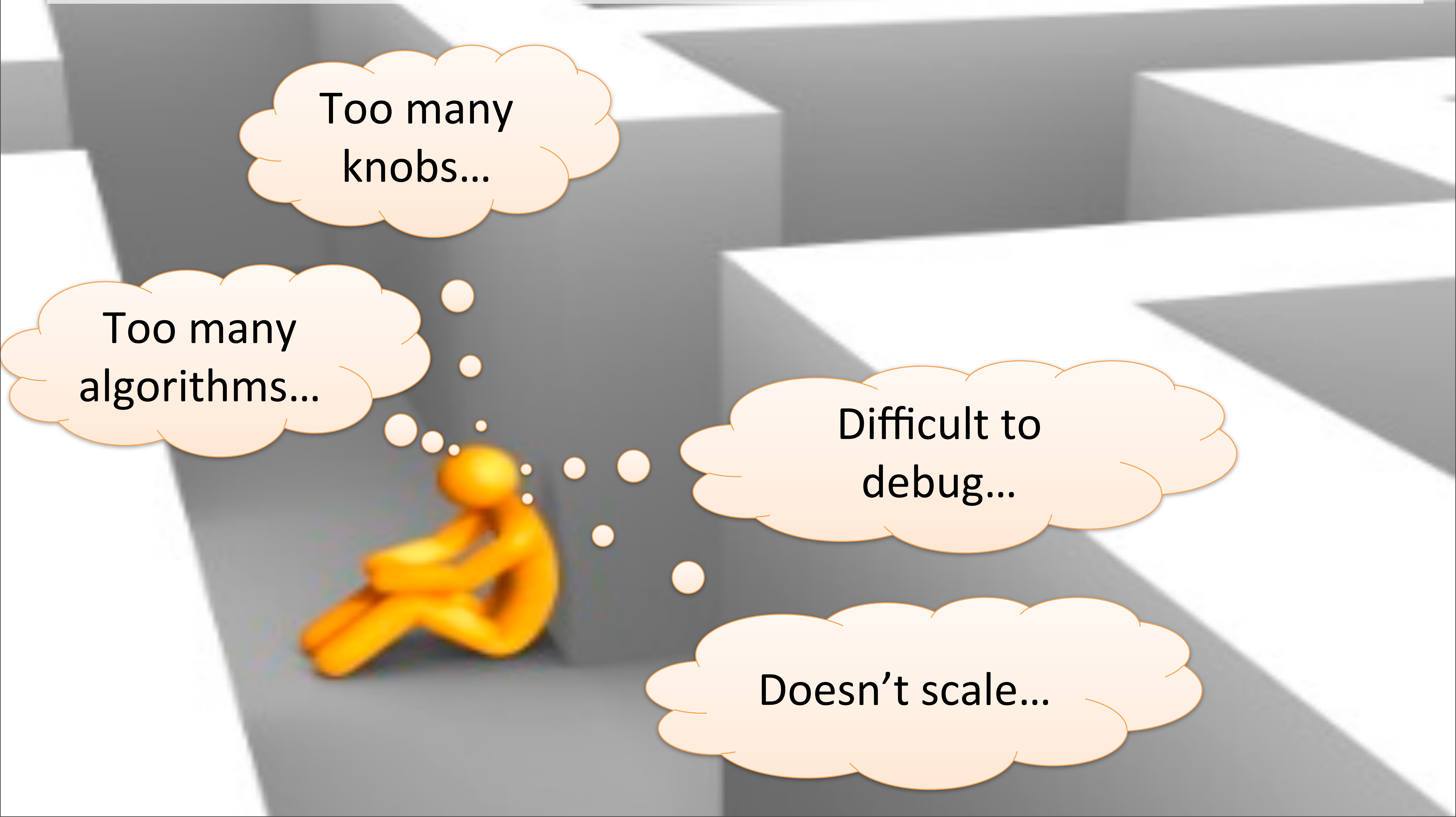
Problem: ML is *difficult* for End Users…



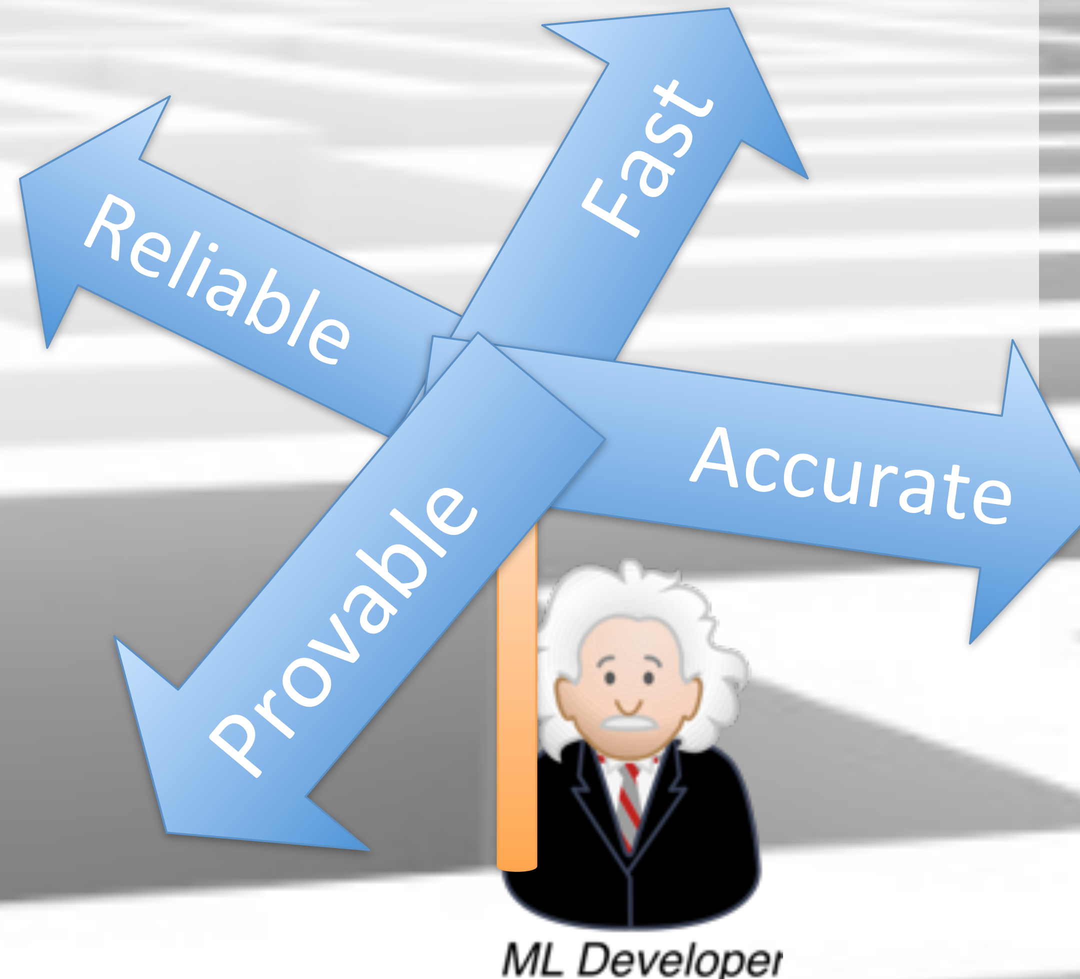
Problem: ML is *difficult* for End Users…



Problem: ML is *difficult* for End Users…



Problem: ML is *difficult* for End Users…



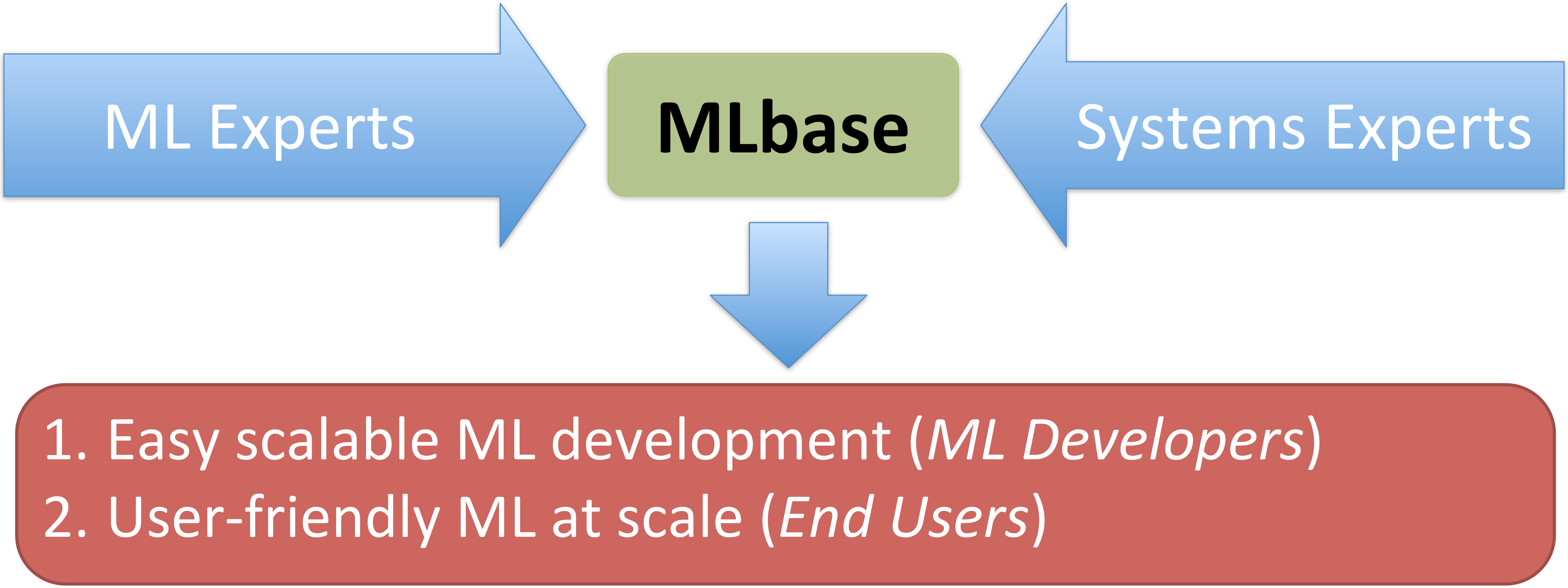
Too many knobs…

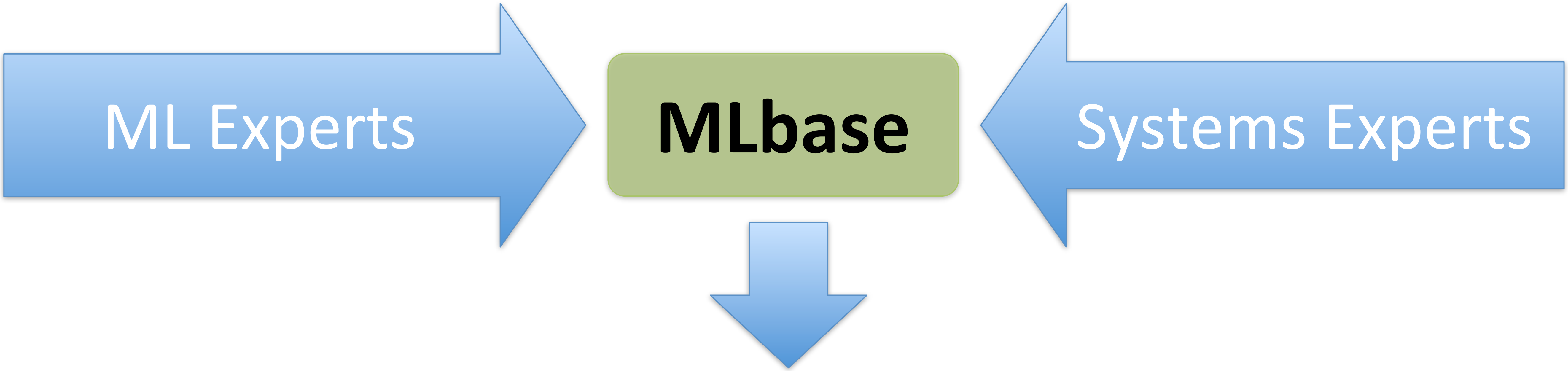
Too many algorithms…

Difficult to debug…

Doesn't scale…


ML Experts
MLbase
Systems Experts


ML Experts
MLbase
Systems Experts
1. Easy scalable ML development (*ML Developers*)
2. User-friendly ML at scale (*End Users*)



1. Easy scalable ML development (*ML Developers*)
2. User-friendly ML at scale (*End Users*)

Along the way, we gain insight into data intensive computing

Vision

MLI Details

Current Status

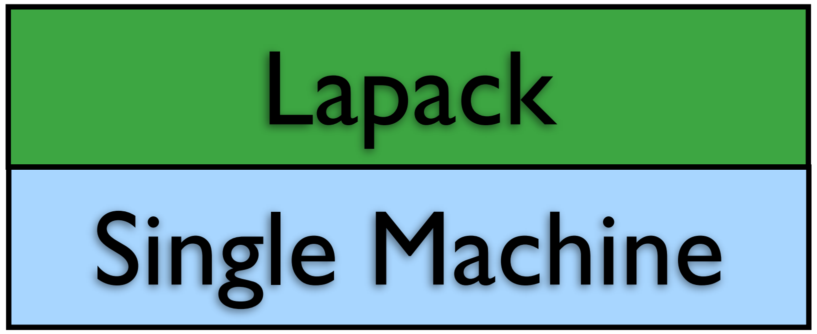
ML Workflow

Matlab Stack

Matlab Stack

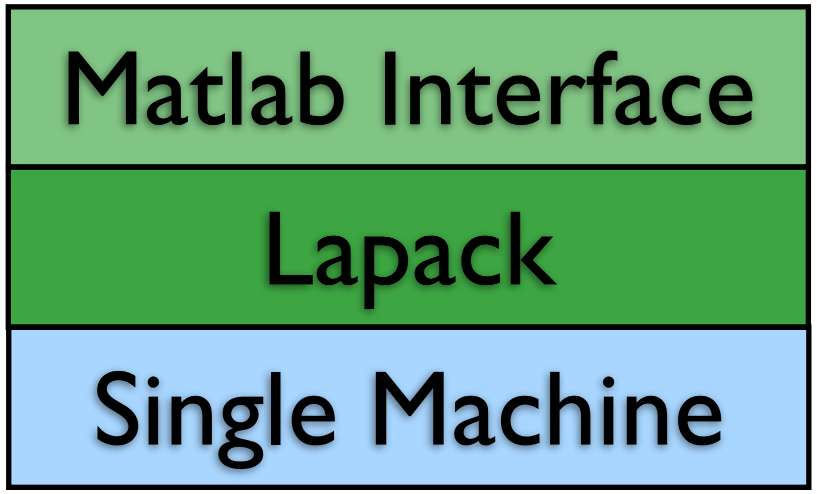
Single Machine

Matlab Stack



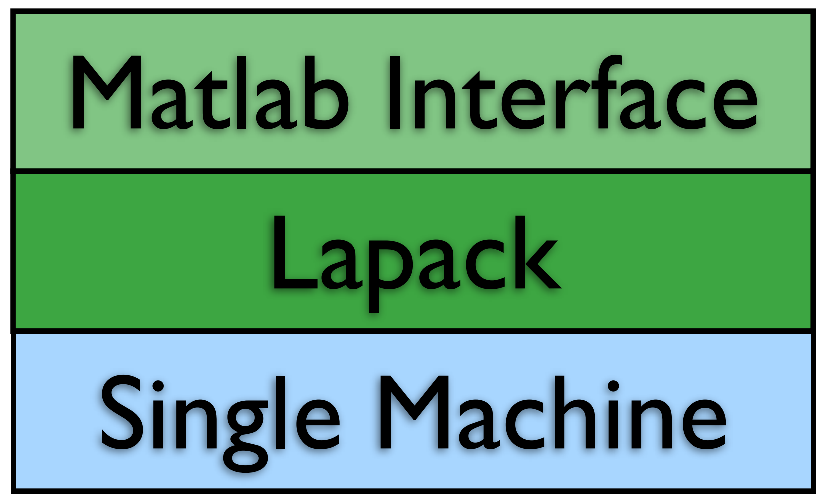
- Lapack: low-level Fortran linear algebra library

Matlab Stack



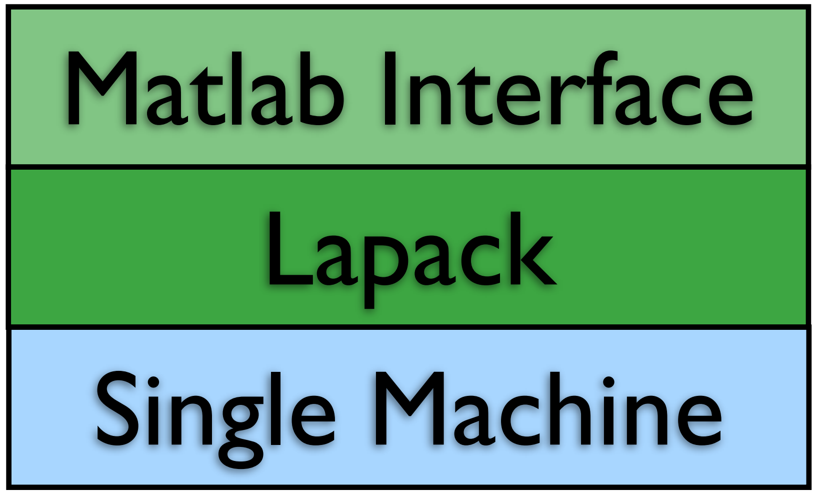
- Lapack: low-level Fortran linear algebra library
- Matlab Interface
  - Higher-level abstractions for data access / processing
  - More extensive functionality than Lapack
  - Leverages Lapack whenever possible

Matlab Stack

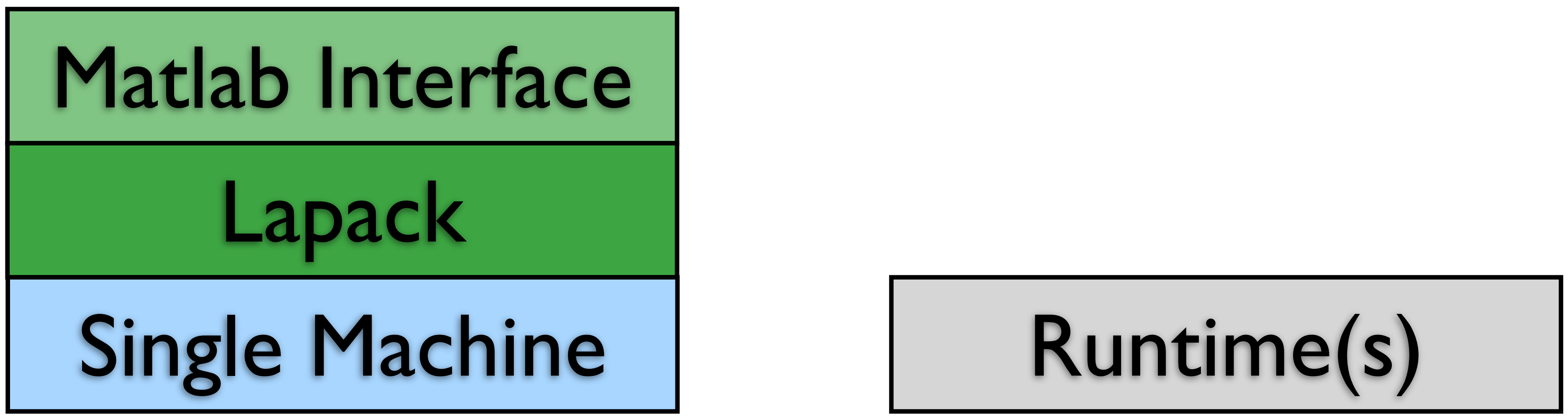


- Lapack: low-level Fortran linear algebra library
- Matlab Interface
  - Higher-level abstractions for data access / processing
  - More extensive functionality than Lapack
  - Leverages Lapack whenever possible
- Similar stories for R and Python

