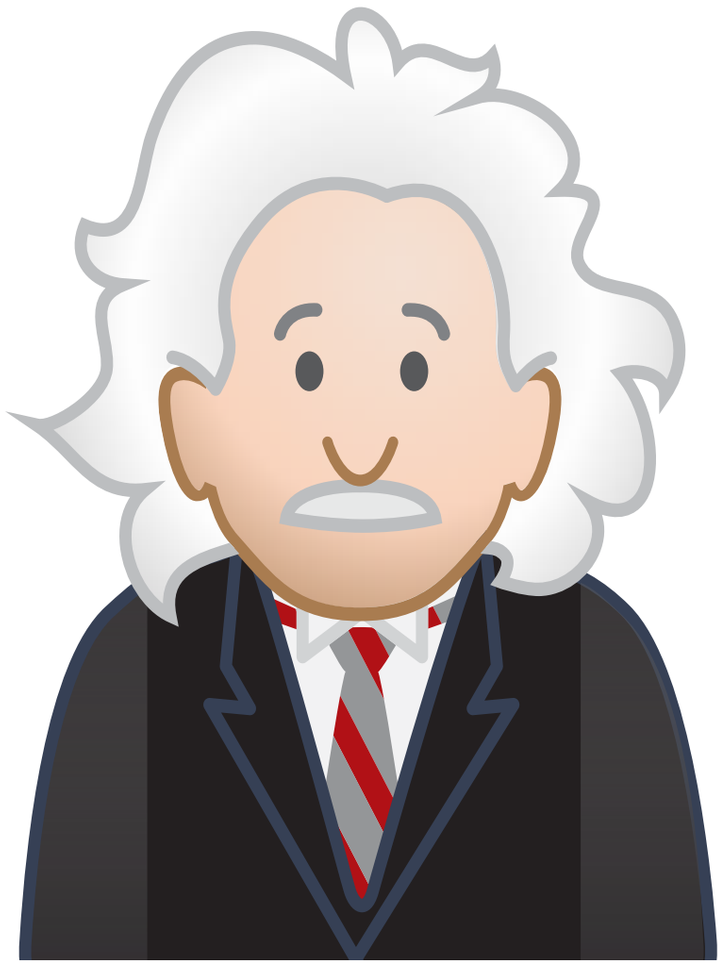
Towards an Optimizer for MLbase

Ameet Talwalkar

Collaborators: Evan Sparks, Michael Franklin, Michael I. Jordan, Tim Kraska

Problem: Scalable implementations *difficult* for ML Developers…



Problem: Scalable implementations *difficult* for ML Developers…

DATABRICKS

Problem: Scalable implementations *difficult* for ML Developers…

VOWPAL WABBIT

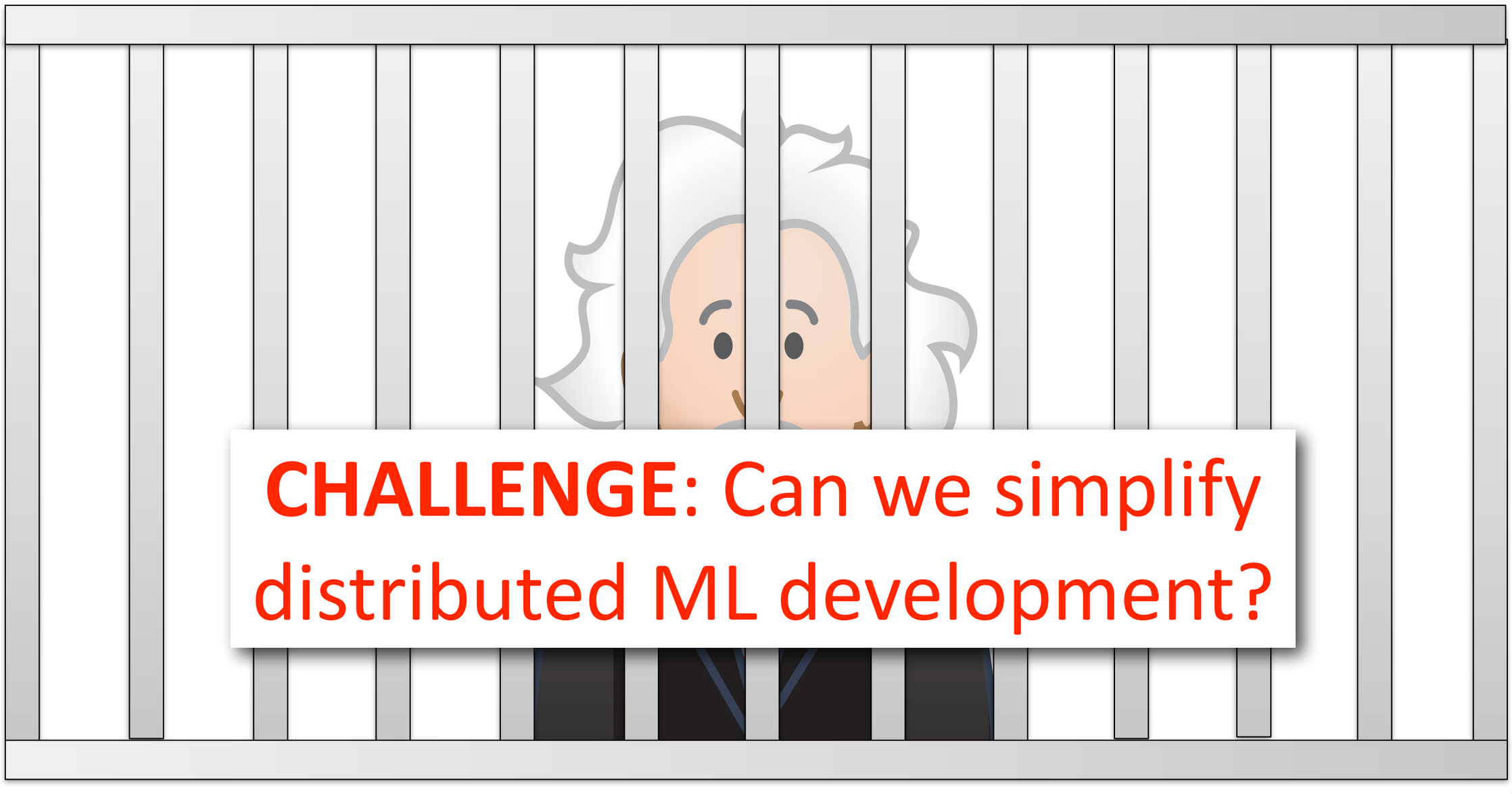
mahout

DATABRICKS

Problem: Scalable implementations *difficult* for ML Developers…

VOWPAL WABBIT

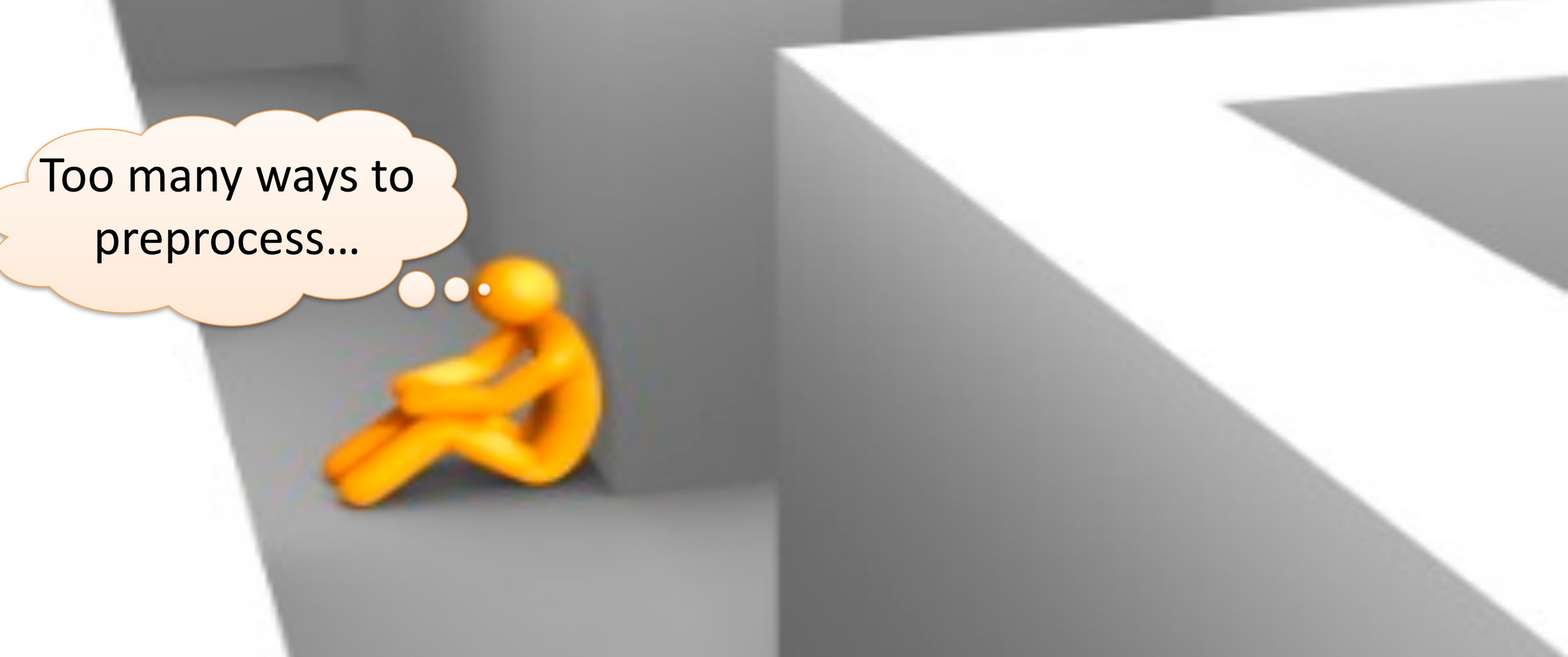
mahout



CHALLENGE: Can we simplify distributed ML development?