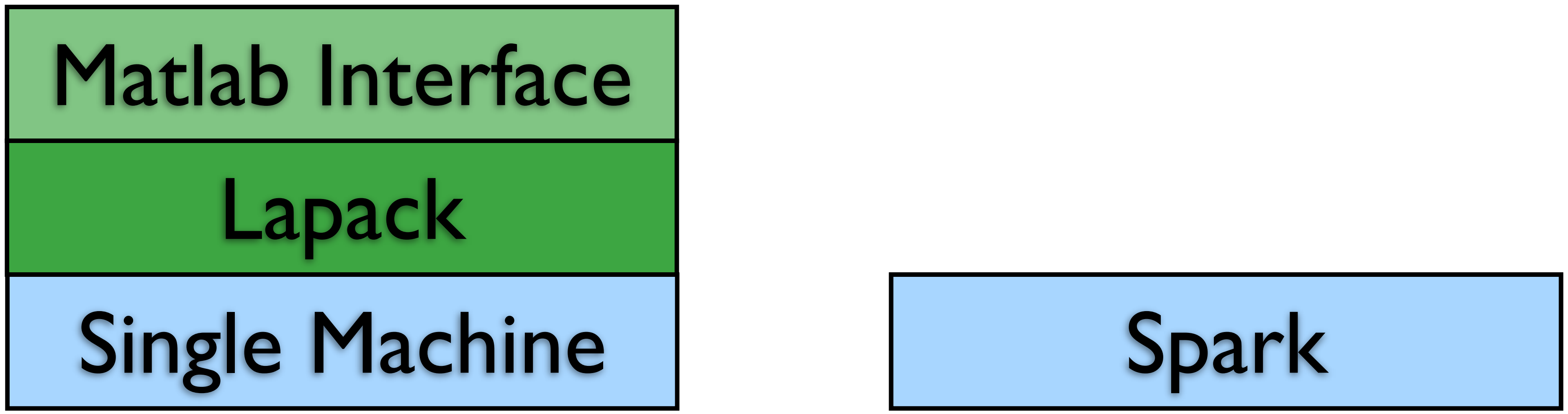
MLbase Stack



MLbase Stack

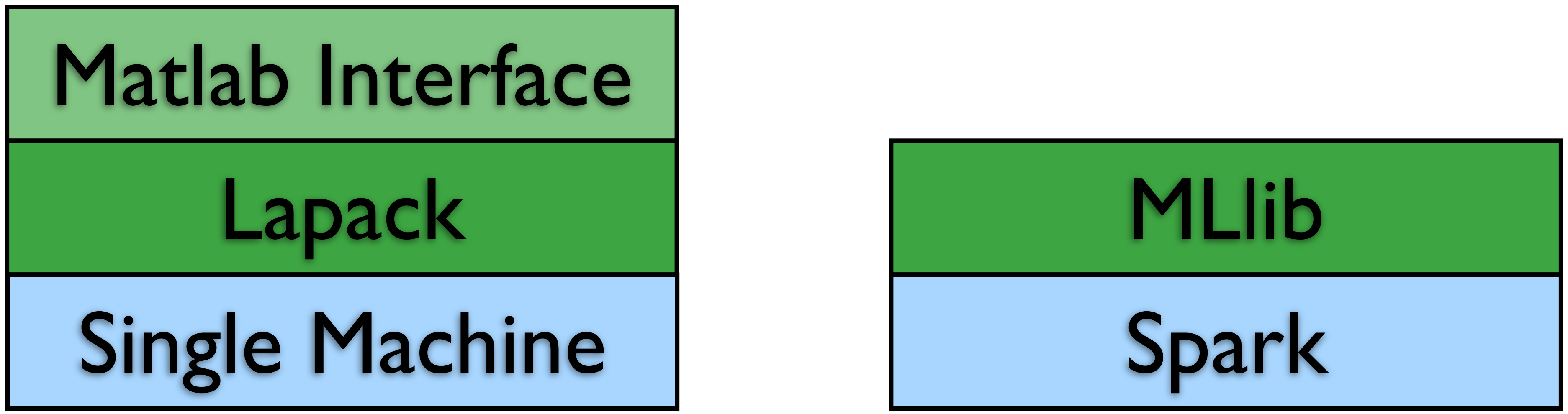


MLbase Stack



Spark: cluster computing system designed for iterative computation

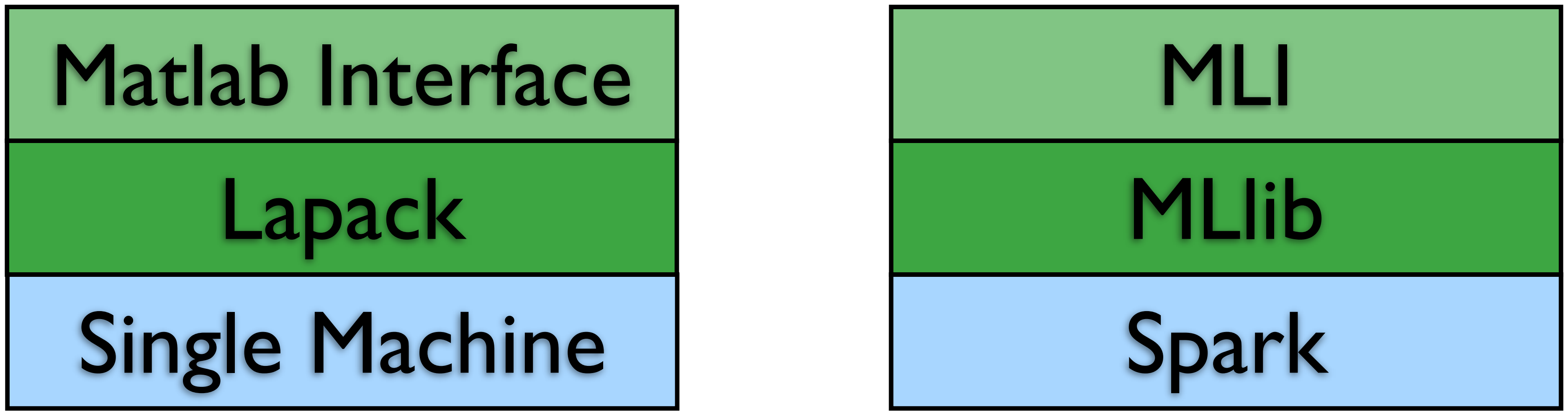
MLbase Stack



Spark: cluster computing system designed for iterative computation

MLlib: low-level ML library in Spark

MLbase Stack



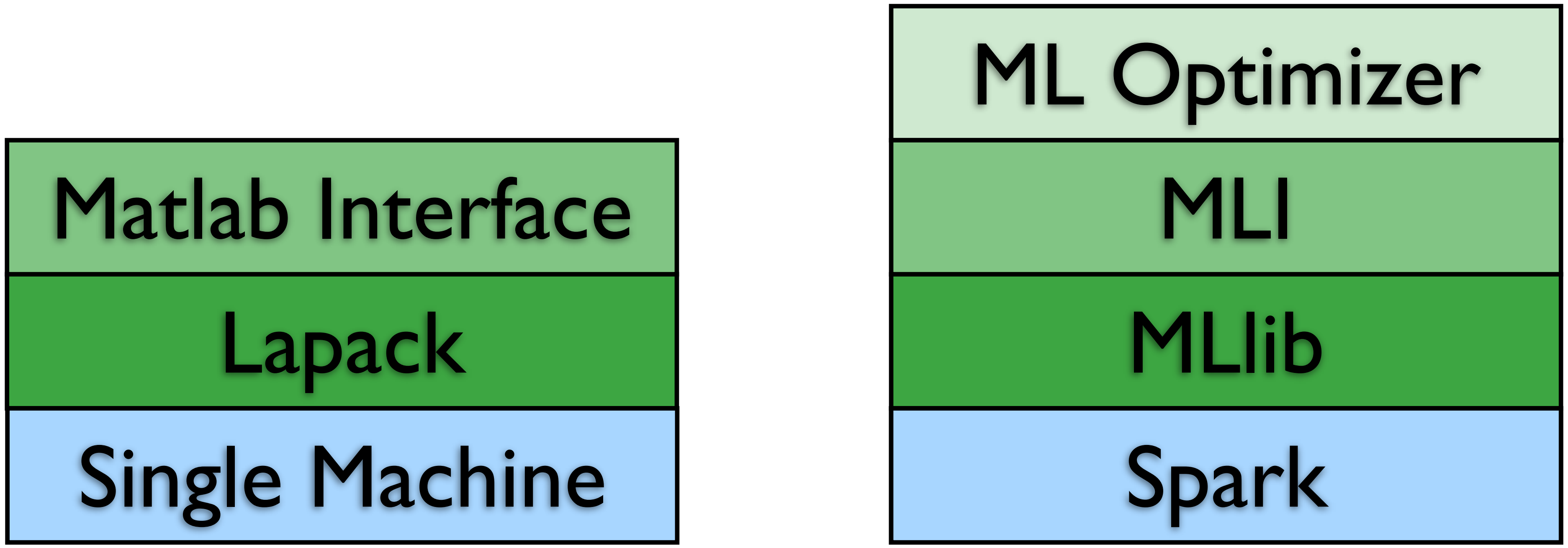
Spark: cluster computing system designed for iterative computation

MLlib: low-level ML library in Spark

MLI: API / platform for feature extraction and algorithm development

- Platform independent

MLbase Stack



Spark: cluster computing system designed for iterative computation

MLlib: low-level ML library in Spark

MLI: API / platform for feature extraction and algorithm development

- Platform independent

ML Optimizer: automates model selection

- Solves a search problem over feature extractors and algorithms in MLI

Example: MLlib

Example: MLlib

- Goal: Classification of text file

Example: MLlib

```scala
def main(args: Array[String]) {
  val sc = new SparkContext("local", "SparkLR")

  //Load data from HDFS
  val data = sc.textFile(args(0))  //RDD[String]

  //User is responsible for formatting/featurizing/normalizing their RDD!
  val featurizedData: RDD[(Double,Array[Double])] = processData(data)
```

- Goal: Classification of text file
- Featurize data manually

Example: MLlib

```
def main(args: Array[String]) {
  val sc = new SparkContext("local", "SparkLR")

  //Load data from HDFS
  val data = sc.textFile(args(0))  //RDD[String]

  //User is responsible for formatting/featurizing/normalizing their RDD!
  val featurizedData: RDD[(Double,Array[Double])] = processData(data)

  //Train the model using MLlib.
  val model = new LogisticRegressionLocalRandomSGD()
                  .setStepSize(0.1)
                  .setNumIterations(50)
                  .train(featurizedData)
}
```

- Goal: Classification of text file
- Featurize data manually
- Calls MLlib's LR function

Example: MLI

Example: MLI

```
def main(args: Array[String]) {
        val mc = new MLContext("local", "MLILR")

        //Read in file from HDFS
        val rawTextTable = mc.csvFile(args(0), Seq("class","text"))

        //Run feature extraction
        val classes = rawTextTable(??, "class")
        val ngrams = tfIdf(nGrams(rawTextTable(??, "text"), n=2, top=30000))
        val featureizedTable = classes.zip(ngrams)
```

- Use built-in feature extraction functionality

Example: MLI

```
def main(args: Array[String]) {
        val mc = new MLContext("local", "MLILR")

        //Read in file from HDFS
        val rawTextTable = mc.csvFile(args(0), Seq("class","text"))

        //Run feature extraction
        val classes = rawTextTable(??, "class")
        val ngrams = tfIdf(nGrams(rawTextTable(??, "text"), n=2, top=30000))
        val featureizedTable = classes.zip(ngrams)

        //Classify the data using Logistic Regression.
        val lrModel = LogisticRegression(featurizedTable, stepSize=0.1, numIter=12)
}
```

- Use built-in feature extraction functionality
- MLI Logistic Regression leverages MLlib

Example: MLI

```
def main(args: Array[String]) {
        val mc = new MLContext("local", "MLILR")

        //Read in file from HDFS
        val rawTextTable = mc.csvFile(args(0), Seq("class","text"))

        //Run feature extraction
        val classes = rawTextTable(??, "class")
        val ngrams = tfIdf(nGrams(rawTextTable(??, "text"), n=2, top=30000))
        val featureizedTable = classes.zip(ngrams)

        //Classify the data using Logistic Regression.
        val lrModel = LogisticRegression(featurizedTable, stepSize=0.1, numIter=12)
}
```

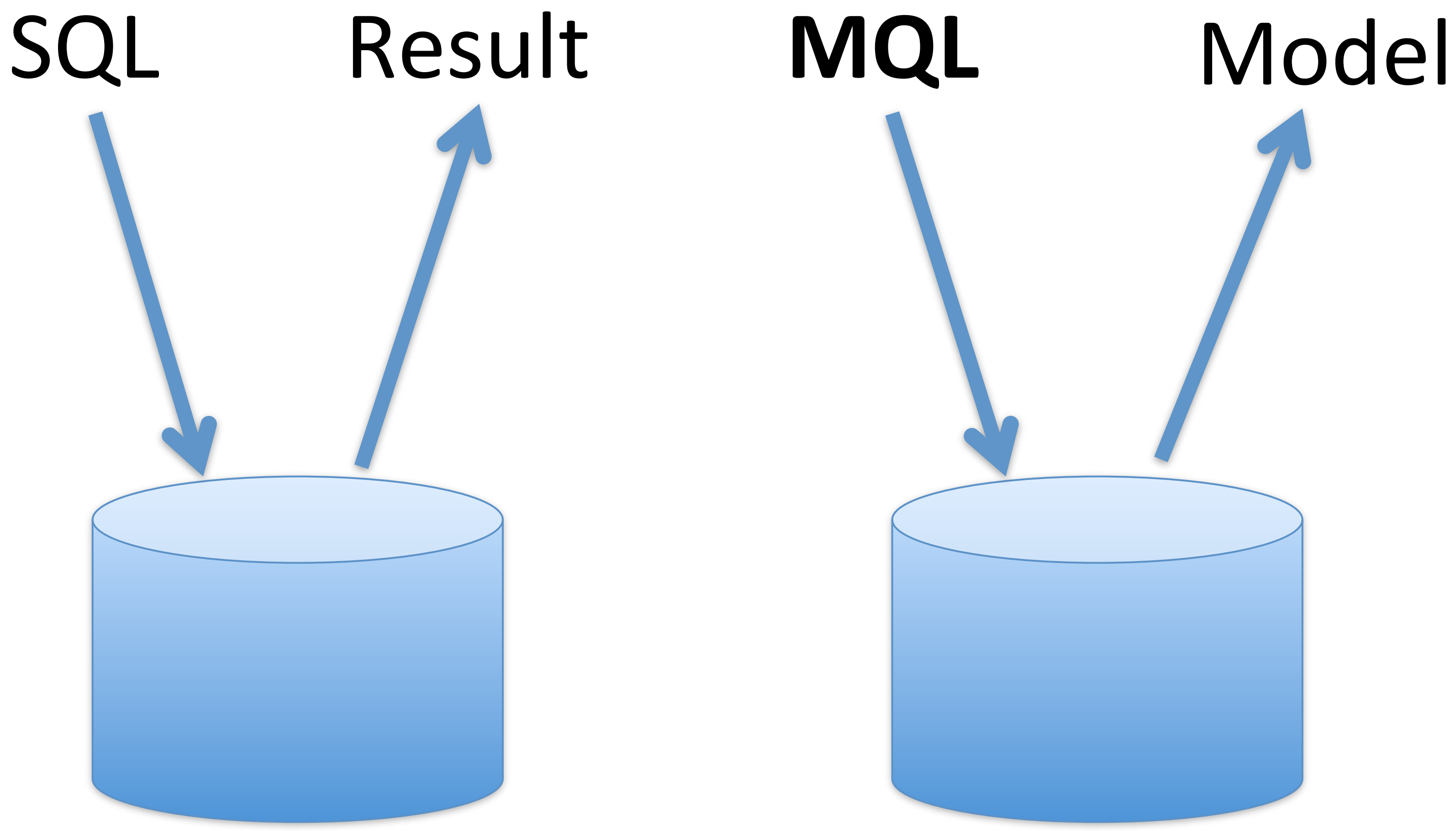
- Use built-in feature extraction functionality
- MLI Logistic Regression leverages MLlib
- Extensions:
  - Embed in cross-validation routine
  - Use different feature extractors / algorithms
  - Write new ones

Example: ML Optimizer

```
var X = load("text_file", 2 to 10)
var y = load("text_file", 1)
var (fn-model, summary) = doClassify(X, y)
```

- User declaratively specifies task
- ML Optimizer searches through MLI

Example: ML Optimizer



- User declaratively specifies task
- ML Optimizer searches through MLI

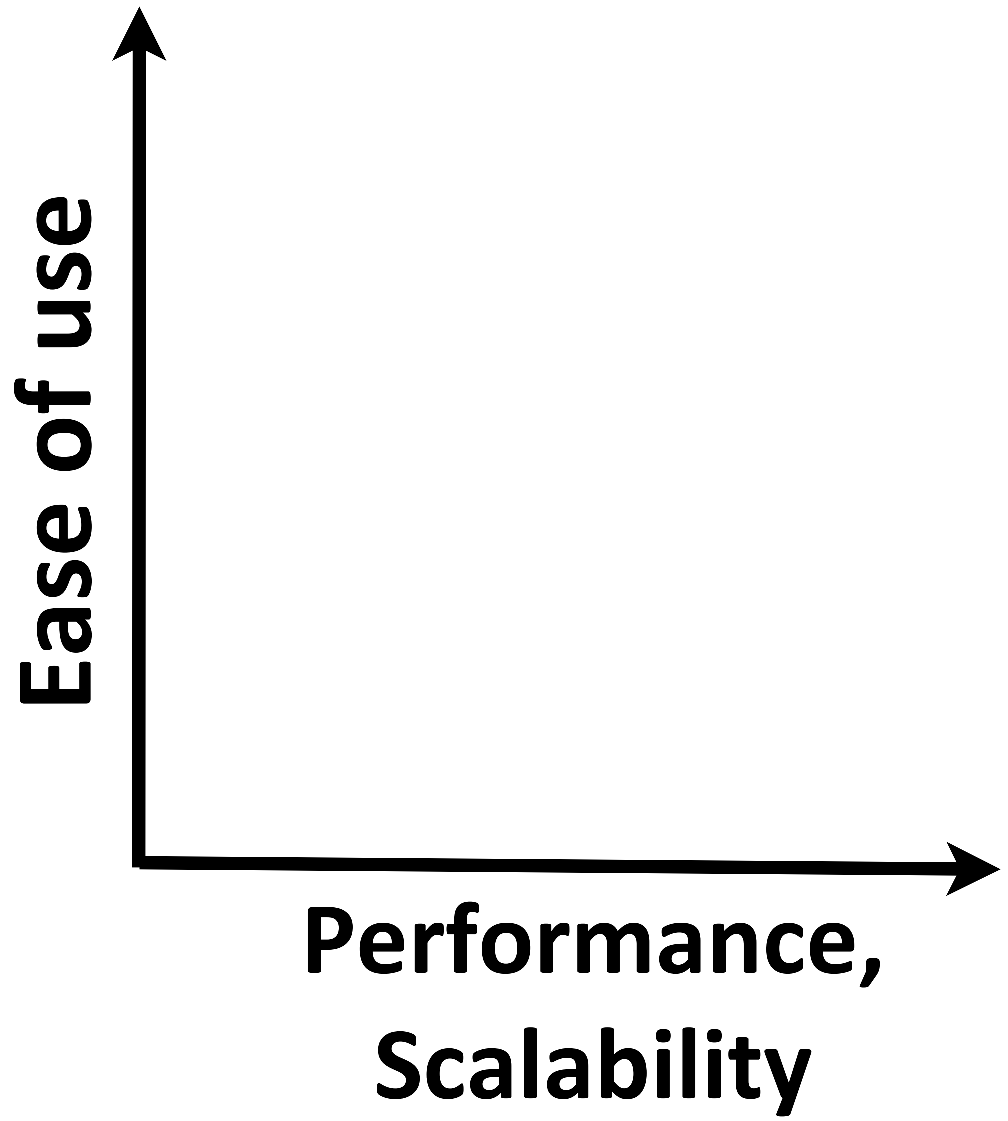
Vision

MLI Details

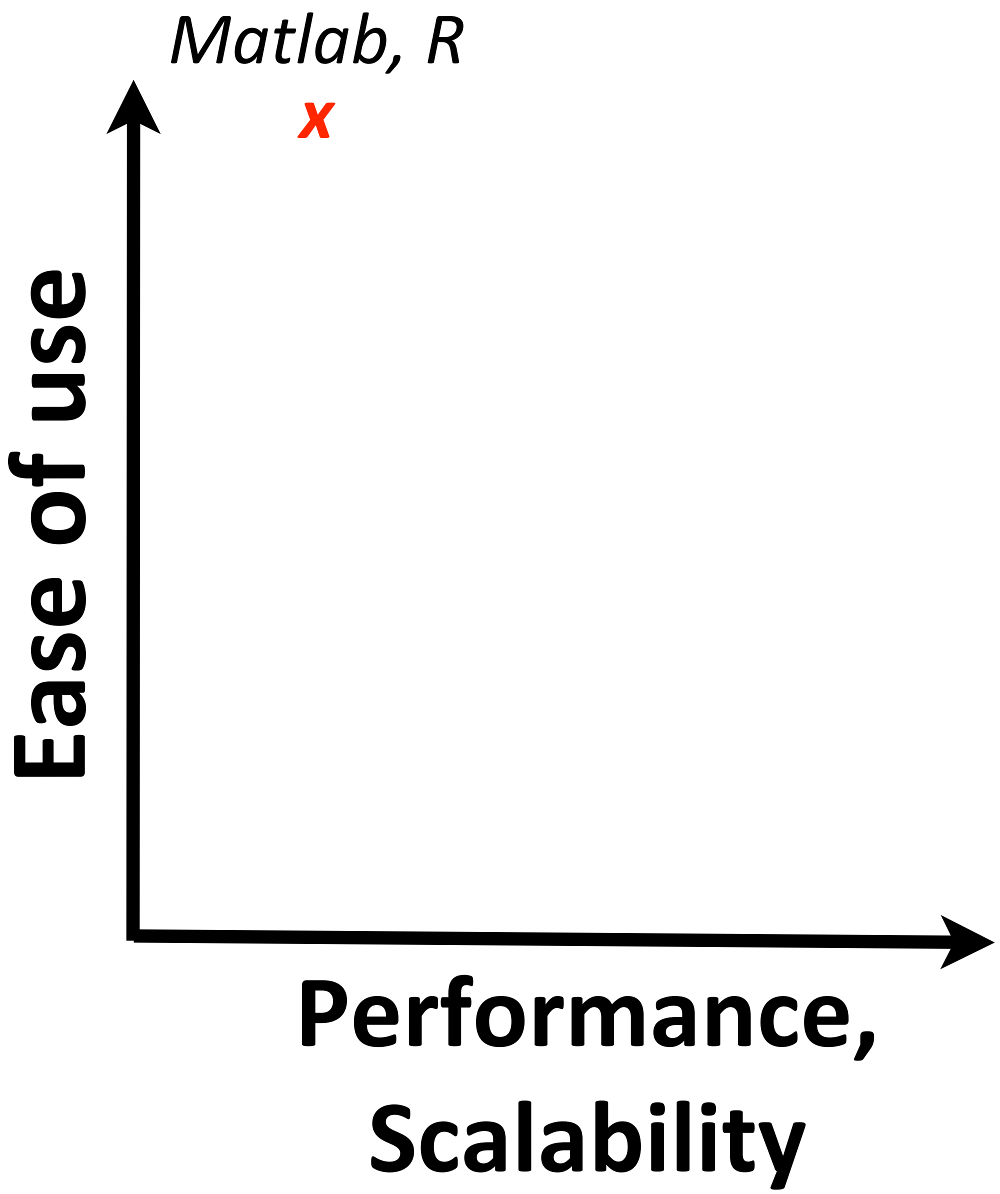
Current Status

ML Workflow

Lay of the Land

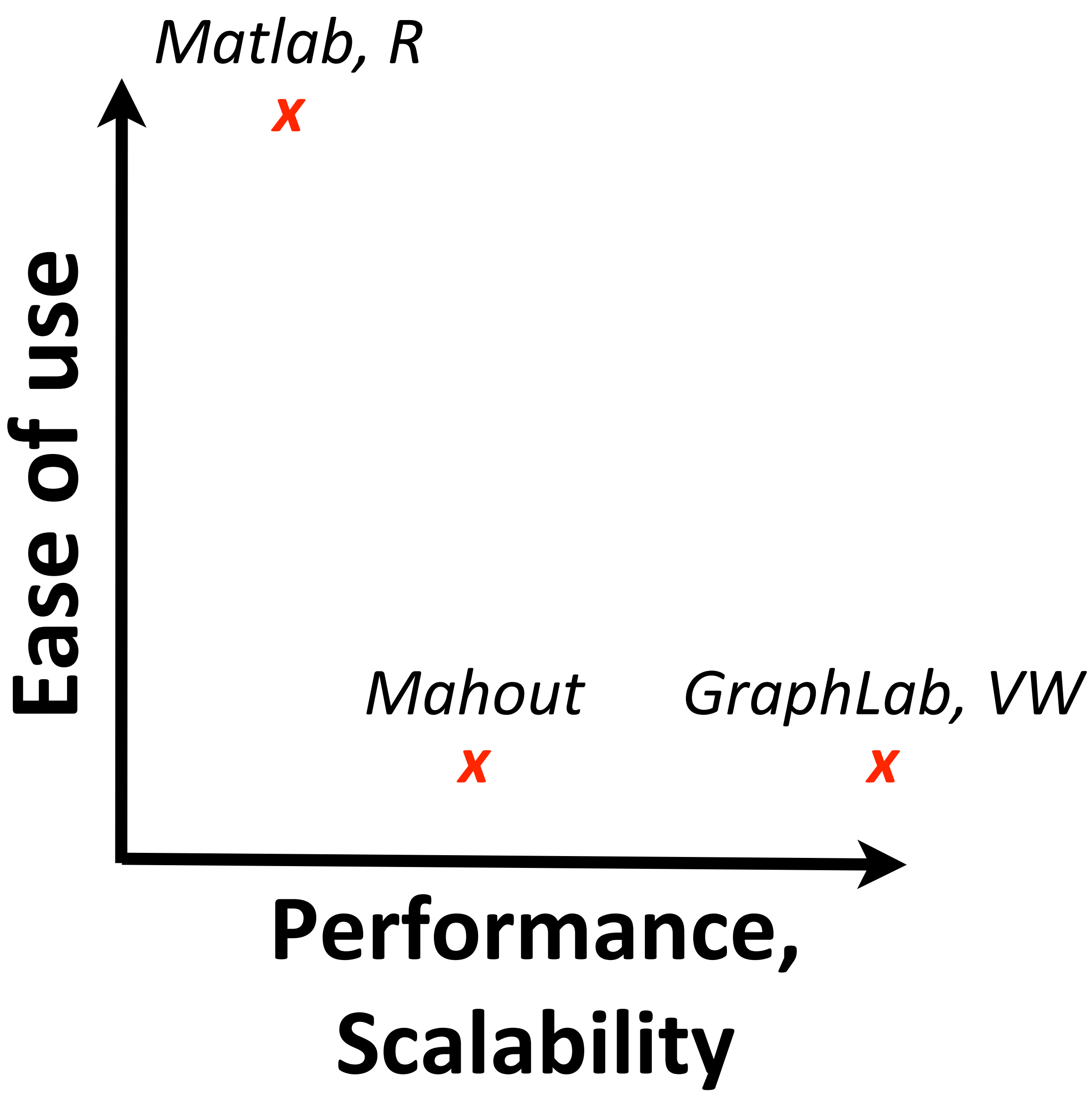


Lay of the Land



- \+ Easy (Resembles math, limited set up)
- \+ Sufficient for prototyping / writing papers
- — Ad-hoc, non-scalable scripts
- — Loss of translation upon re-implementation

Lay of the Land



+ Easy (Resembles math, limited set up)
+ Sufficient for prototyping / writing papers
— Ad-hoc, non-scalable scripts
— Loss of translation upon re-implementation

+ Scalable and (sometimes) fast
+ Existing open-source libraries
— Difficult to set up, extend

Examples

Examples

‘Distributed’ Divide-Factor-Combine (DFC)

- Initial studies in MATLAB (Not distributed)
- Distributed prototype involving compiled MATLAB

Examples

'Distributed' Divide-Factor-Combine (DFC)

- Initial studies in MATLAB (Not distributed)
- Distributed prototype involving compiled MATLAB

Mahout ALS with Early Stopping

- Theory: simple if-statement (3 lines of code)

Examples

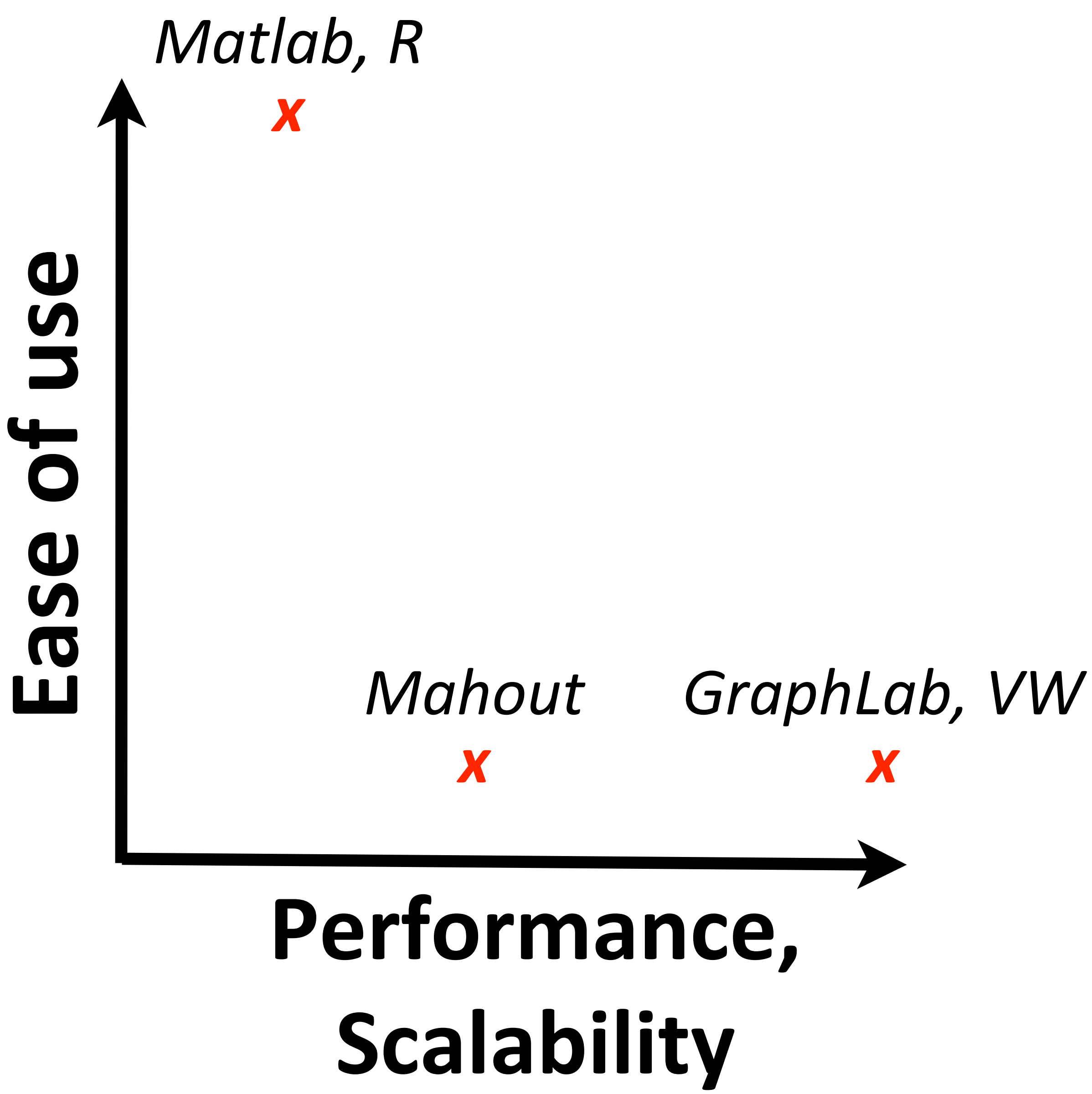
'Distributed' Divide-Factor-Combine (DFC)

- Initial studies in MATLAB (Not distributed)
- Distributed prototype involving compiled MATLAB

Mahout ALS with Early Stopping

- Theory: simple if-statement (3 lines of code)
- Practice: sift through 7 files, nearly 1K lines of code

Lay of the Land



MATLAB
R

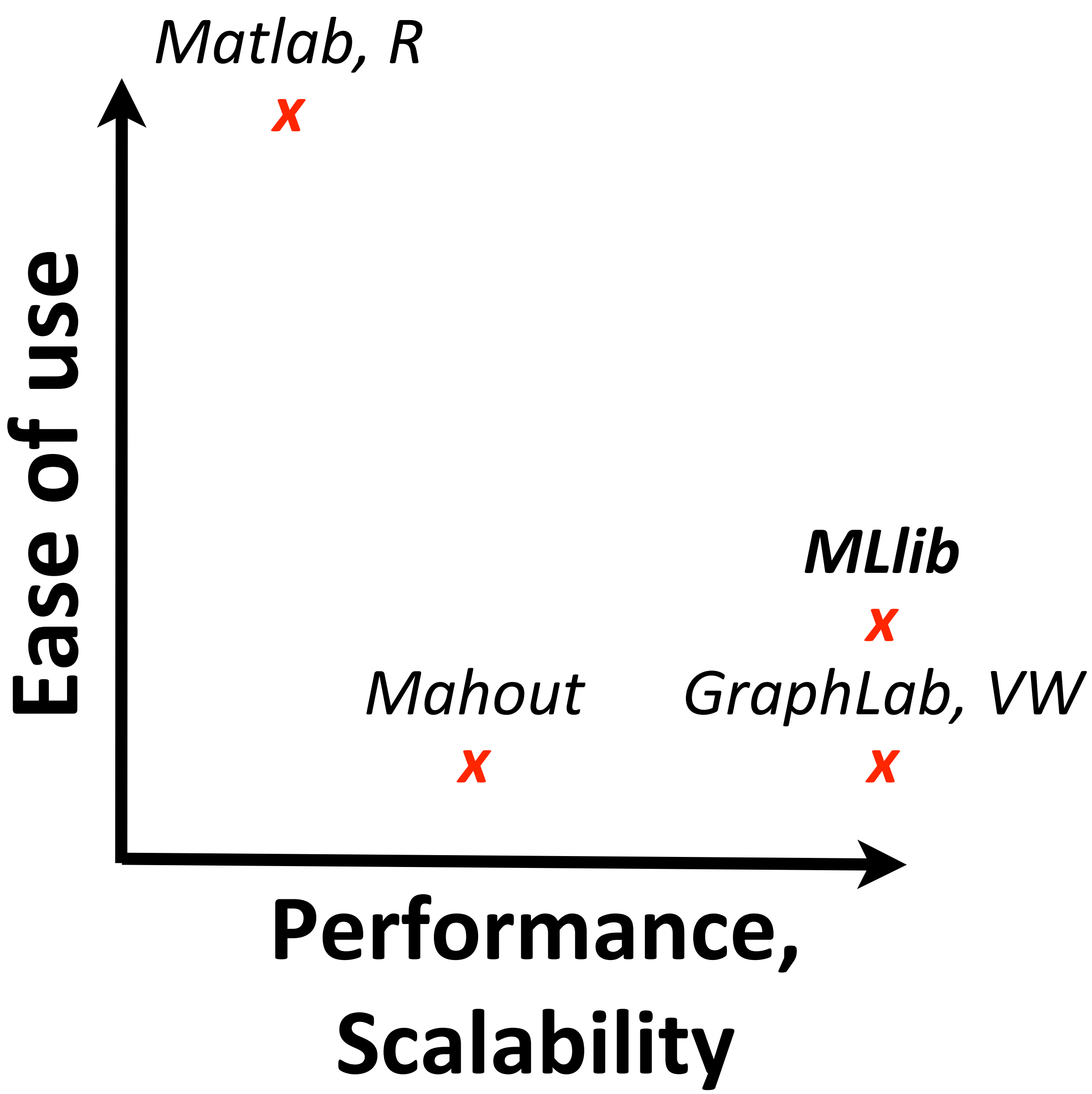
- \+ Easy (Resembles math, limited set up)
- \+ Sufficient for prototyping / writing papers
- — Ad-hoc, non-scalable scripts
- — Loss of translation upon re-implementation

- \+ Scalable and (sometimes) fast
- \+ Existing open-source libraries
- — Difficult to set up, extend

mahout

VOWPAL WABBIT GraphLab

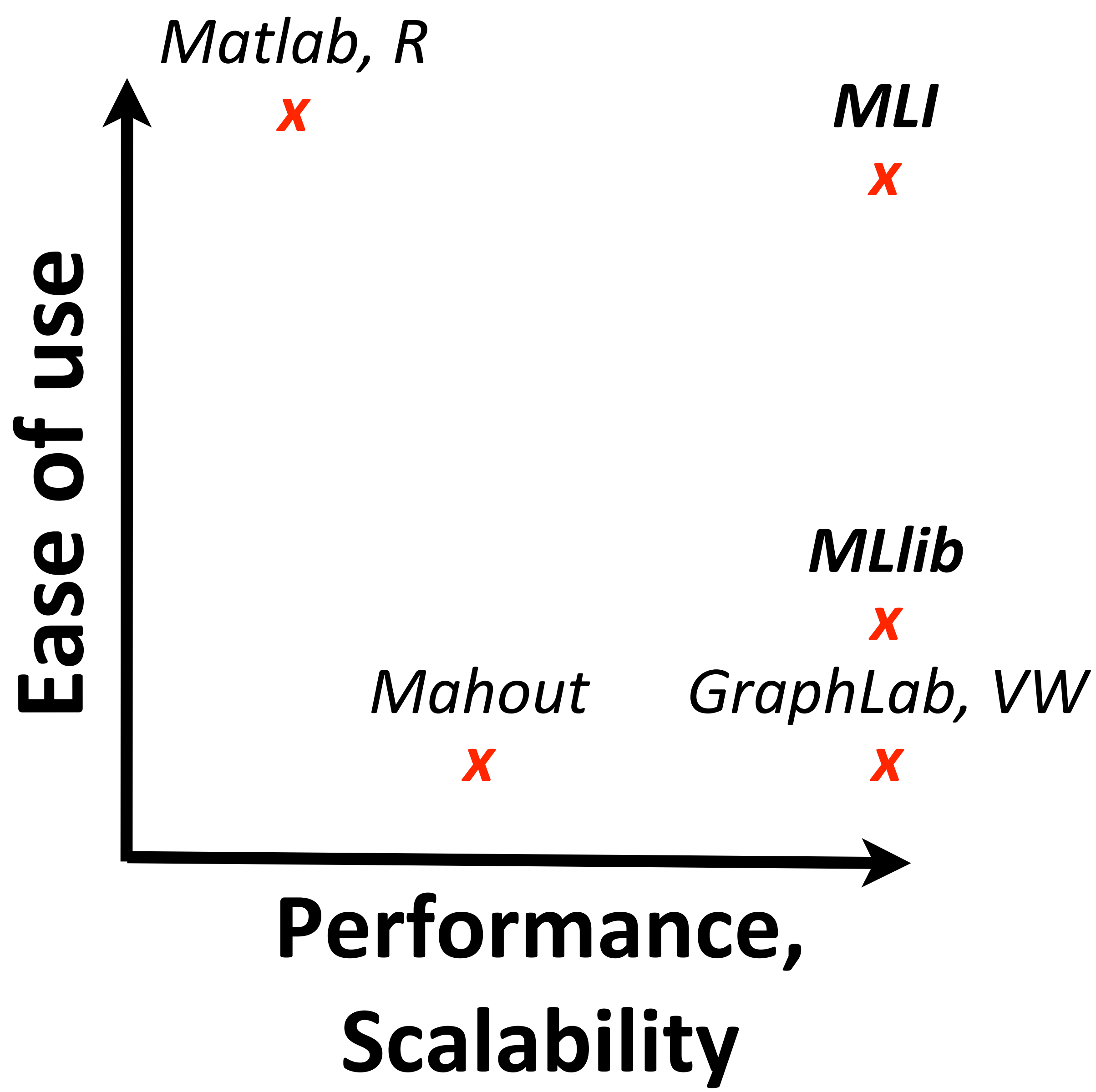
Lay of the Land



+ Easy (Resembles math, limited set up)
+ Sufficient for prototyping / writing papers
— Ad-hoc, non-scalable scripts
— Loss of translation upon re-implementation

+ Scalable and (sometimes) fast
+ Existing open-source libraries
— Difficult to set up, extend

Lay of the Land



- \+ Easy (Resembles math, limited set up)
- \+ Sufficient for prototyping / writing papers
- — Ad-hoc, non-scalable scripts
- — Loss of translation upon re-implementation

- \+ Scalable and (sometimes) fast
- \+ Existing open-source libraries
- — Difficult to set up, extend

ML Developer API (MLI)

ML Developer API (MLI)

OLD

```
val x: RDD[Array[Double]]
```

ML Developer API (MLI)

OLD

val x: RDD[Array[Double]]

val x: RDD[spark.util.Vector]

ML Developer API (MLI)

OLD

```
val x: RDD[Array[Double]]
val x: RDD[spark.util.Vector]
val x: RDD[breeze.linalg.Vector]
```

ML Developer API (MLI)

OLD

```
val x: RDD[Array[Double]]
val x: RDD[spark.util.Vector]
val x: RDD[breeze.linalg.Vector]
val x: RDD[BIDMat.SMat]
```

ML Developer API (MLI)

OLD

~~val x: RDD[Array[Double]]~~

~~val x: RDD[spark.util.Vector]~~

~~val x: RDD[breeze.linalg.Vector]~~

~~val x: RDD[BIDMat.SMat]~~

ML Developer API (MLI)

OLD

~~val x: RDD[Array[Double]]~~
~~val x: RDD[spark.util.Vector]~~
~~val x: RDD[breeze.linalg.Vector]~~
~~val x: RDD[BIDMat.SMat]~~

NEW

val x: MLTable

ML Developer API (MLI)

OLD

~~val x: RDD[Array[Double]]~~
~~val x: RDD[spark.util.Vector]~~
~~val x: RDD[breeze.linalg.Vector]~~
~~val x: RDD[BIDMat.SMat]~~

NEW

val x: MLTable

- Abstract interface for arbitrary backend
- Common interface to support an optimizer

ML Developer API (MLI)

ML Developer API (MLI)

- Shield ML Developers from low-details
  - provide familiar mathematical operators in distributed setting

ML Developer API (MLI)

- Shield ML Developers from low-details
  - provide familiar mathematical operators in distributed setting

- Table Computation (*MLTable*)
  - Flexibility when loading data (heterogenous, missing)
  - Common interface for feature extraction / algorithms
  - Supports MapReduce and relational operators

ML Developer API (MLI)