Problem: ML is *difficult* for End Users…

Problem: ML is *difficult* for End Users…



Problem: ML is *difficult* for End Users…

Too many algorithms…

Too many ways to preprocess…

Problem: ML is *difficult* for End Users…

Too many algorithms…

Too many knobs…

Too many ways to preprocess…

Problem: ML is *difficult* for End Users…

Too many algorithms…

Too many knobs…

Too many ways to preprocess…

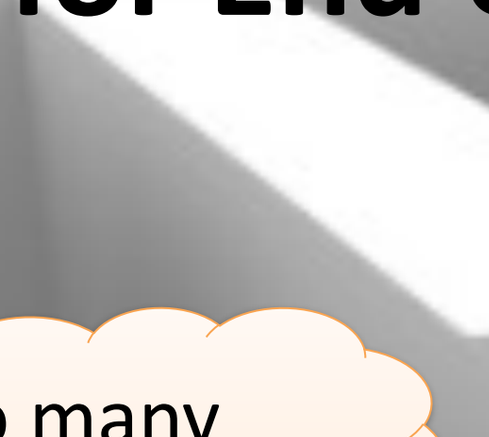
Difficult to debug…

Problem: ML is *difficult* for End Users…

Too many algorithms…

Too many knobs…

Too many ways to preprocess…



Difficult to debug…

Doesn’t scale…

Problem: ML is *difficult* for End Users…

Too many algorithms…

Too many knobs…

Too many ways to preprocess…

Difficult to debug

CHALLENGE: Can we automate ML pipeline construction?

MLbase

MLbase aims to simplify development and deployment of ML pipelines

MLbase

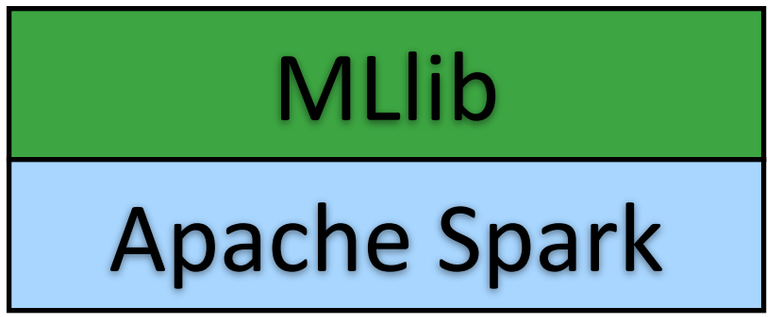
MLbase aims to simplify development and deployment of ML pipelines

Apache Spark

Spark: Cluster computing system designed for iterative computation

MLbase

MLbase aims to simplify development and deployment of ML pipelines

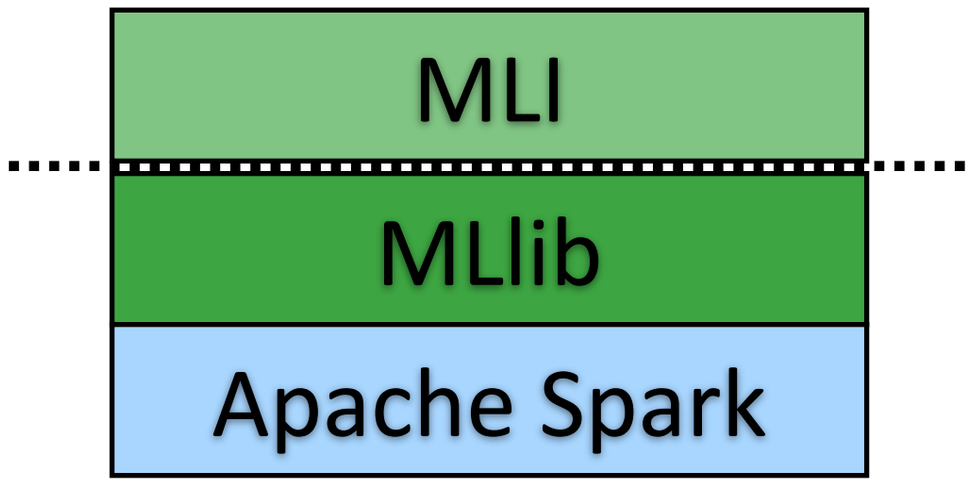


Spark: Cluster computing system designed for iterative computation

MLlib: Spark's core ML library

MLbase

MLbase aims to simplify development and deployment of ML pipelines



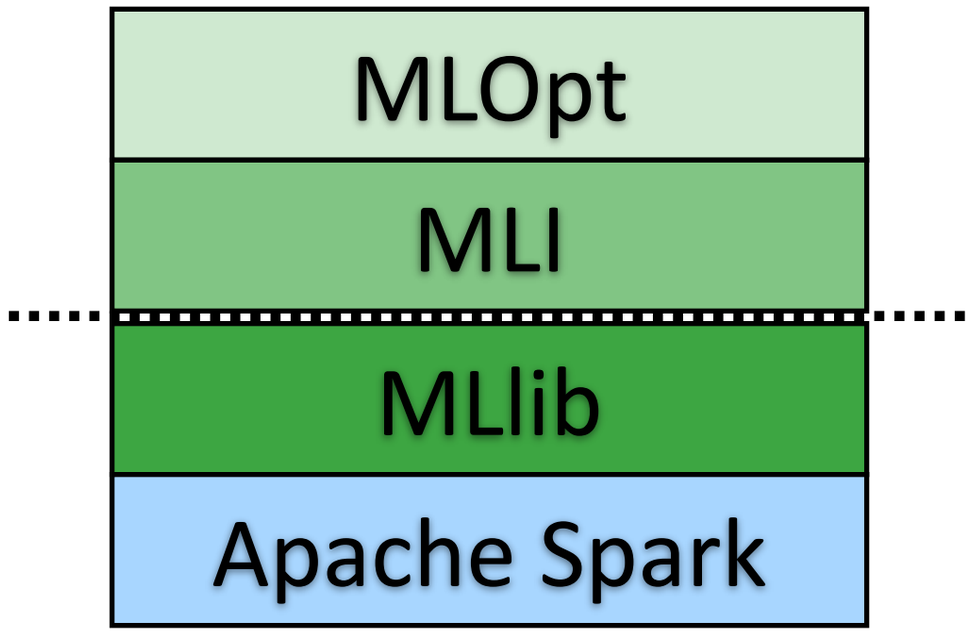
Spark: Cluster computing system designed for iterative computation

MLlib: Spark's core ML library

MLI: API to simplify ML development

MLbase

MLbase aims to simplify development and deployment of ML pipelines



Spark: Cluster computing system designed for iterative computation

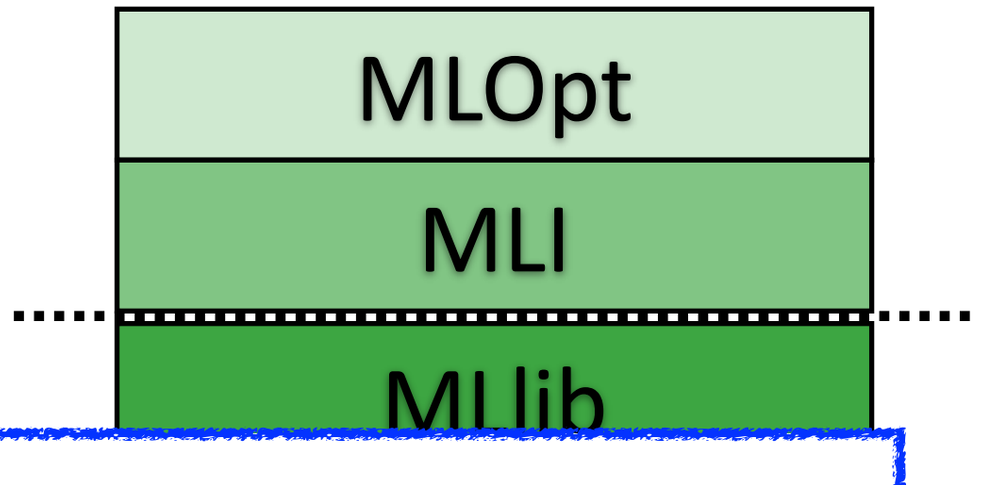
MLlib: Spark's core ML library

MLI: API to simplify ML development

MLOpt: Declarative layer that aims to automate ML pipeline construction via search over feature extractors and models

MLbase

MLbase aims to simplify development and deployment of ML



MLOpt and MLI are experimental testbeds

Spark: ...ation

MLlib: Spark's core ML library

MLI: API to simplify ML development

MLOpt: Declarative layer that aims to automate ML pipeline construction via search over feature extractors and models