- Shield ML Developers from low-details
  - provide familiar mathematical operators in distributed setting
- Table Computation (*MLTable*)
  - Flexibility when loading data (heterogenous, missing)
  - Common interface for feature extraction / algorithms
  - Supports MapReduce and relational operators

- Linear Algebra (*MLSubMatrix*)
  - Linear algebra on *local* partitions
  - Sparse and Dense matrix support

ML Developer API (MLI)

- Shield ML Developers from low-details
  - provide familiar mathematical operators in distributed setting
- Table Computation (*MLTable*)
  - Flexibility when loading data (heterogenous, missing)
  - Common interface for feature extraction / algorithms
  - Supports MapReduce and relational operators

- Linear Algebra (*MLSubMatrix*)
  - Linear algebra on *local* partitions
  - Sparse and Dense matrix support

- Optimization Primitives (*MLSolve*)
  - Distributed implementations of common patterns

ML Developer API (MLI)

- Shield ML Developers from low-details
  - provide familiar mathematical operators in distributed setting
- Table Computation (*MLTable*)
  - Flexibility when loading data (heterogenous, missing)
  - Common interface for feature extraction / algorithms
  - Supports MapReduce and relational operators

- Linear Algebra (*MLSubMatrix*)
  - Linear algebra on *local* partitions
  - Sparse and Dense matrix support

- Optimization Primitives (*MLSolve*)
  - Distributed implementations of common patterns

- DFC: ~50 lines of code
- ALS: early stopping in 3 lines; < 40 lines total

MLI Ease of Use

MLI Ease of Use

Logistic Regression

System	Lines of Code
Matlab	11

Alternating Least Squares

System	Lines of Code
Matlab	20

MLI Ease of Use

Logistic Regression

System	Lines of Code
Matlab	11
Vowpal Wabbit	721

Alternating Least Squares

System	Lines of Code
Matlab	20
Mahout	865
GraphLab	383

MLI Ease of Use

Logistic Regression

System	Lines of Code
Matlab	11
Vowpal Wabbit	721
MLI	55

Alternating Least Squares

System	Lines of Code
Matlab	20
Mahout	865
GraphLab	383
MLI	32

MLI/Spark Performance

MLI/Spark Performance

- **Walltime**: elapsed time to execute task

MLI/Spark Performance

- **Walltime**: elapsed time to execute task
- **Weak scaling**
  - fix problem size *per processor*
  - ideally: constant walltime as we grow cluster

MLI/Spark Performance

- **Walltime**: elapsed time to execute task
- **Weak scaling**
  - fix problem size *per processor*
  - ideally: constant walltime as we grow cluster
- **Strong scaling**
  - fix total problem size
  - ideally: linear speed up as we grow cluster

MLI/Spark Performance

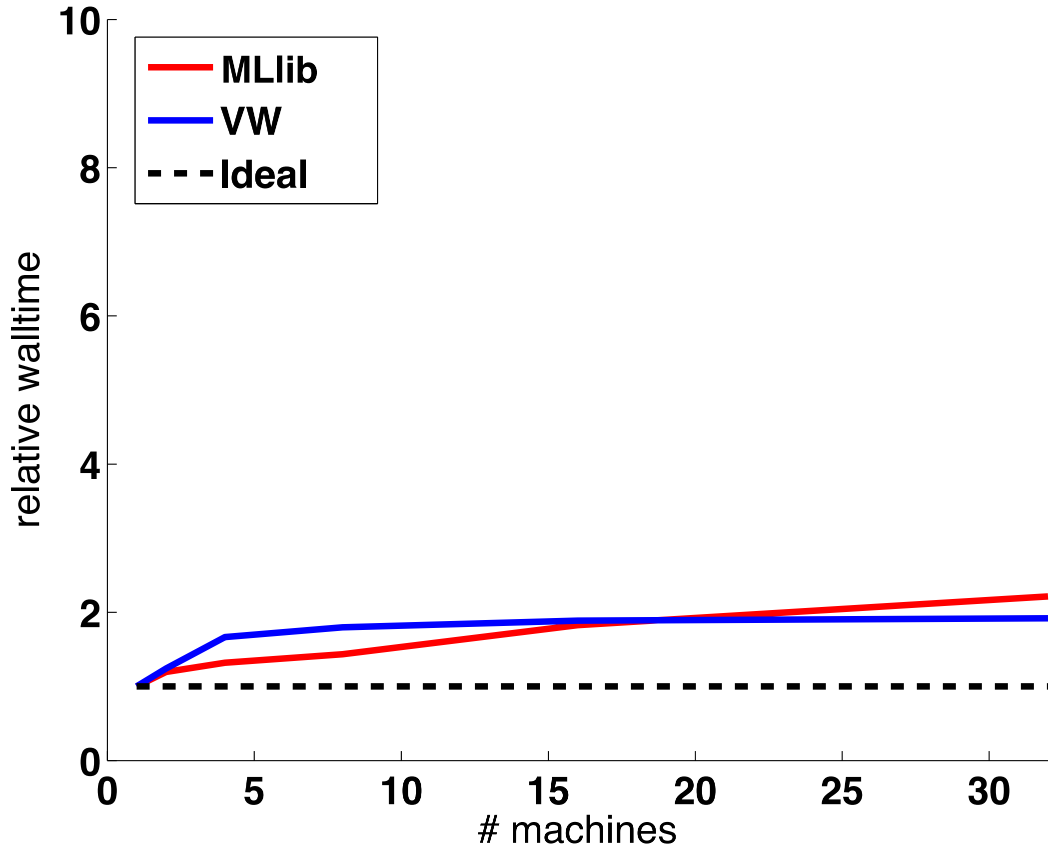
- **Walltime**: elapsed time to execute task
- **Weak scaling**
  - fix problem size *per processor*
  - ideally: constant walltime as we grow cluster
- **Strong scaling**
  - fix total problem size
  - ideally: linear speed up as we grow cluster
- **EC2 Experiments**
  - m2.4xlarge instances, up to 32 machine clusters

Logistic Regression - Weak Scaling

Logistic Regression - Weak Scaling

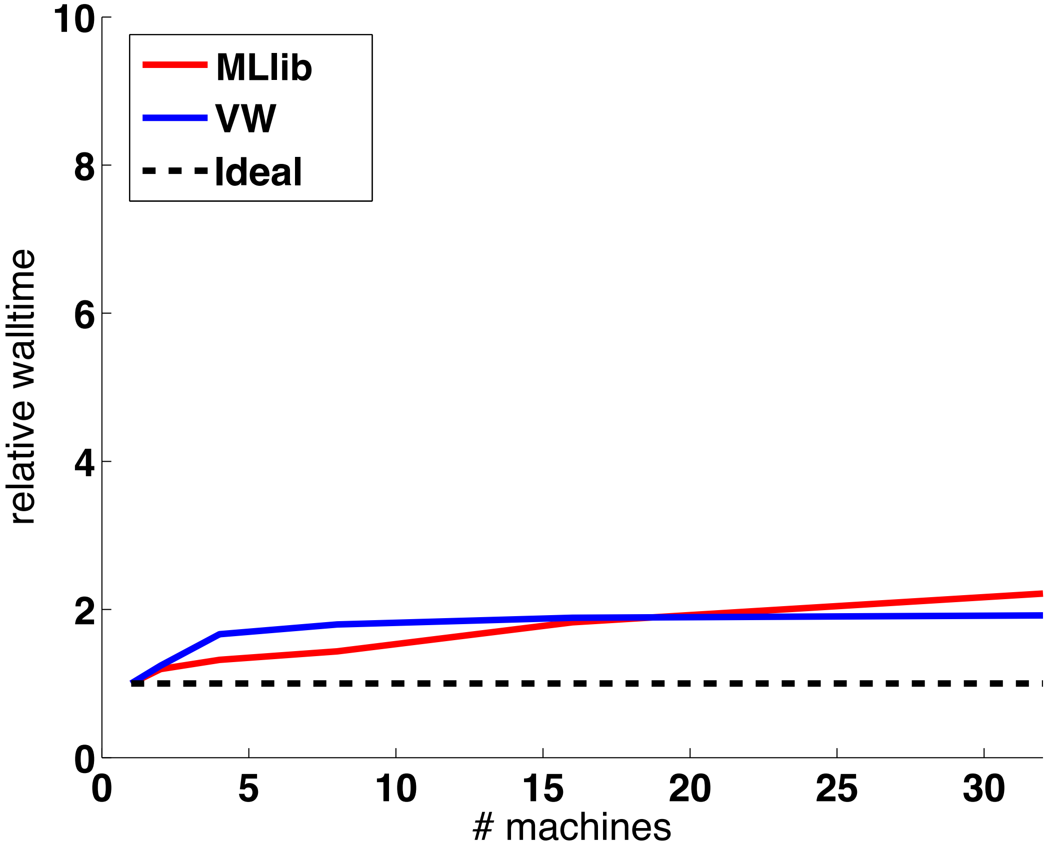
- Full dataset: 200K images, 160K dense features

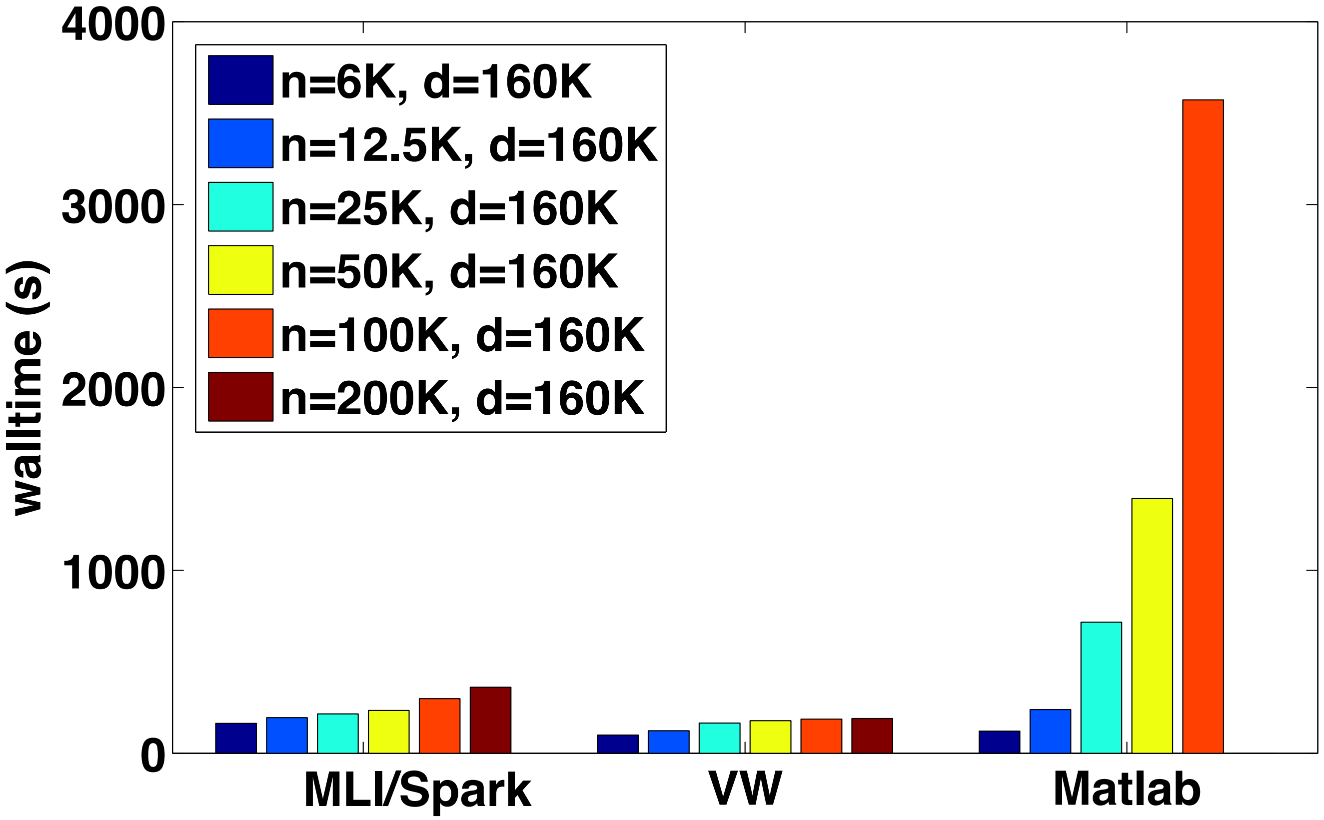
Logistic Regression - Weak Scaling



- ✦ Full dataset: 200K images, 160K dense features
- ✦ Similar weak scaling

Logistic Regression - Weak Scaling





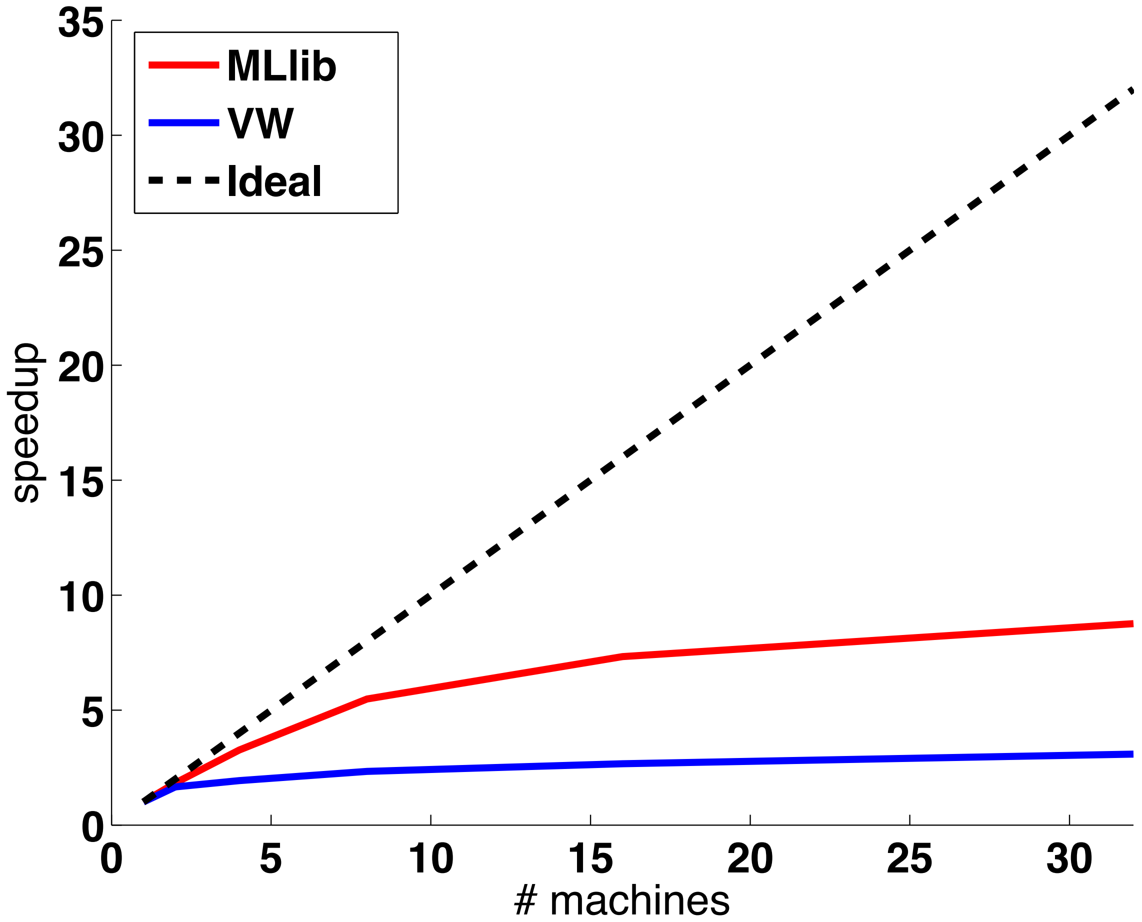
- Full dataset: 200K images, 160K dense features
- Similar weak scaling
- MLI/Spark within a factor of 2 of VW's walltime

Logistic Regression - Strong Scaling

Logistic Regression - Strong Scaling

- Fixed Dataset: 50K images, 160K dense features

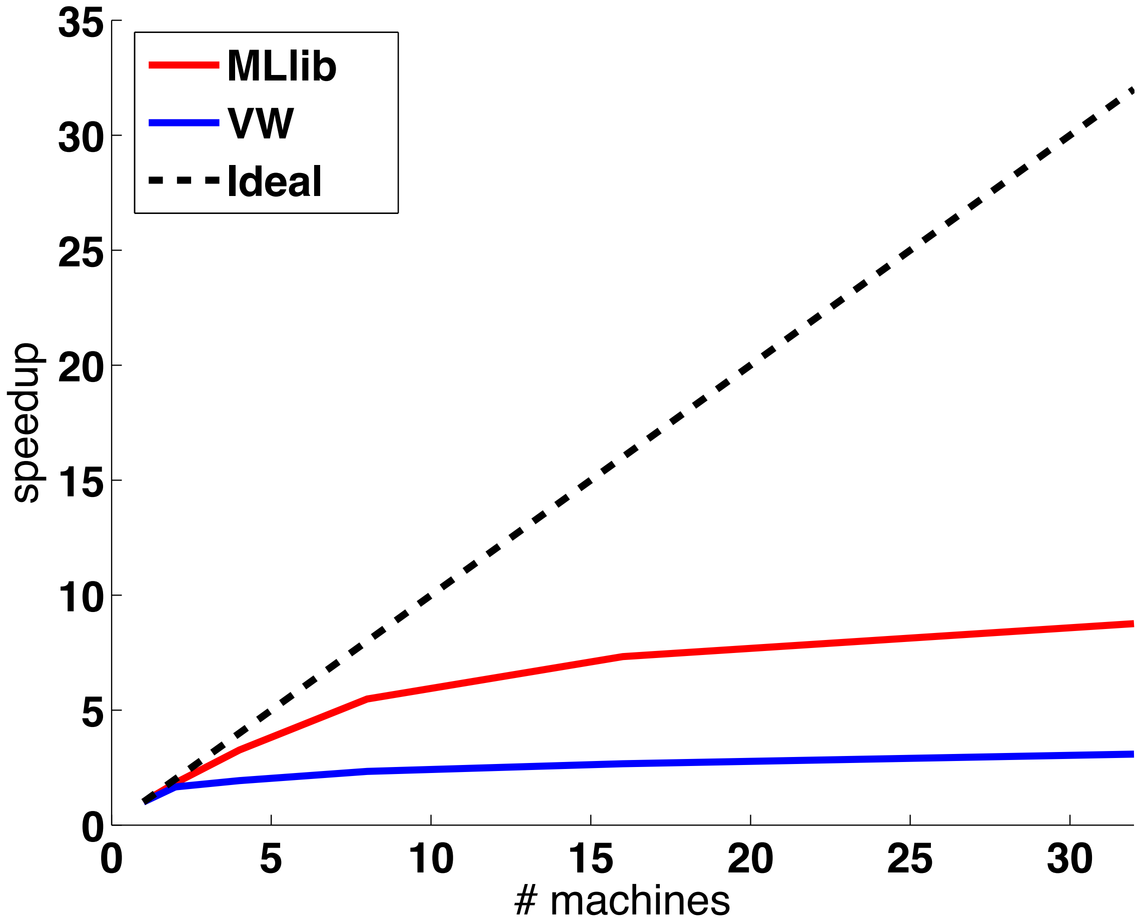
Logistic Regression - Strong Scaling

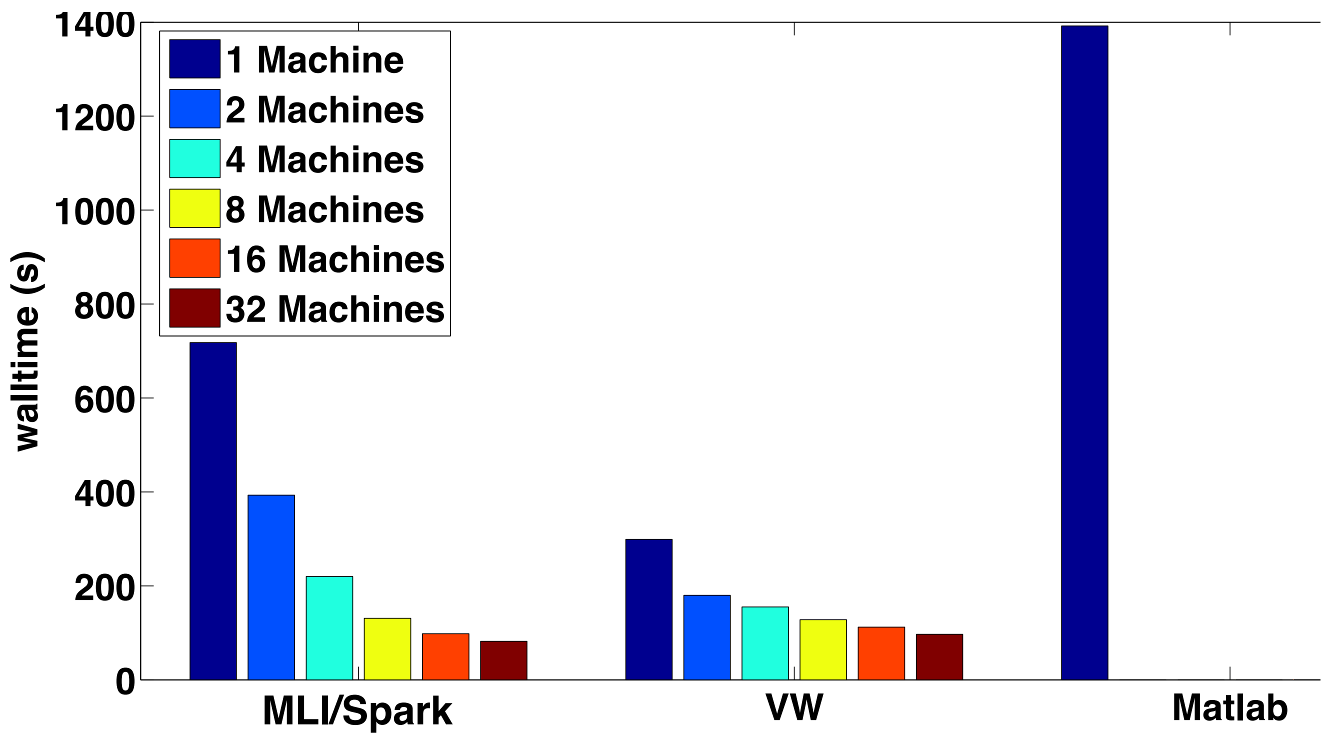


✦ Fixed Dataset: 50K images, 160K dense features

✦ MLI/Spark exhibits better scaling properties

Logistic Regression - Strong Scaling





- Fixed Dataset: 50K images, 160K dense features
- MLI/Spark exhibits better scaling properties
- MLI/Spark faster than VW with 16 and 32 machines