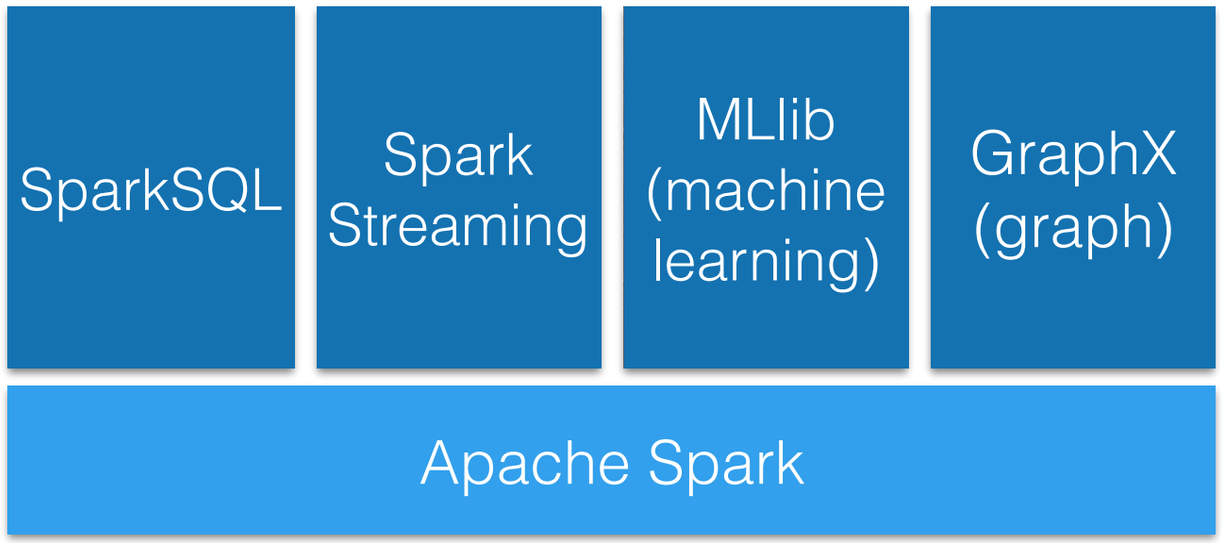
Vision

MLlib and MLI

MLOpt

MLlib

+ Scalable and fast
+ Simple development environment
+ Part of Spark's robust ecosystem



Active Development

Active Development

Initial Release

- Developed by MLbase team in AMPLab (11 contributors)
- Scala, Java
- Shipped with Spark v0.8 (Sep 2013)

Active Development

Initial Release

- Developed by MLbase team in AMPLab (11 contributors)
- Scala, Java
- Shipped with Spark v0.8 (Sep 2013)

11 months later…

- 55+ contributors from various organizations
- Scala, Java, Python
- Improved documentation / code examples, API stability
- Latest release part of Spark v1.0 (May 2014)

Algorithms in v0.8

- **classification**: logistic regression, linear support vector machines (SVM)
- **regression**: linear regression,
- **collaborative filtering**: alternating least squares (ALS)
- **clustering**: k-means
- **optimization**: stochastic gradient descent (SGD)

Algorithms in v1.0

- **classification**: logistic regression, linear support vector machines (SVM), naive Bayes, decision trees
- **regression**: linear regression, regression trees
- **collaborative filtering**: alternating least squares (ALS)
- **clustering**: k-means
- **optimization**: stochastic gradient descent (SGD), limited-memory BFGS (L-BFGS)
- **dimensionality reduction**: singular value decomposition (SVD), principal component analysis (PCA)

MLlib, MLI and Roadmap

MLlib, MLI and Roadmap

- MLI: Shield ML Developers from low-details
  - Provide familiar mathematical operators in distributed setting (tables, matrices, optimization primitives)
  - Standard APIs defining ML algorithms and feature extractors

MLlib, MLI and Roadmap

- MLI: Shield ML Developers from low-details
  - Provide familiar mathematical operators in distributed setting (tables, matrices, optimization primitives)
  - Standard APIs defining ML algorithms and feature extractors
- Many of these ideas are (or soon will be) in MLlib

MLlib, MLI and Roadmap

- MLI: Shield ML Developers from low-details
  - Provide familiar mathematical operators in distributed setting (tables, matrices, optimization primitives)
  - Standard APIs defining ML algorithms and feature extractors

- Many of these ideas are (or soon will be) in MLlib

- Next release of Spark and MLlib being tested now
  - statistical toolbox, python decision tree API, online logistic regression, …

MLlib, MLI and Roadmap

- MLI: Shield ML Developers from low-details
  - Provide familiar mathematical operators in distributed setting (tables, matrices, optimization primitives)
  - Standard APIs defining ML algorithms and feature extractors

- Many of these ideas are (or soon will be) in MLlib

- Next release of Spark and MLlib being tested now
  - statistical toolbox, python decision tree API, online logistic regression, …

- Longer term
  - Scalable implementations of standard ML algorithms and underlying optimization primitives
  - Support for ML pipeline development (related to MLOpt)

MLlib, MLI and Roadmap

- MLI: Shield ML Developers from low-details
  - Provide familiar mathematical operators in distributed setting (tables, matrices, optimization primitives)
  - Standard APIs defining ML algorithms and feature extractors
- Many of these ideas are (or soon will be) in MLlib

Feedback and Contributions Encouraged!