ALS - Walltime

ALS - Walltime

- Dataset: Scaled version of Netflix data (9X in size)
- Cluster: 9 machines

ALS - Walltime

System	Walltime (seconds)
Matlab	15443

- Dataset: Scaled version of Netflix data (9X in size)
- Cluster: 9 machines

ALS - Walltime

System	Walltime (seconds)
Matlab	15443
Mahout	4206

- Dataset: Scaled version of Netflix data (9X in size)
- Cluster: 9 machines

ALS - Walltime

System	Walltime (seconds)
Matlab	15443
Mahout	4206
GraphLab	291
MLI/Spark	481

- Dataset: Scaled version of Netflix data (9X in size)
- Cluster: 9 machines
- MLI/Spark an order of magnitude faster than Mahout
- MLI/Spark within factor of 2 of GraphLab

Vision

MLI Details

Current Status

ML Workflow

MLI Functionality

Regression: Linear Regression (+Lasso, Ridge)

Collaborative Filtering: Alternating Least Squares

Clustering: K-Means

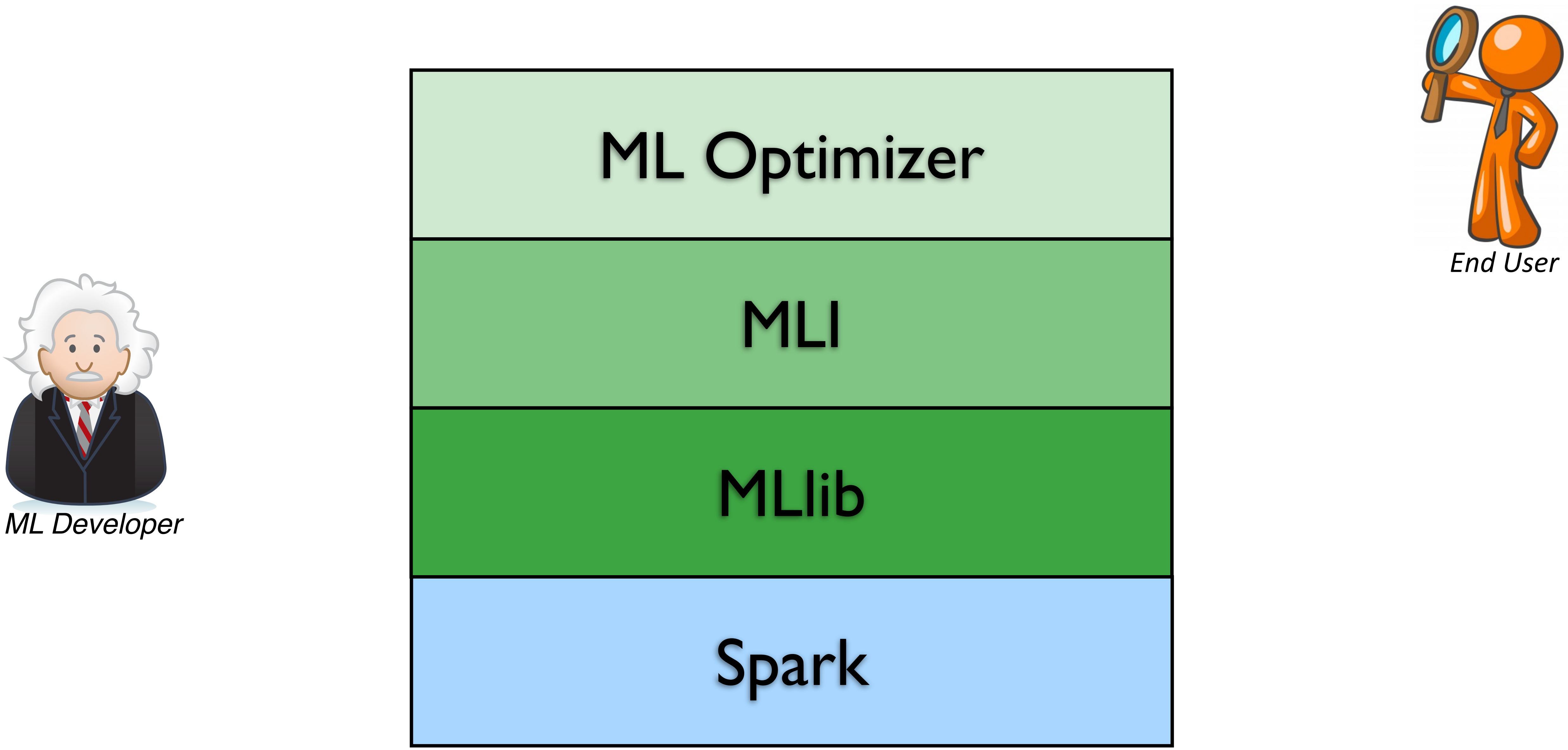
Classification: Logistic Regression, Linear SVM (+L1, L2)

Optimization Primitives: Parallel Gradient

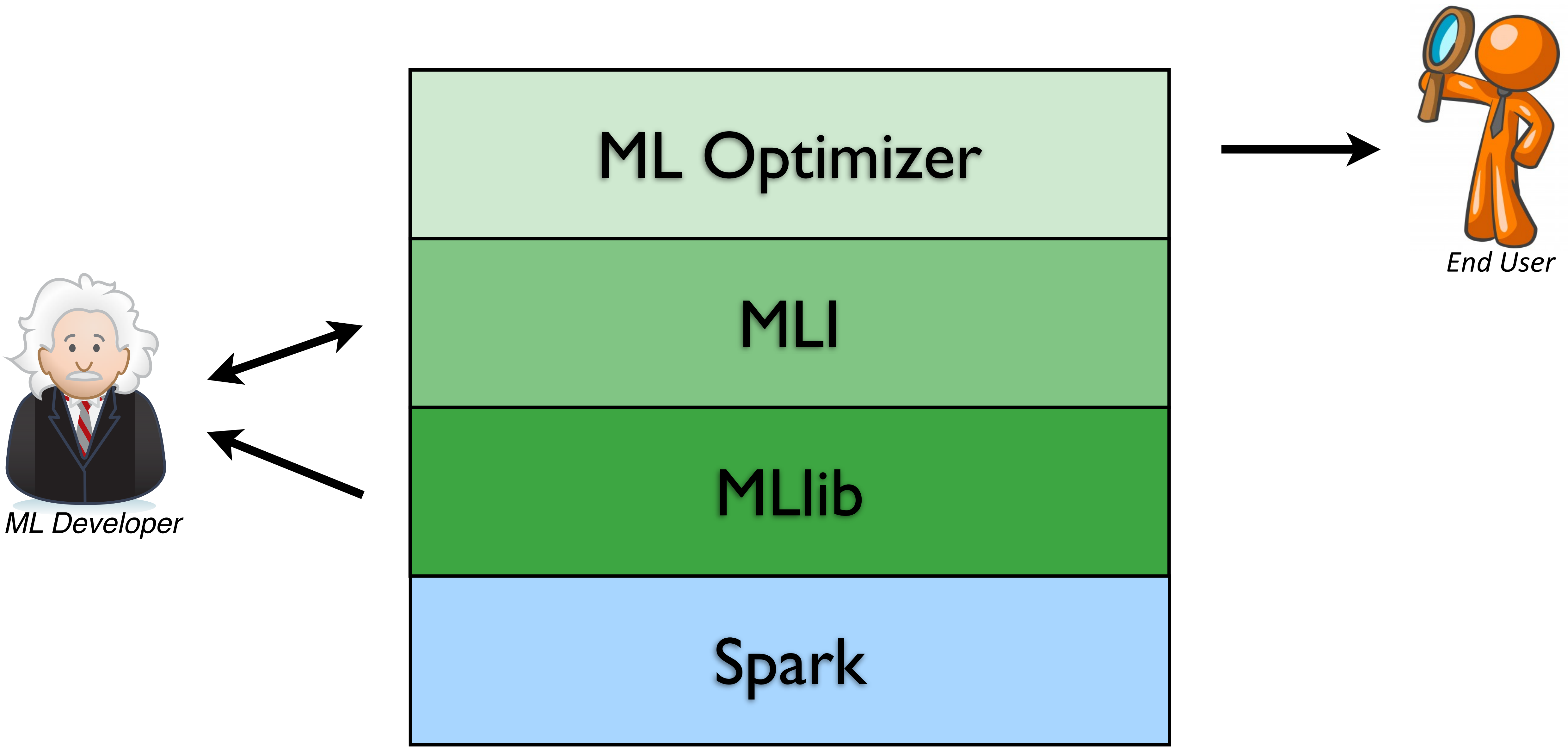
MLI Functionality

Regression:	Linear Regression (+Lasso, Ridge)
Collaborative Filtering:	Alternating Least Squares, DFC
Clustering:	K-Means, DP-Means
Classification:	Logistic Regression, Linear SVM (+L1, L2), Multinomial Regression, Naive Bayes, Decision Trees
Optimization Primitives:	Parallel Gradient, Local SGD, L-BFGS, ADMM, Adagrad
Feature Extraction:	Principal Component Analysis (PCA), N-grams, feature normalization
ML Tools:	Cross Validation, Evaluation Metrics

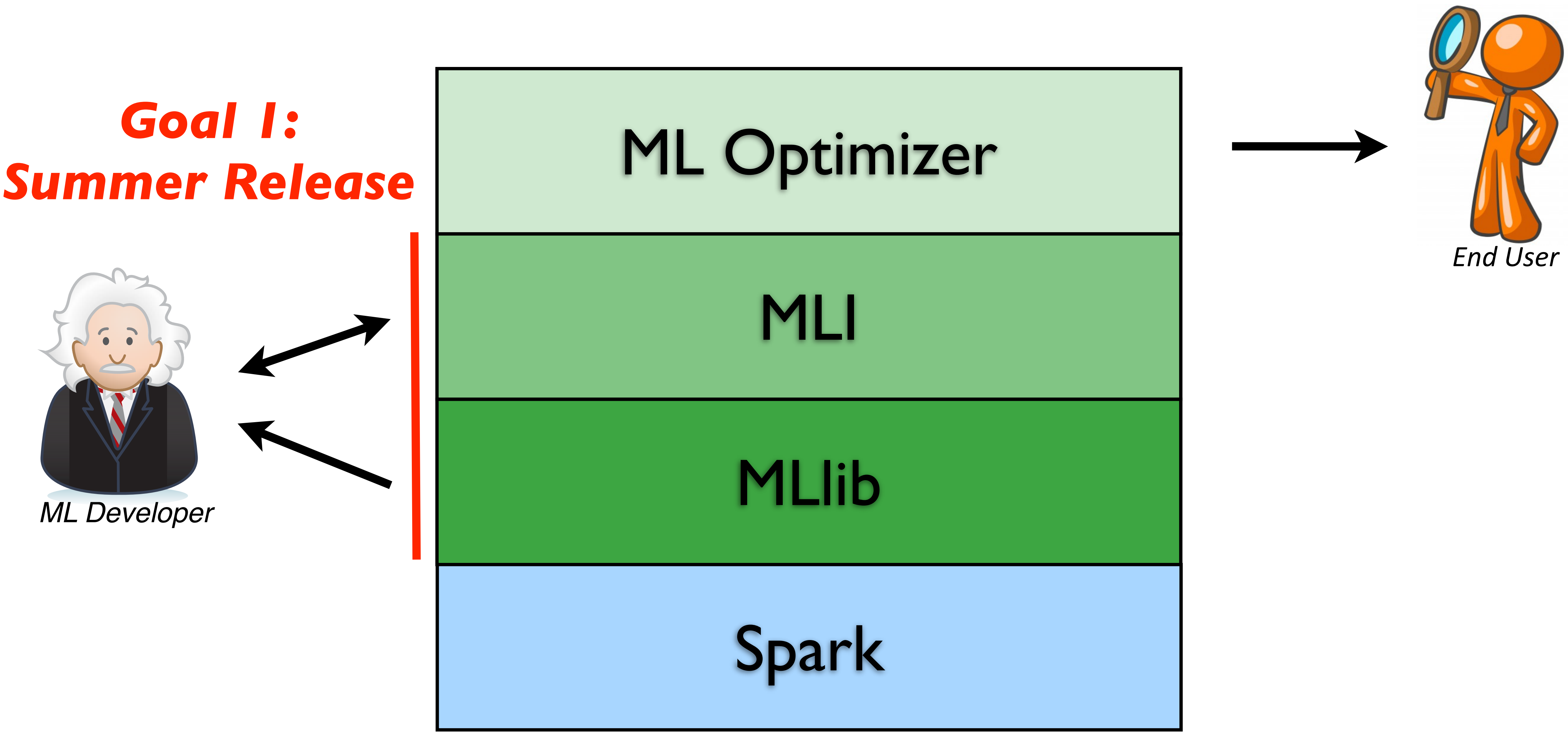
MLbase Stack Status



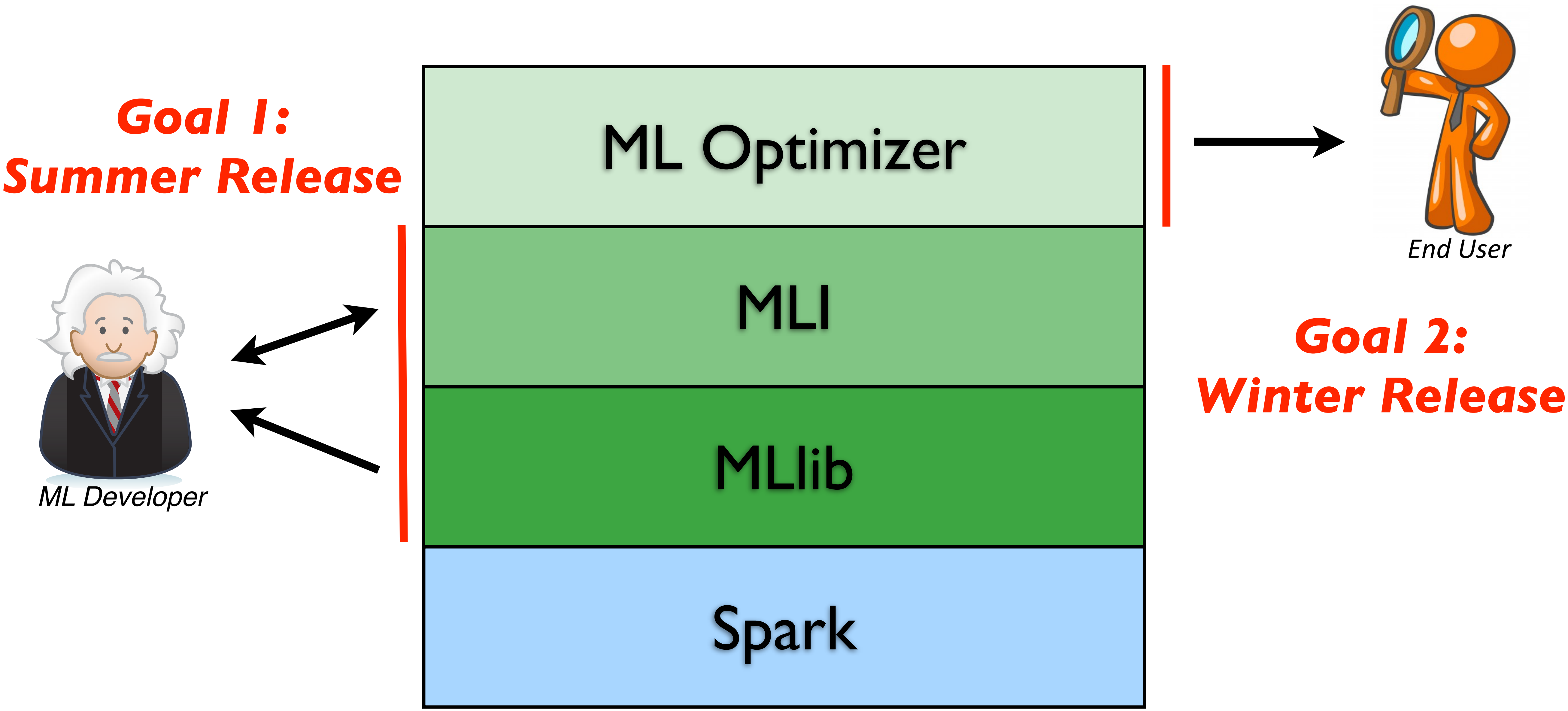
MLbase Stack Status



MLbase Stack Status



MLbase Stack Status



Future Directions

- **Identify minimal set of ML operators**
  - Expose internals of ML algorithms to optimizer
- Plug-ins to **Python, R**
- **Visualization** for unsupervised learning and exploration
- **Advanced ML capabilities**
  - Time-series algorithms
  - Graphical models
  - Advanced Optimization (e.g., asynchronous computation)
  - Online updates
  - Sampling for efficiency