- Longer term
  - Scalable implementations of standard ML algorithms and underlying optimization primitives
  - Support for ML pipeline development (related to MLOpt)

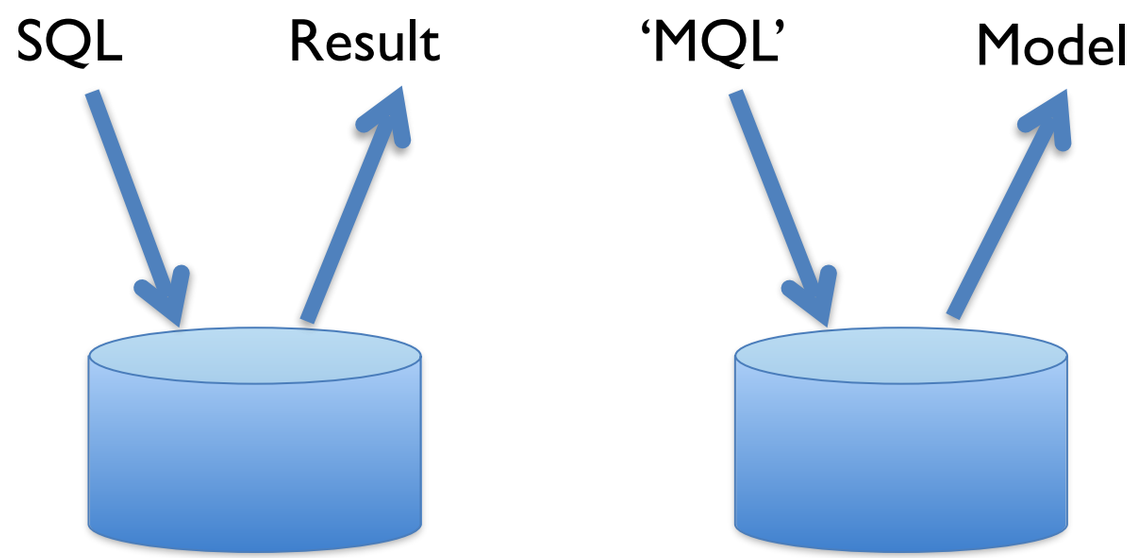
Vision

MLlib and MLI

MLOpt

DATABRICKS

Grand Vision

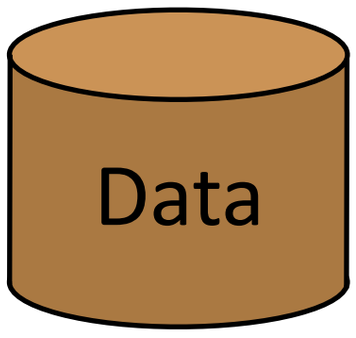


- User declaratively specifies a task
- Search through MLlib/MLI to find the best model/pipeline

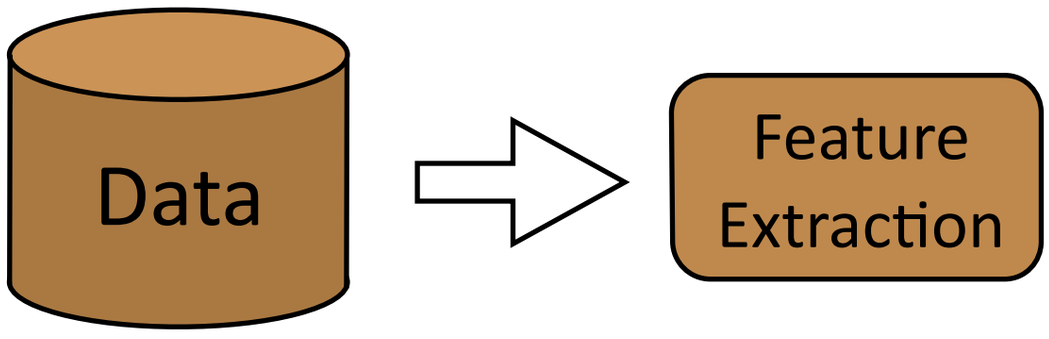
DATABRICKS

A Standard ML Pipeline

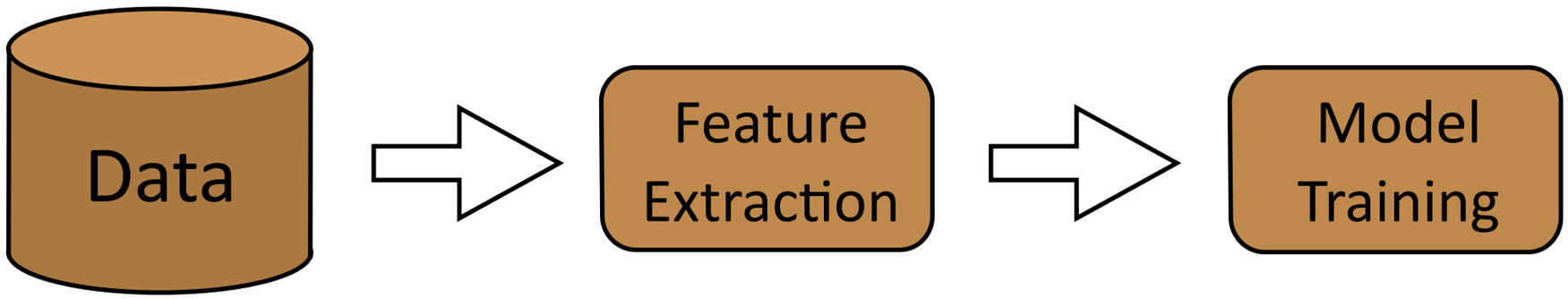
A Standard ML Pipeline



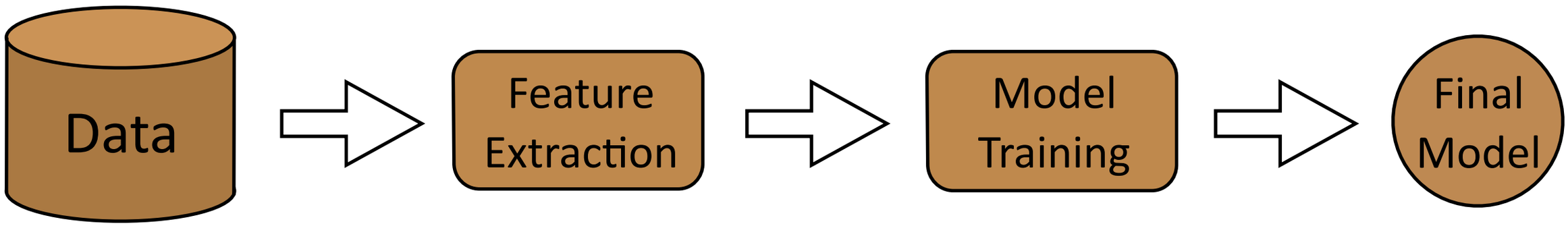
A Standard ML Pipeline



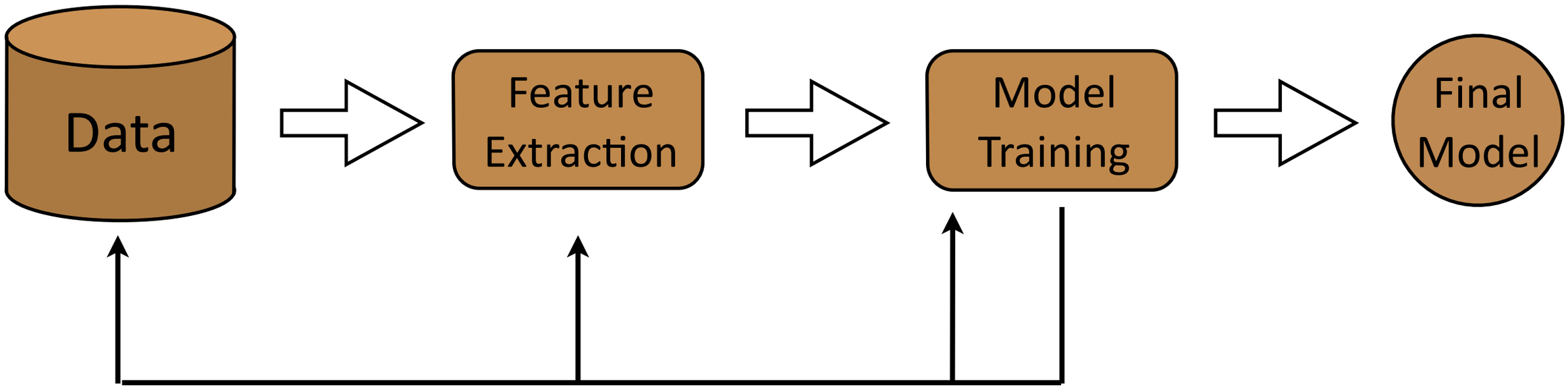
A Standard ML Pipeline



A Standard ML Pipeline

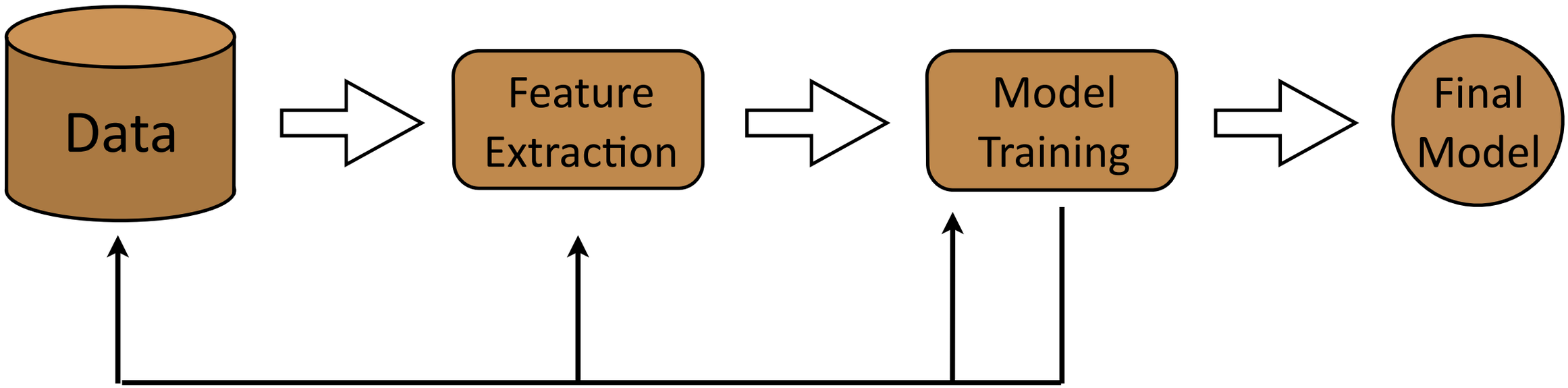


A Standard ML Pipeline



- In practice, model building is an iterative process of continuous refinement

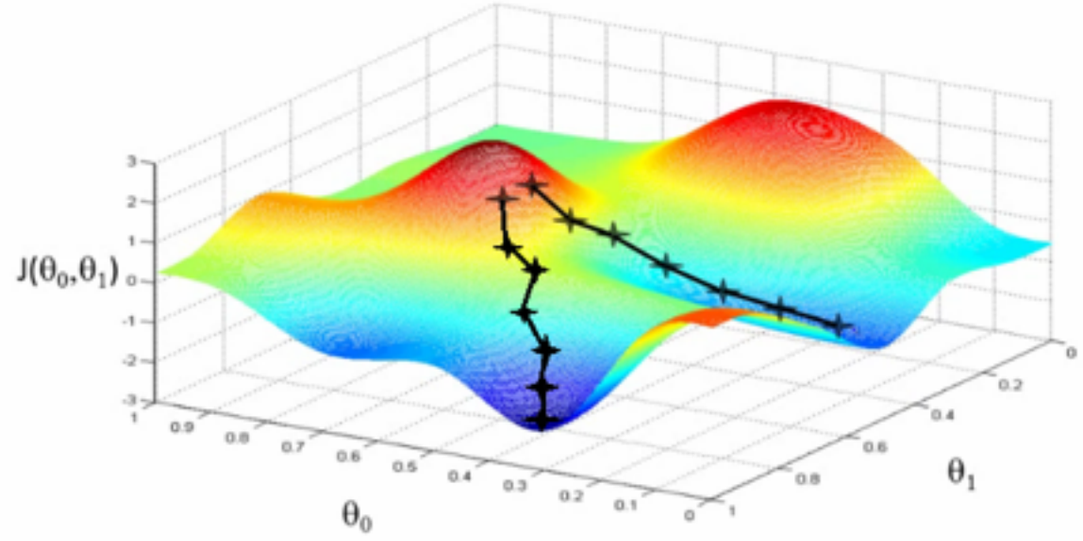