Vision

MLI Details

Current Status

ML Workflow

Typical Data Analysis Workflow

Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow

Spark, MLI **Obtain / Load Raw Data**

Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow

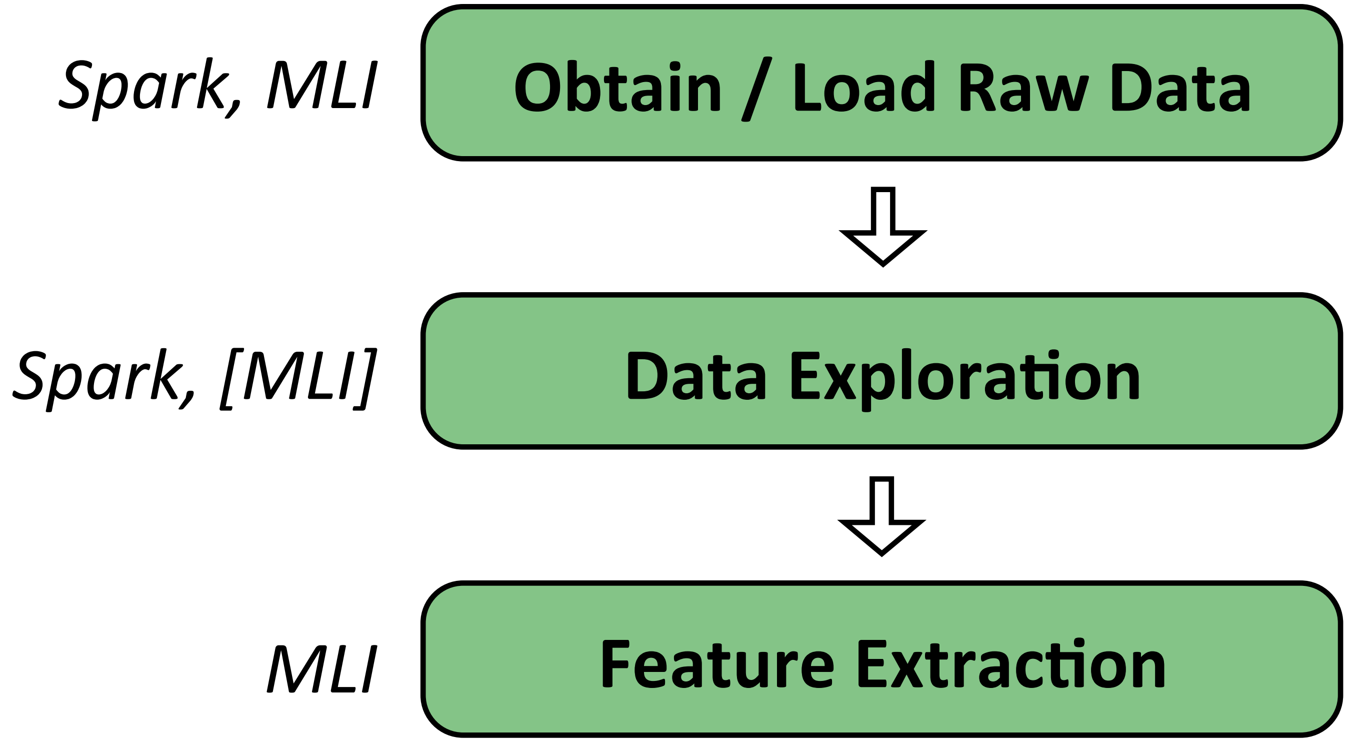
Spark, MLI — **Obtain / Load Raw Data**

⇩

Spark, [MLI] — **Data Exploration**

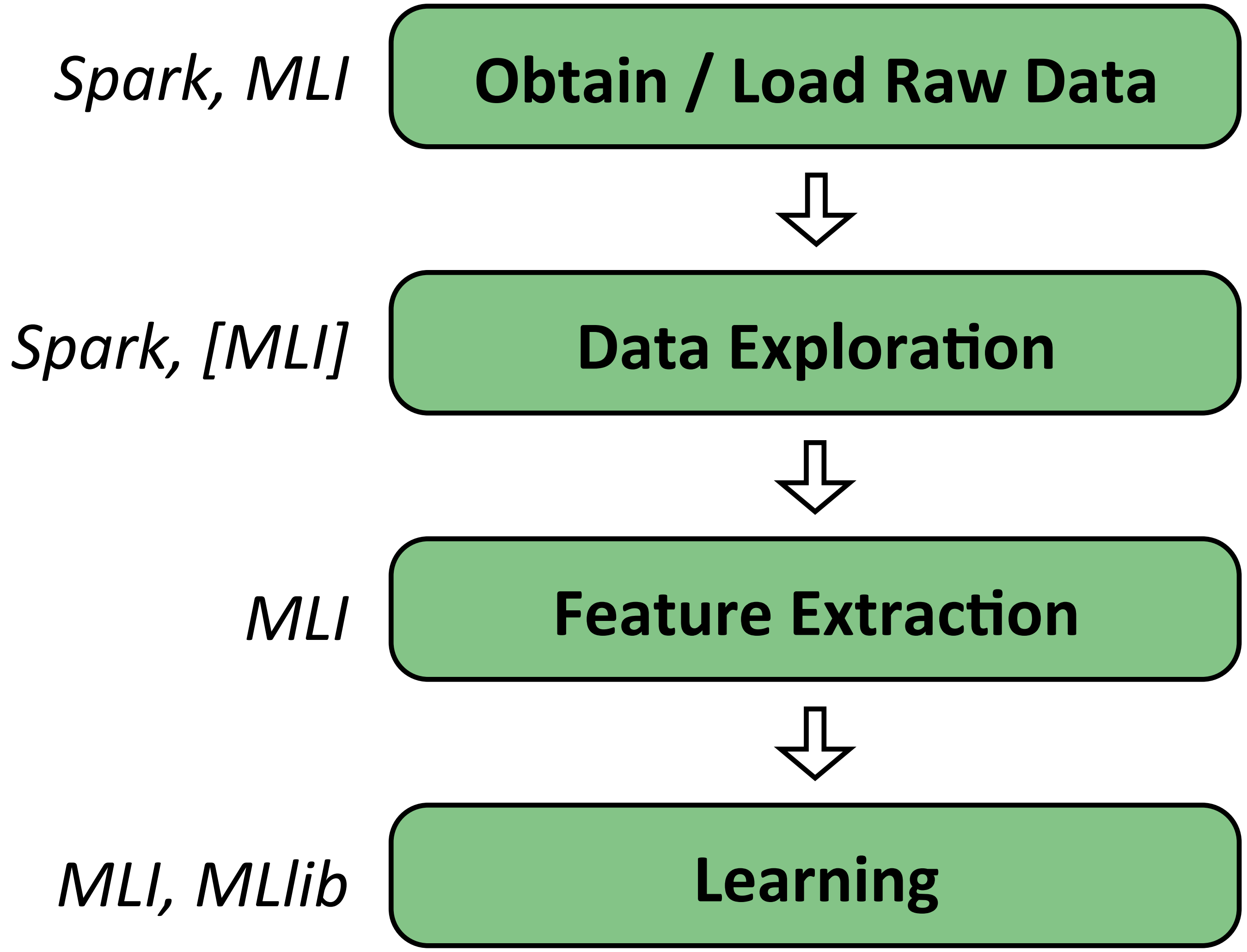
Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow



Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow



Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow

Tools	Step
Spark, MLI	**Obtain / Load Raw Data**
Spark, [MLI]	**Data Exploration**
MLI	**Feature Extraction**
MLI, MLlib	**Learning**
MLI	**Evaluation**

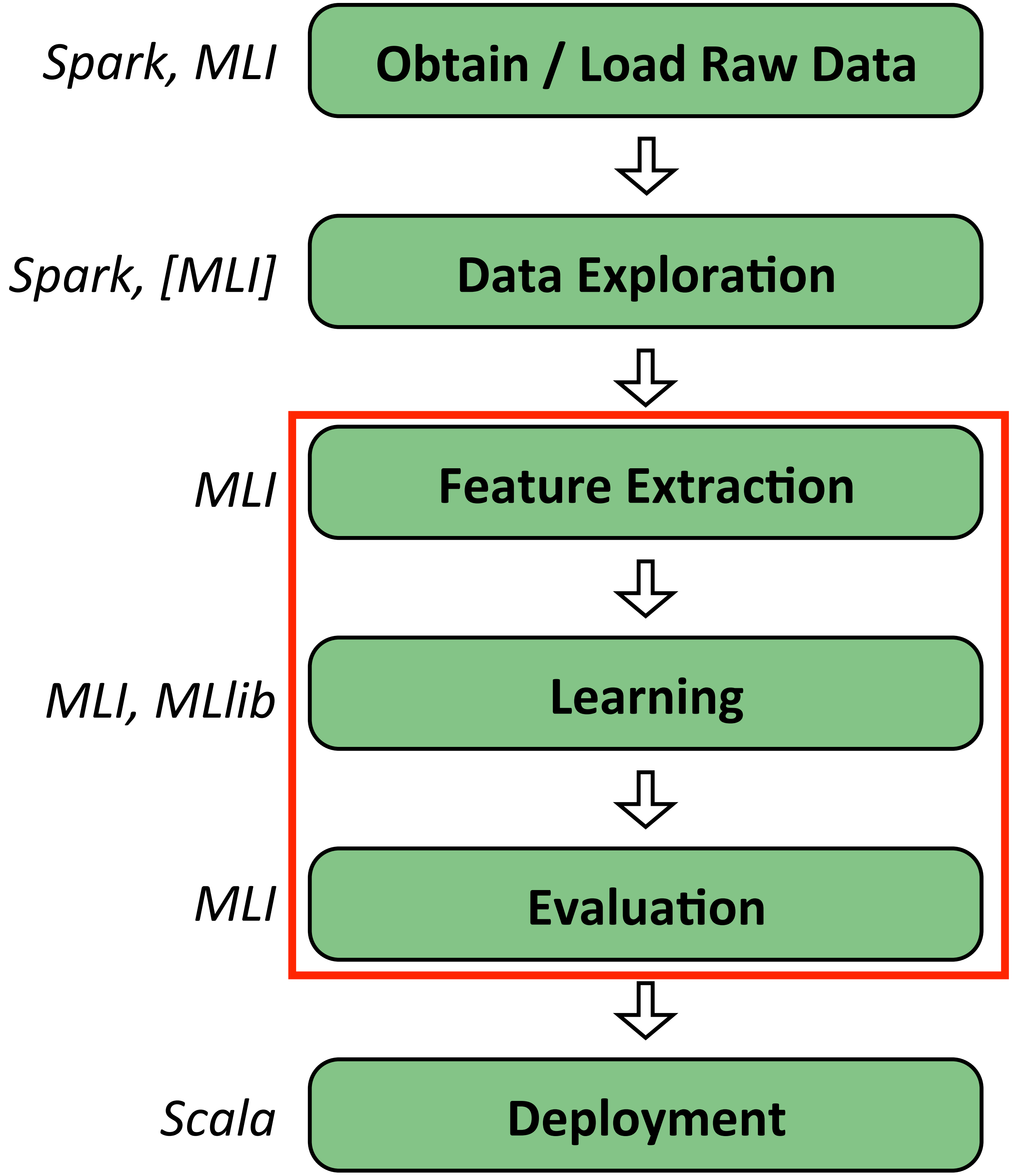
Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow

Spark, MLI	**Obtain / Load Raw Data**
Spark, [MLI]	**Data Exploration**
MLI	**Feature Extraction**
MLI, MLlib	**Learning**
MLI	**Evaluation**
Scala	**Deployment**

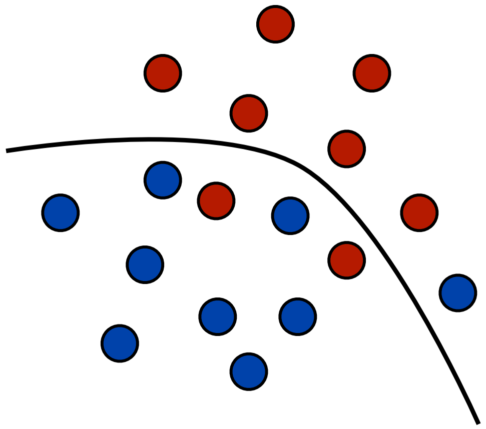
Adapted from slides by Ariel Kleiner

Typical Data Analysis Workflow



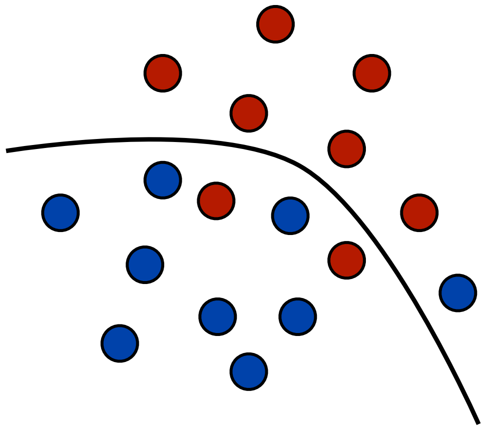
Adapted from slides by Ariel Kleiner

Binary Classification



Adapted from slides by Ariel Kleiner

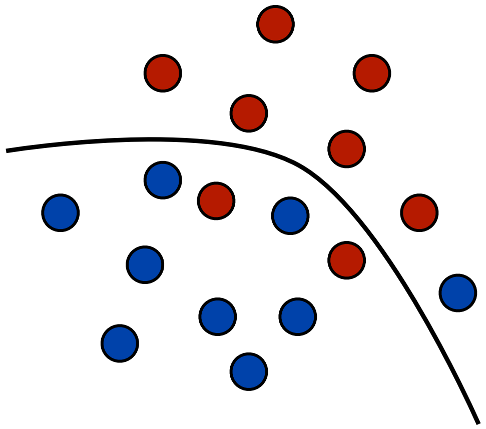
Binary Classification



Goal: Learn a mapping from entities to discrete labels

Adapted from slides by Ariel Kleiner

Binary Classification

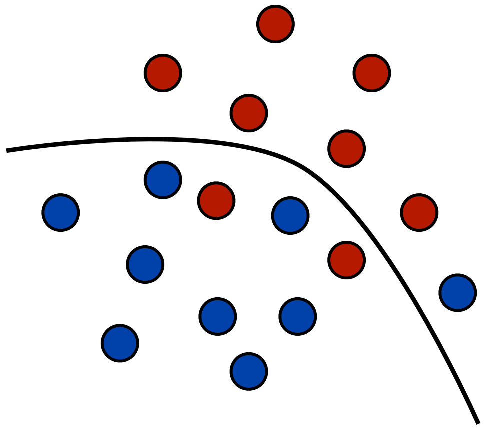


Goal: Learn a mapping from entities to discrete labels

Example: Spam Classification

- Entities are emails
- Labels are {*spam*, *not-spam*}

Binary Classification



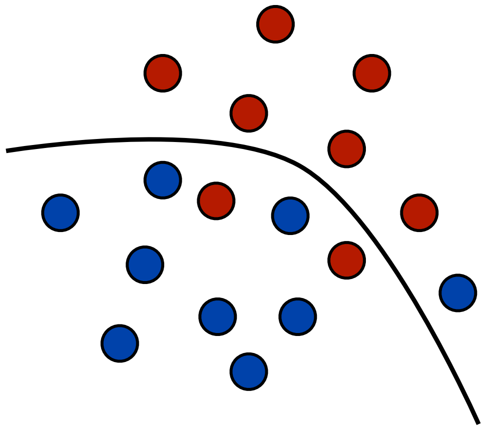
Goal: Learn a mapping from entities to discrete labels

Example: Spam Classification

- Entities are emails
- Labels are {*spam*, *not-spam*}
- Given past labeled emails, we want to predict whether a new email is *spam* or *not-spam*

Adapted from slides by Ariel Kleiner

Binary Classification

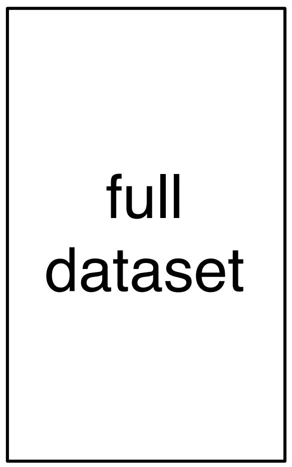


Goal: Learn a mapping from entities to discrete labels

Other Examples:

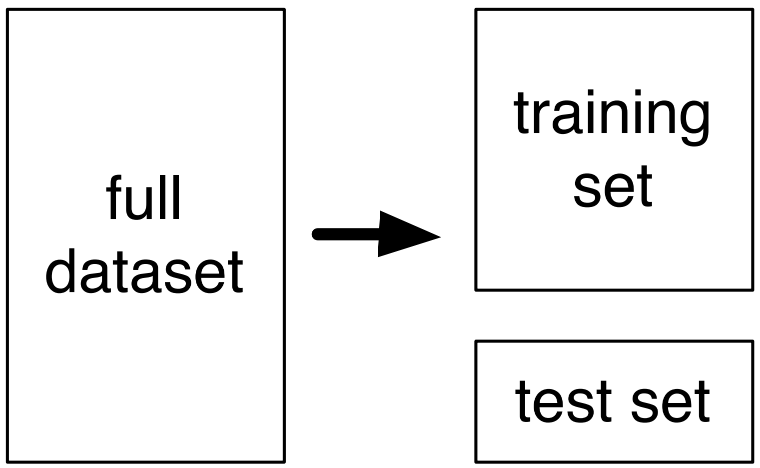
- Click (and clickthrough rate) prediction
- Fraud detection
- Face detection
- Exercise: "ARTS" vs "LIFE" on Wikipedia
  - Real data

Classification Pipeline



Adapted from slides by Ariel Kleiner

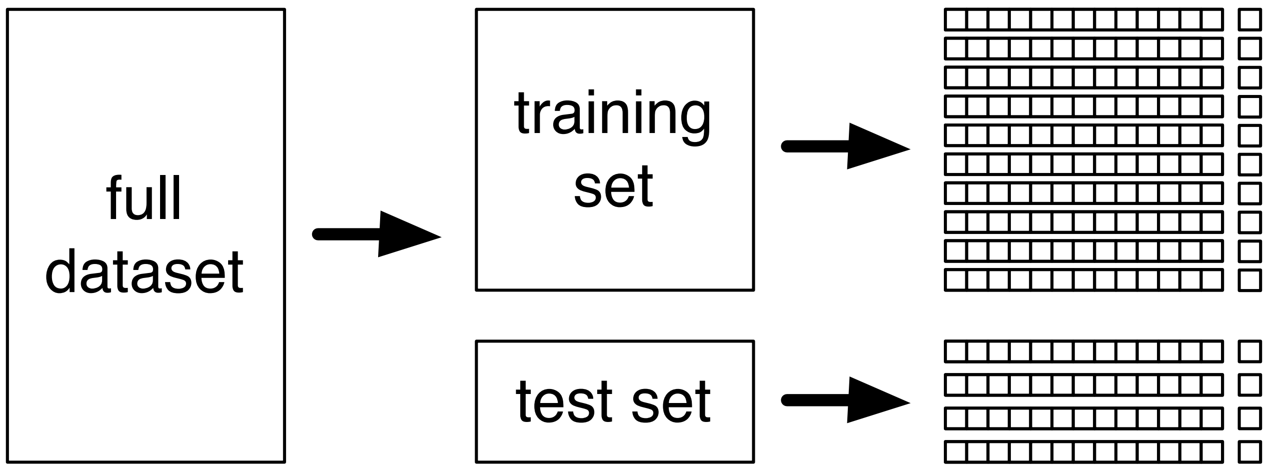
Classification Pipeline



1. Randomly split full data into disjoint subsets

Adapted from slides by Ariel Kleiner

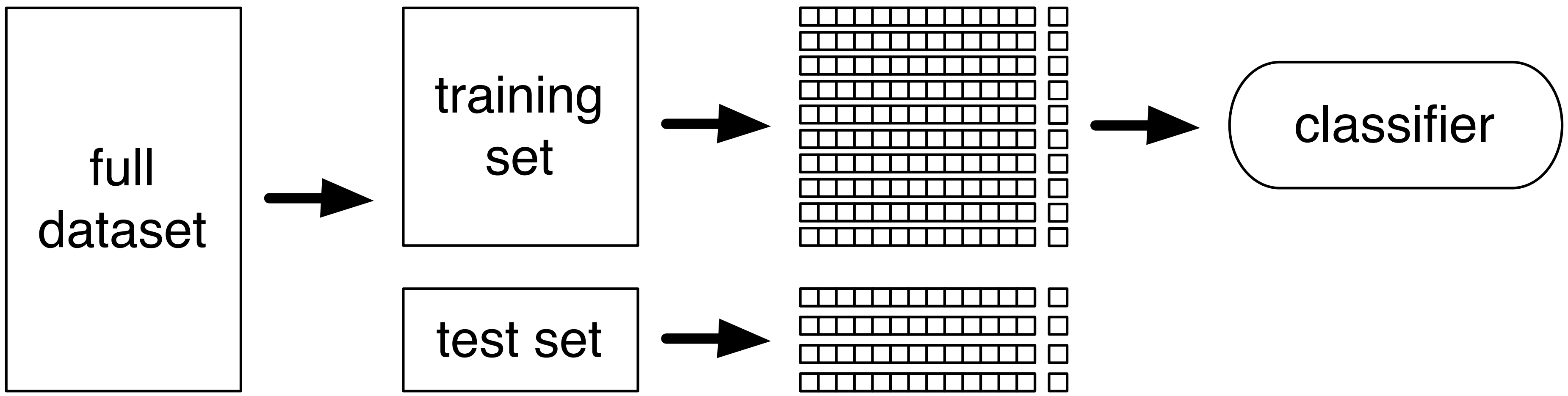
Classification Pipeline



1. Randomly split full data into disjoint subsets
2. Featurize the data

Adapted from slides by Ariel Kleiner

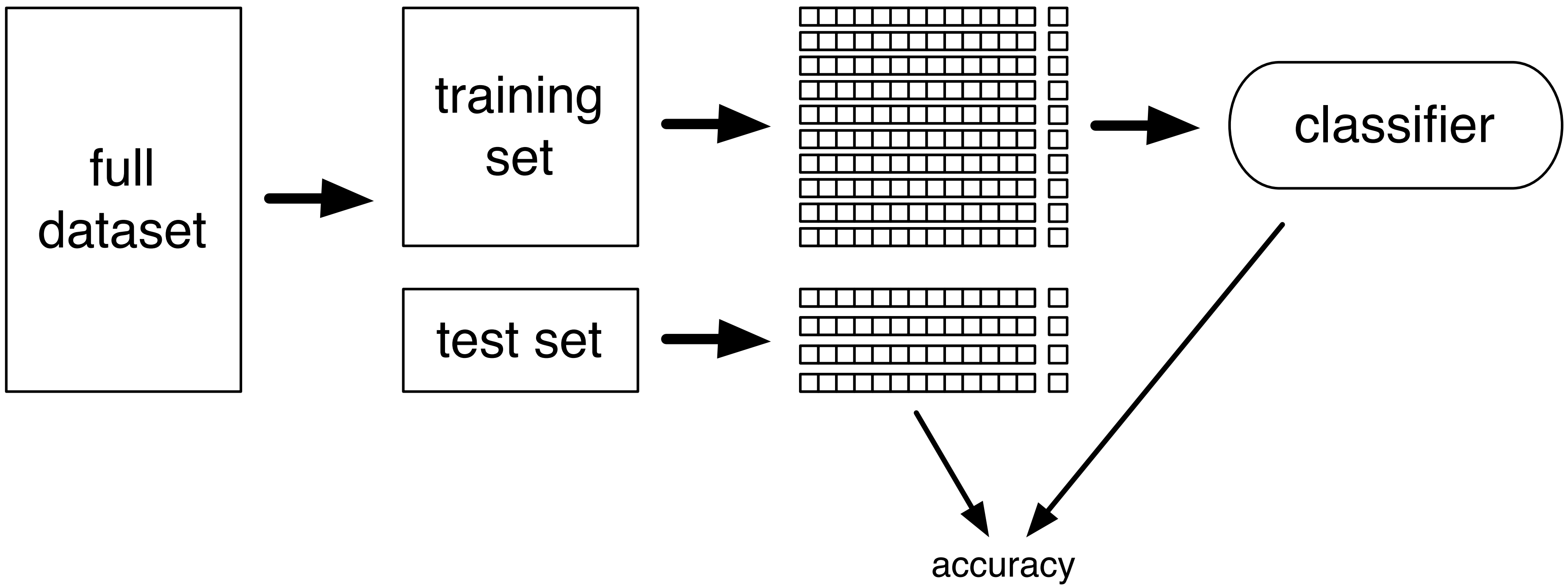
Classification Pipeline



1. Randomly split full data into disjoint subsets
2. Featurize the data
3. Use training set to learn a classifier

Adapted from slides by Ariel Kleiner

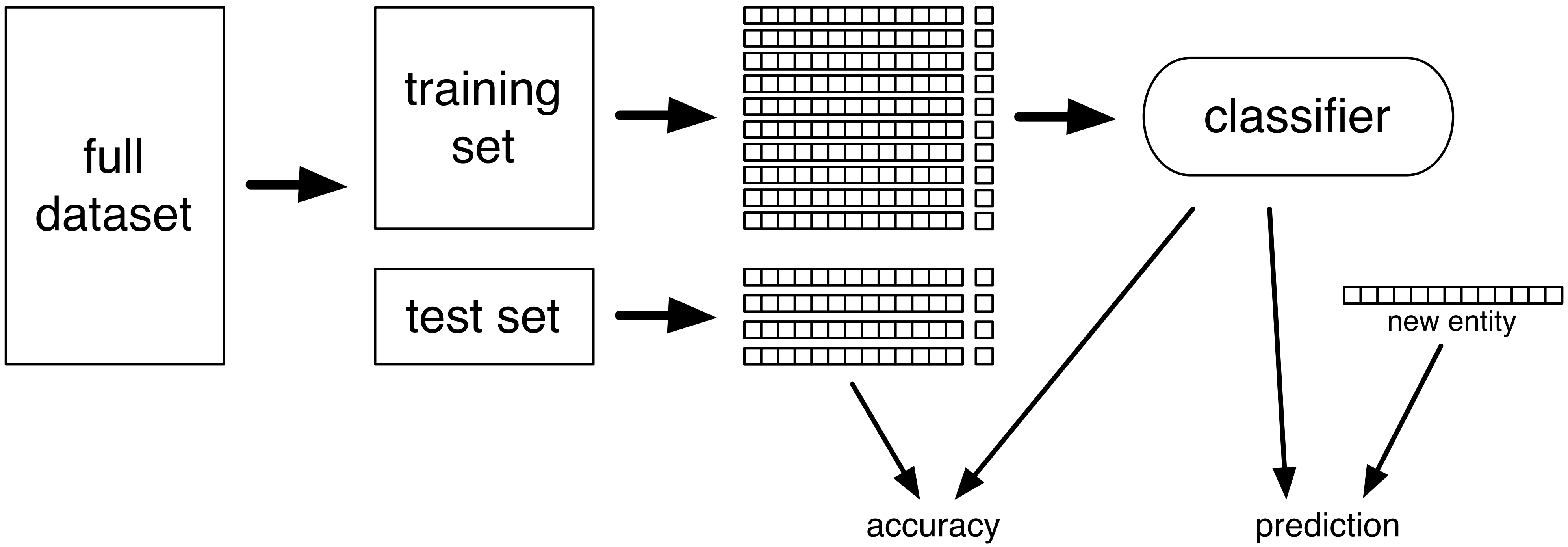
Classification Pipeline



1. Randomly split full data into disjoint subsets
2. Featurize the data
3. Use training set to learn a classifier
4. Evaluate classifier on test set (avoid overfitting)

Adapted from slides by Ariel Kleiner

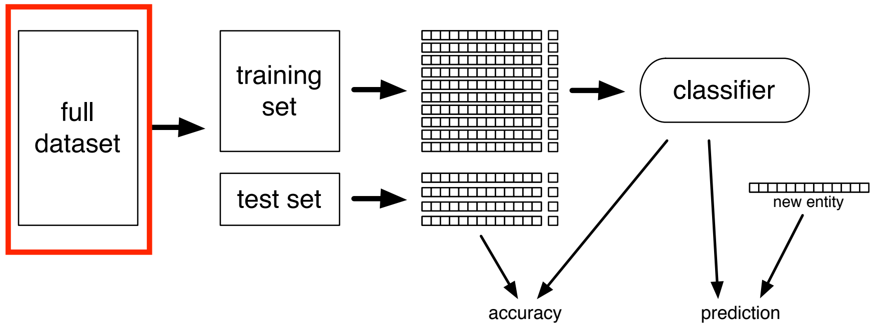
Classification Pipeline



1. Randomly split full data into disjoint subsets
2. Featurize the data
3. Use training set to learn a classifier
4. Evaluate classifier on test set (avoid overfitting)
5. Use classifier to predict in the wild

Adapted from slides by Ariel Kleiner

E.g., Spam Classification




```
From: illegitimate@bad.com

"Eliminate your debt by
giving us your money..."
```

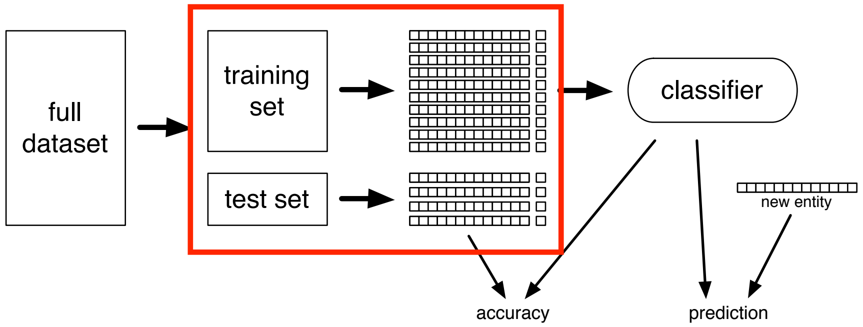
spam

```
From: bob@good.com

"Hi, it's been a while!
How are you? ..."
```

not-spam

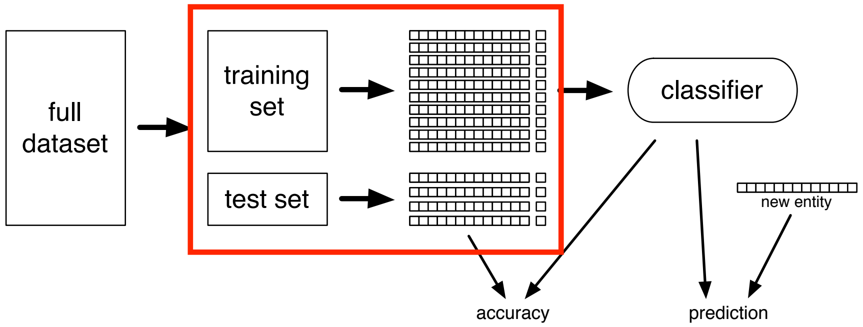
Adapted from slides by Ariel Kleiner

Featurization



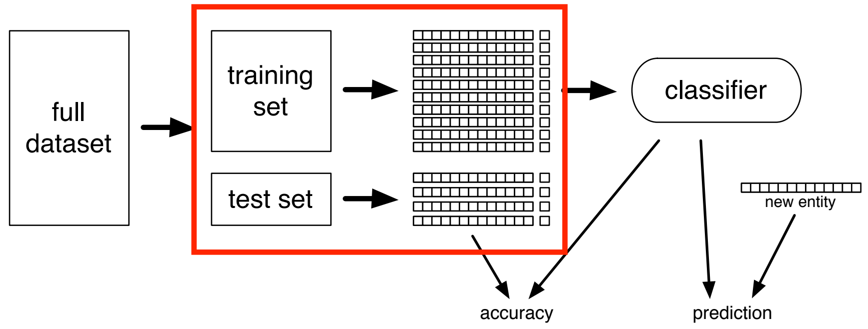
Adapted from slides by Ariel Kleiner

Featurization



- Most classifiers require numeric descriptions of entities

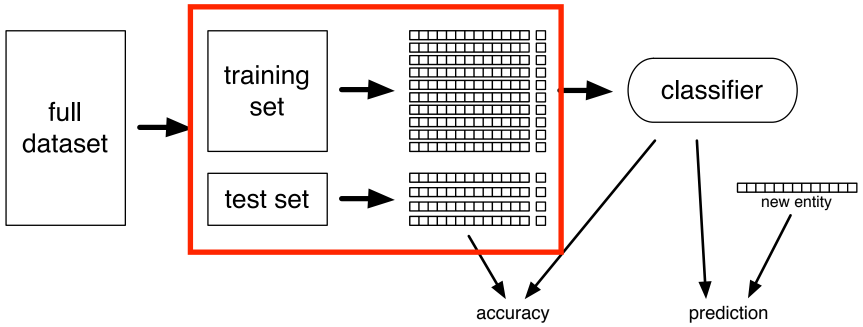
Featurization



- Most classifiers require numeric descriptions of entities

- **Featurization**: Transform each entity into a vector of real numbers

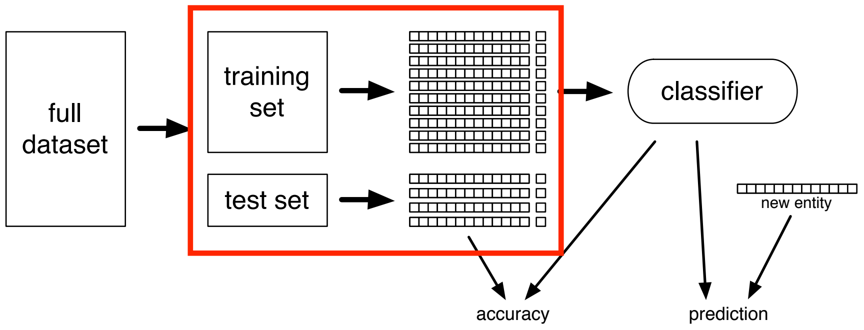
Featurization



- Most classifiers require numeric descriptions of entities

- **Featurization**: Transform each entity into a vector of real numbers
  - Opportunity to incorporate domain knowledge
  - Useful even when original data is already numeric

E.g., “Bag of Words”

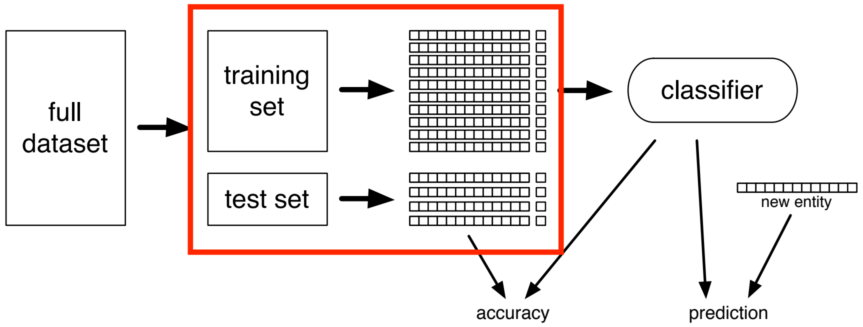



```
From: illegitimate@bad.com

"Eliminate your debt by
giving us your money..."
```

```
From: bob@good.com

"Hi, it's been a while!
How are you? ..."
```

Adapted from slides by Ariel Kleiner

E.g., “Bag of Words”

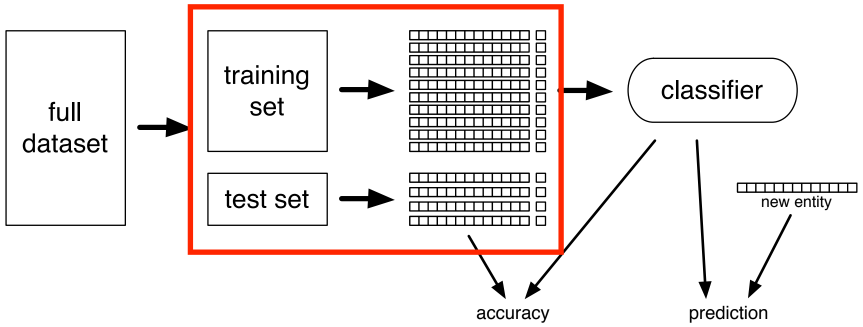


- Entities are documents

```
From: illegitimate@bad.com

"Eliminate your debt by
giving us your money..."
```

```
From: bob@good.com

"Hi, it's been a while!
How are you? ..."
```

E.g., “Bag of Words”



- Entities are documents
- Build Vocabulary

```
From: illegitimate@bad.com

"Eliminate your debt by
giving us your money..."
```

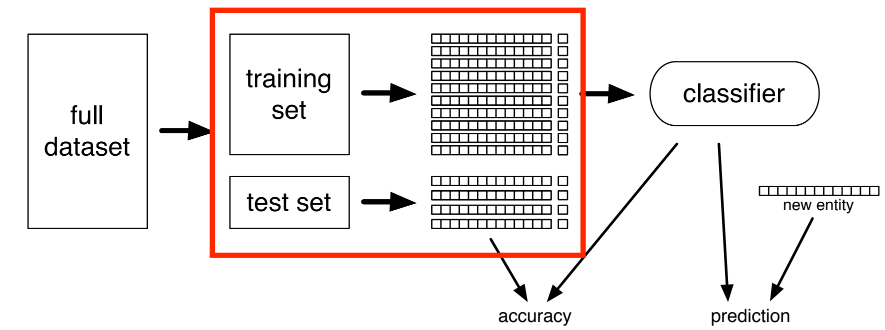
```
From: bob@good.com

"Hi, it's been a while!
How are you? ..."
```

Vocabulary

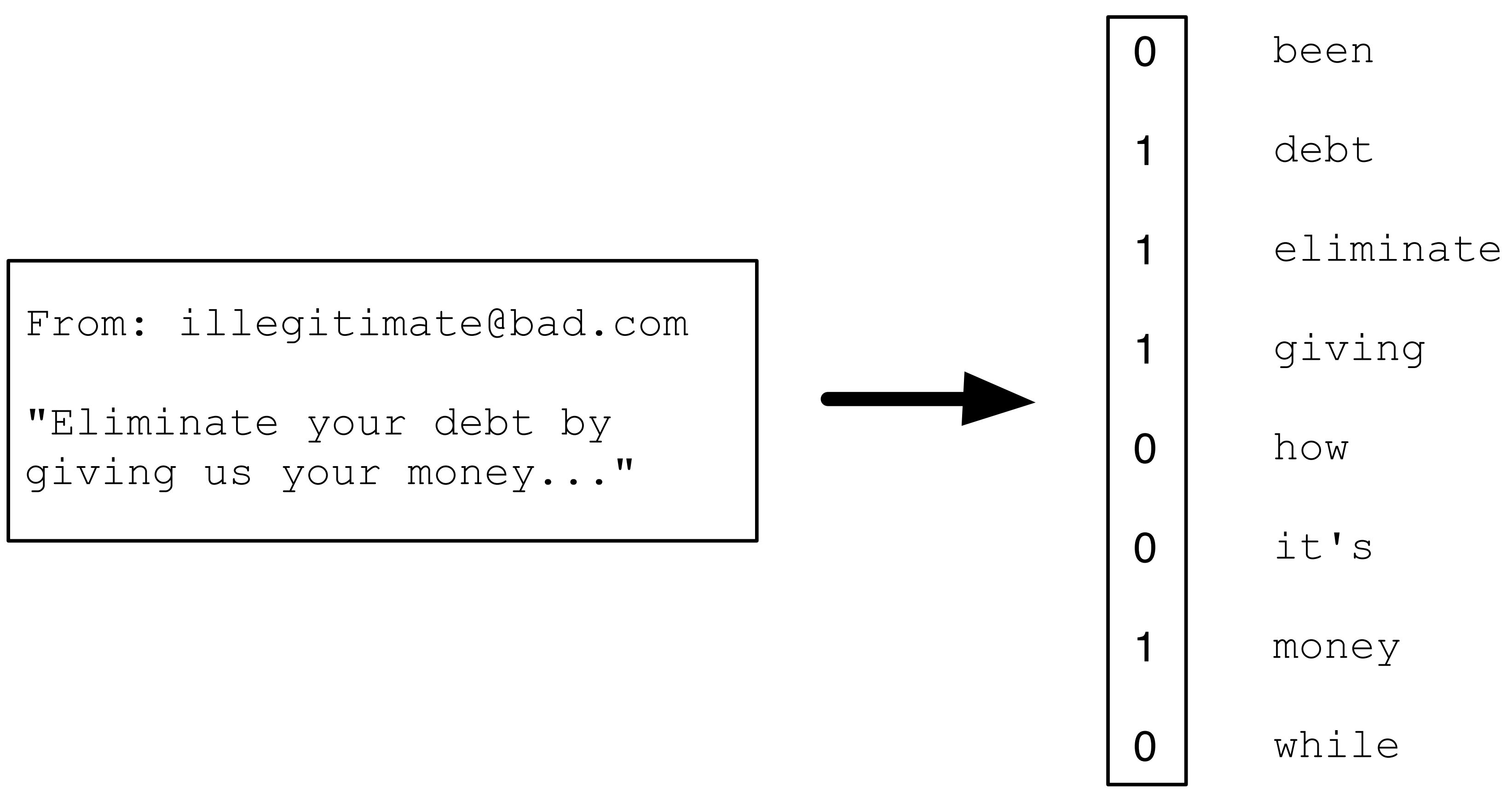
```
been
debt
eliminate
giving
how
it's
money
while
```

Adapted from slides by Ariel Kleiner

E.g., "Bag of Words"

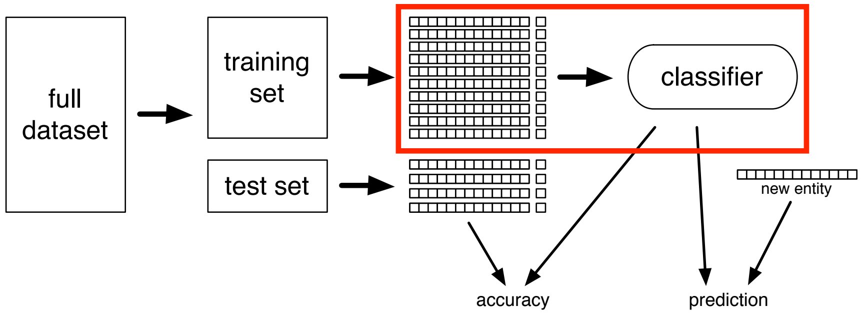


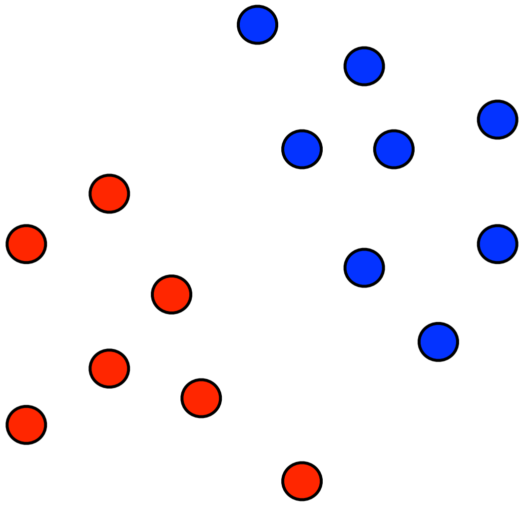
- Entities are documents
- Build Vocabulary
- Derive feature vectors from Vocabulary
  - Exercise: we'll use *bigrams*



Adapted from slides by Ariel Kleiner

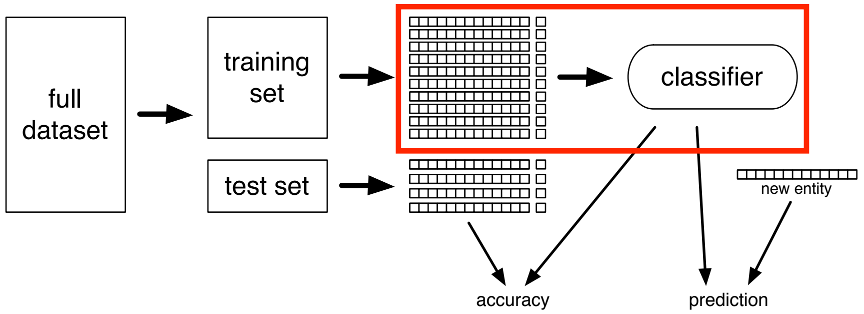
Support Vector Machines (SVMs)

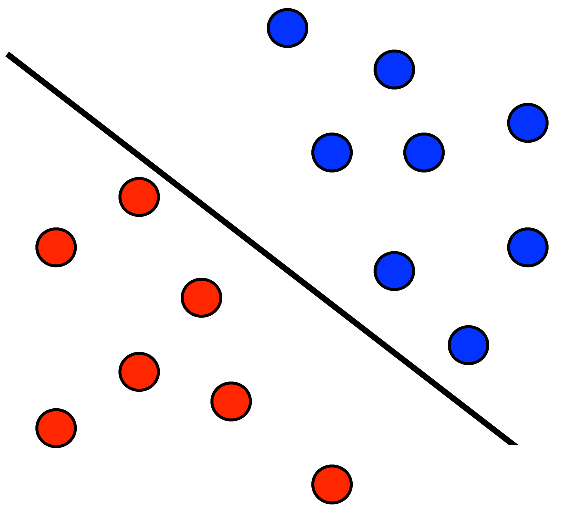




Credit: Foundations of Machine Learning
Mohri, Rostamizadeh, Talwalkar

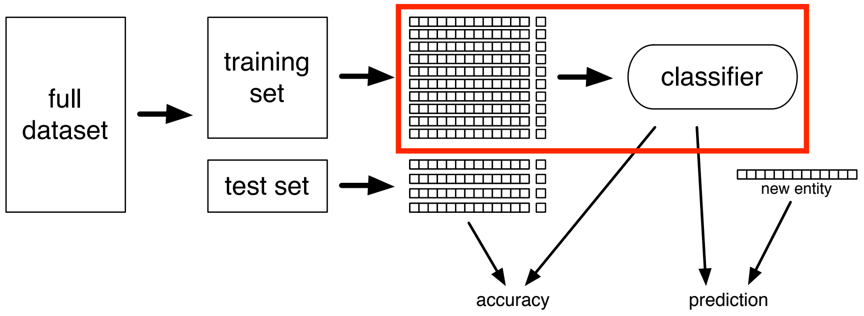
Support Vector Machines (SVMs)

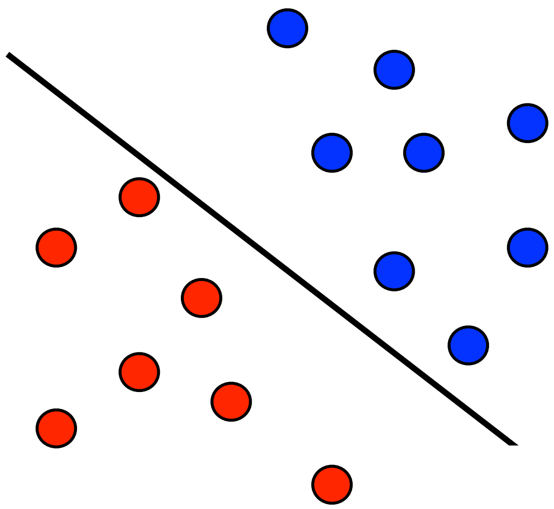


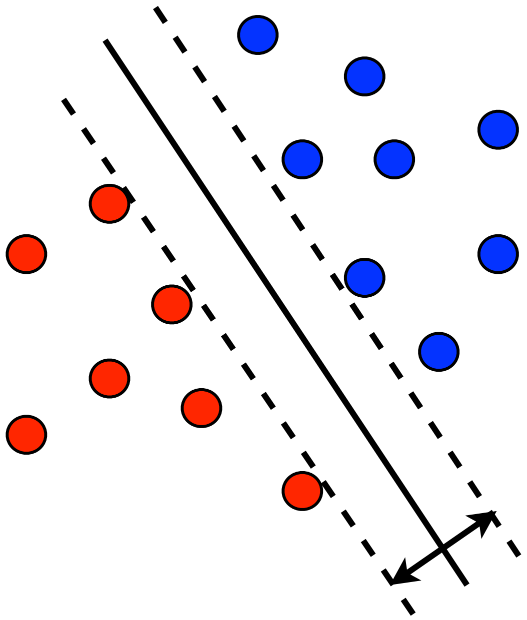


Credit: Foundations of Machine Learning
Mohri, Rostamizadeh, Talwalkar

Support Vector Machines (SVMs)

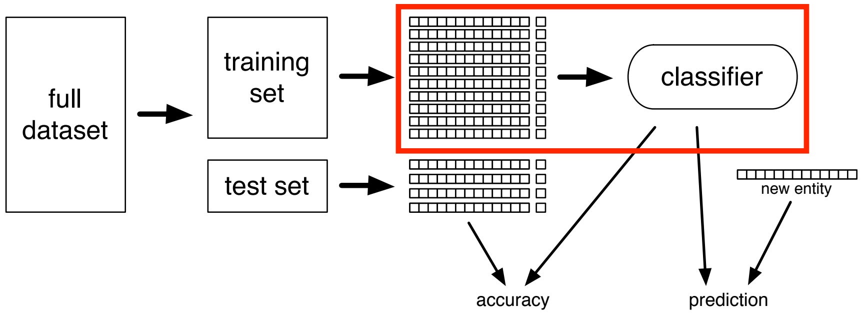




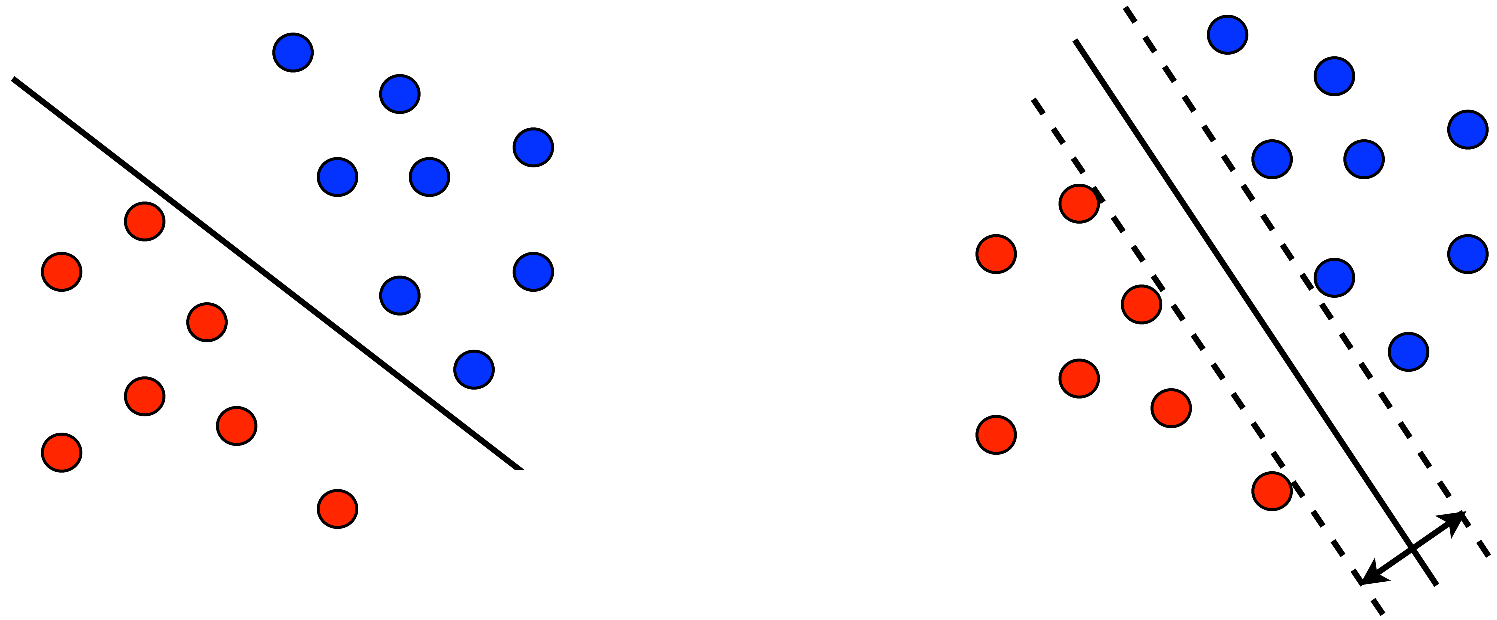


Credit: Foundations of Machine Learning
Mohri, Rostamizadeh, Talwalkar

Support Vector Machines (SVMs)

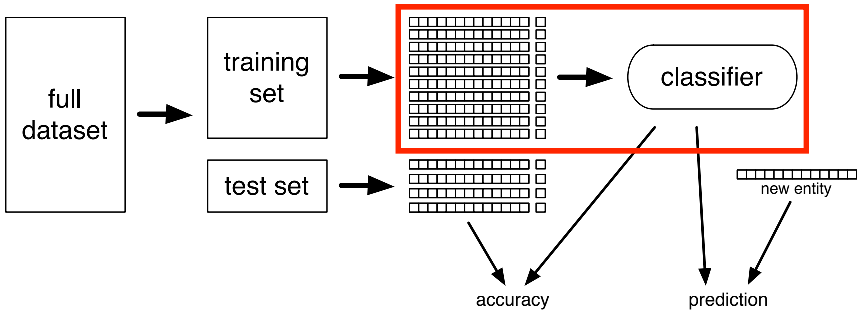


- "Max-Margin": find linear separator with the largest separation between the two classes

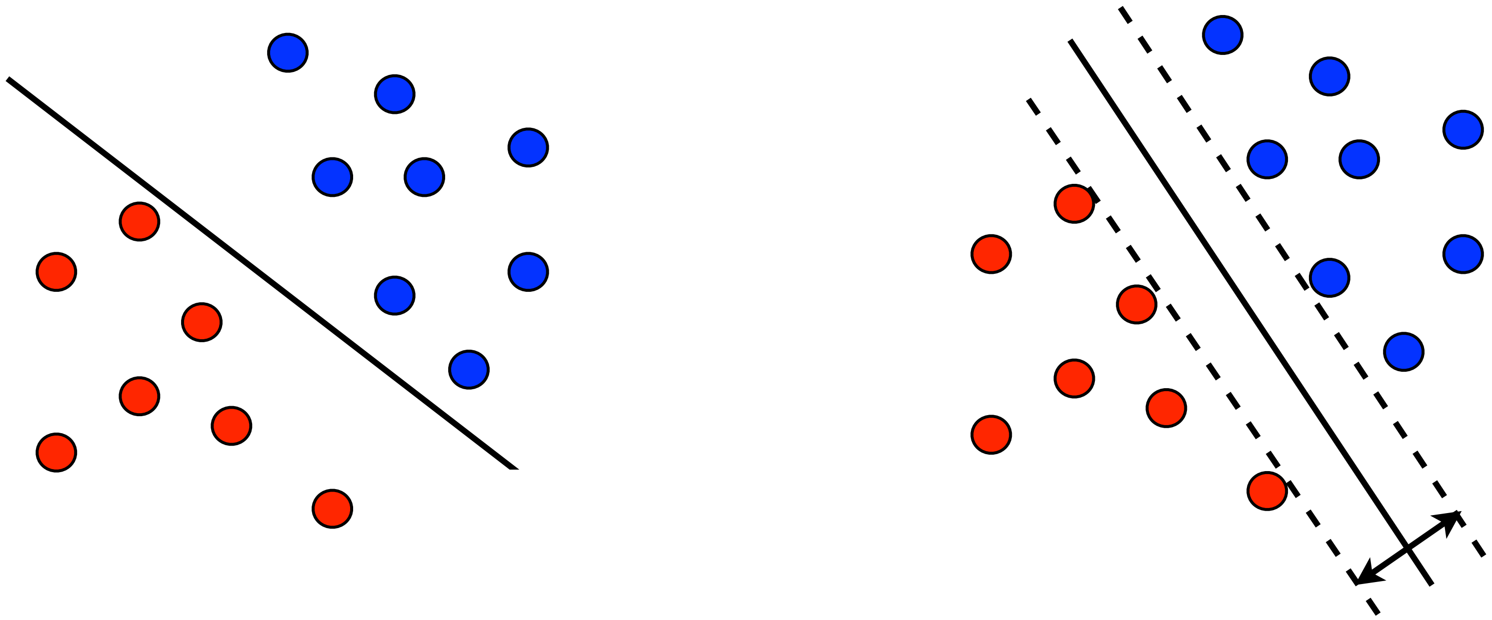


Credit: Foundations of Machine Learning
Mohri, Rostamizadeh, Talwalkar

Support Vector Machines (SVMs)

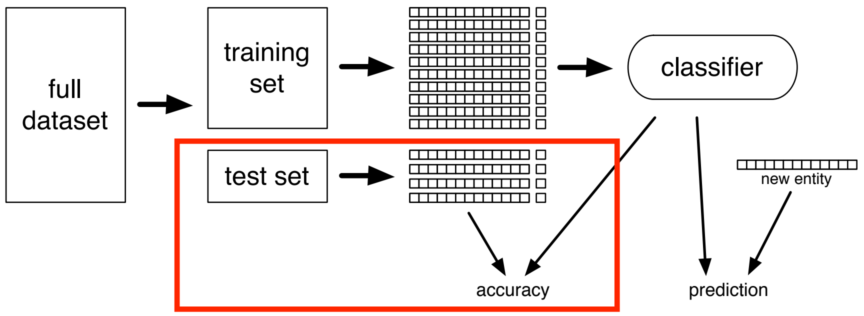


- "Max-Margin": find linear separator with the largest separation between the two classes
- Extensions:
  - non-separable setting
  - non-linear classifiers (kernels)



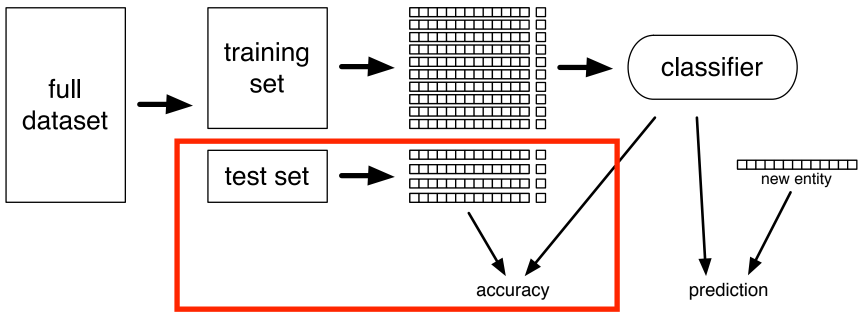
Credit: Foundations of Machine Learning
Mohri, Rostamizadeh, Talwalkar

Model Evaluation



Adapted from slides by Ariel Kleiner

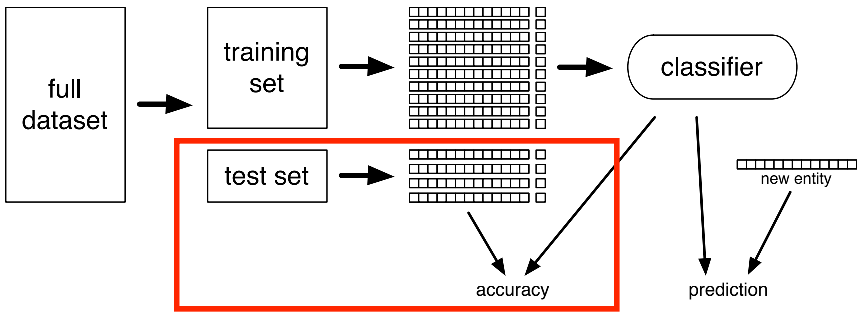
Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - "Overfitting"; "Generalization"

Adapted from slides by Ariel Kleiner

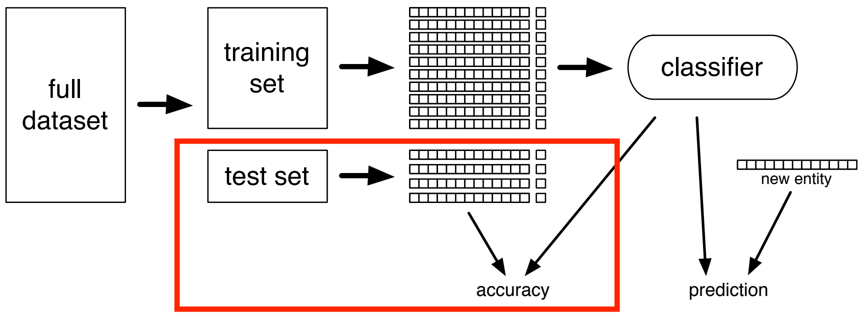
Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - "Overfitting"; "Generalization"
- Various metrics for quality; accuracy is most common

Adapted from slides by Ariel Kleiner

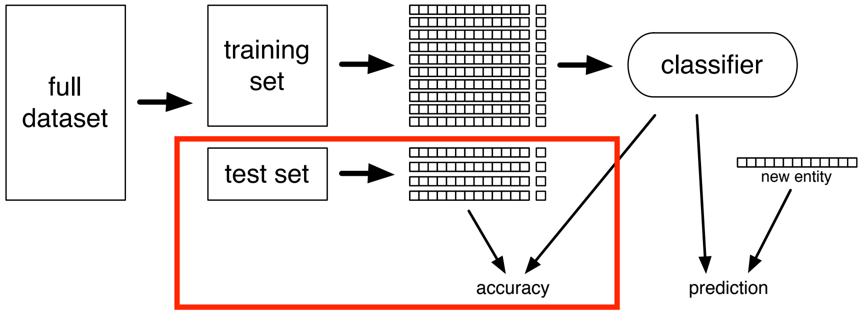
Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - “Overfitting”; “Generalization”
- Various metrics for quality; accuracy is most common
- Evaluation process

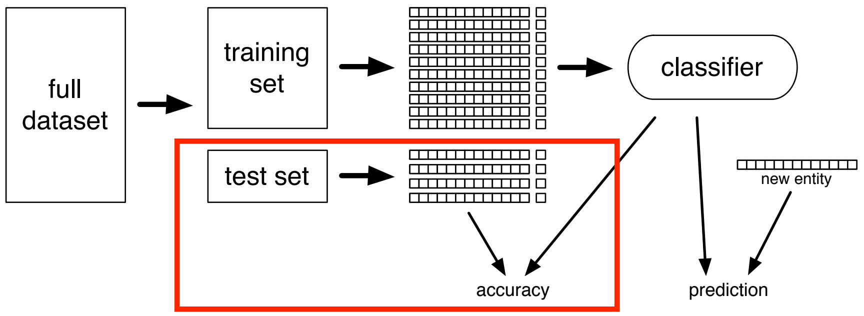
Adapted from slides by Ariel Kleiner

Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - "Overfitting"; "Generalization"
- Various metrics for quality; accuracy is most common
- Evaluation process
  - Train on training set (don't expose test set to classifier)

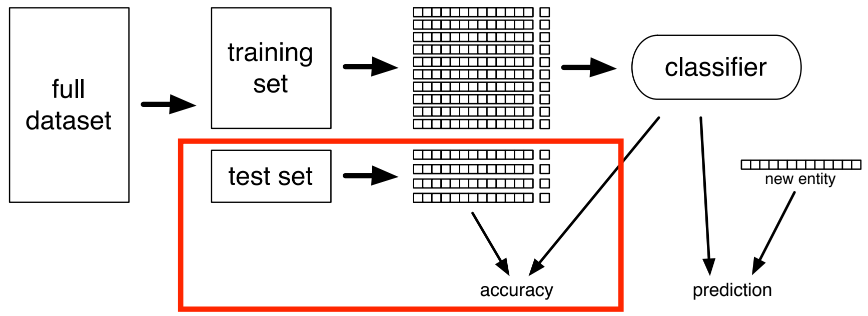
Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - “Overfitting”; “Generalization”
- Various metrics for quality; accuracy is most common
- Evaluation process
  - Train on training set (don’t expose test set to classifier)
  - Make predictions using test set (ignoring test labels)

Adapted from slides by Ariel Kleiner

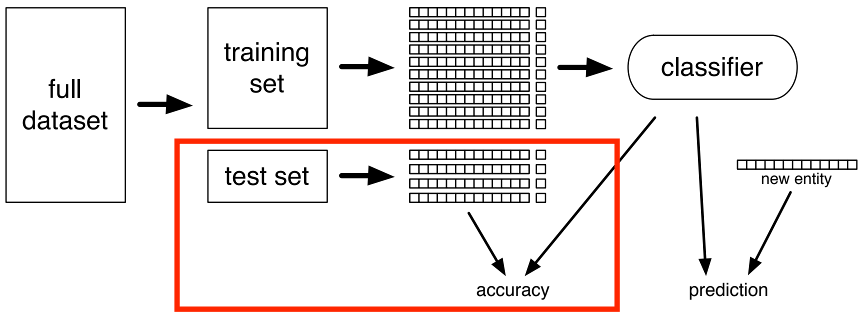
Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - "Overfitting"; "Generalization"
- Various metrics for quality; accuracy is most common
- Evaluation process
  - Train on training set (don't expose test set to classifier)
  - Make predictions using test set (ignoring test labels)
  - Compute fraction of correct predictions on test set

Adapted from slides by Ariel Kleiner

Model Evaluation



- Test set simulates performance on new entity
  - Performance on training data overly optimistic!
  - "Overfitting"; "Generalization"
- Various metrics for quality; accuracy is most common
- Evaluation process
  - Train on training set (don't expose test set to classifier)
  - Make predictions using test set (ignoring test labels)
  - Compute fraction of correct predictions on test set
- Other more sophisticated evaluation methods, e.g., cross-validation

Adapted from slides by Ariel Kleiner

Contributions encouraged!

MLbase

www.mlbase.org

SIMONS INSTITUTE for the Theory of Computing

biglearn.org