A Standard ML Pipeline



- In practice, model building is an iterative process of continuous refinement
- Our grand vision is to automate the construction of these pipelines

Training A Model

$$w := w - \alpha \nabla Q(w) = w - \alpha \sum_{i=1}^{n} \nabla Q_i(w),$$

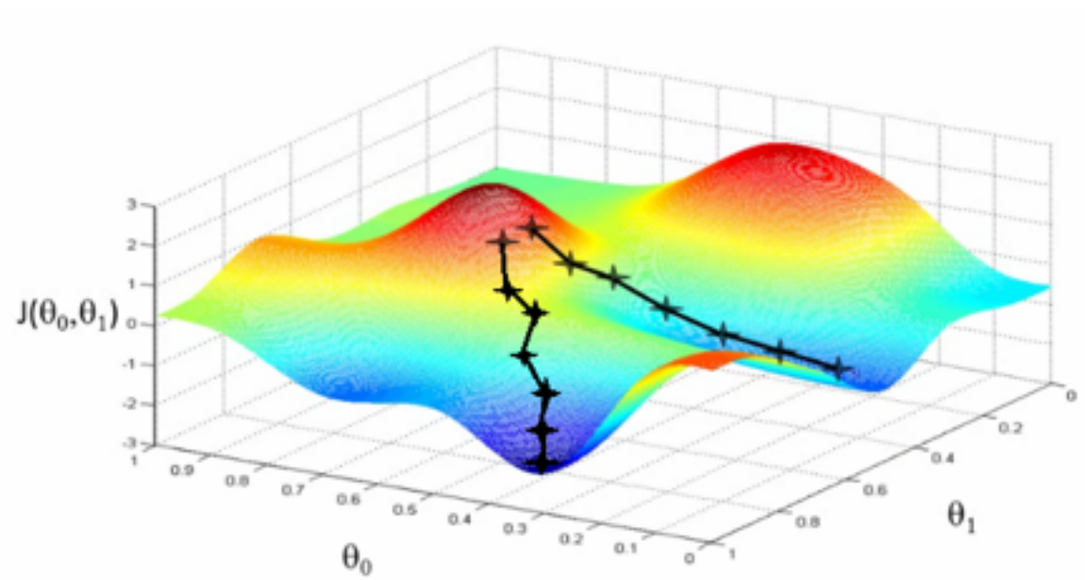


DATABRICKS

Training A Model

- For each point in dataset
  - compute gradient
  - update model
  - repeat until converged

$$w := w - \alpha \nabla Q(w) = w - \alpha \sum_{i=1}^{n} \nabla Q_i(w),$$

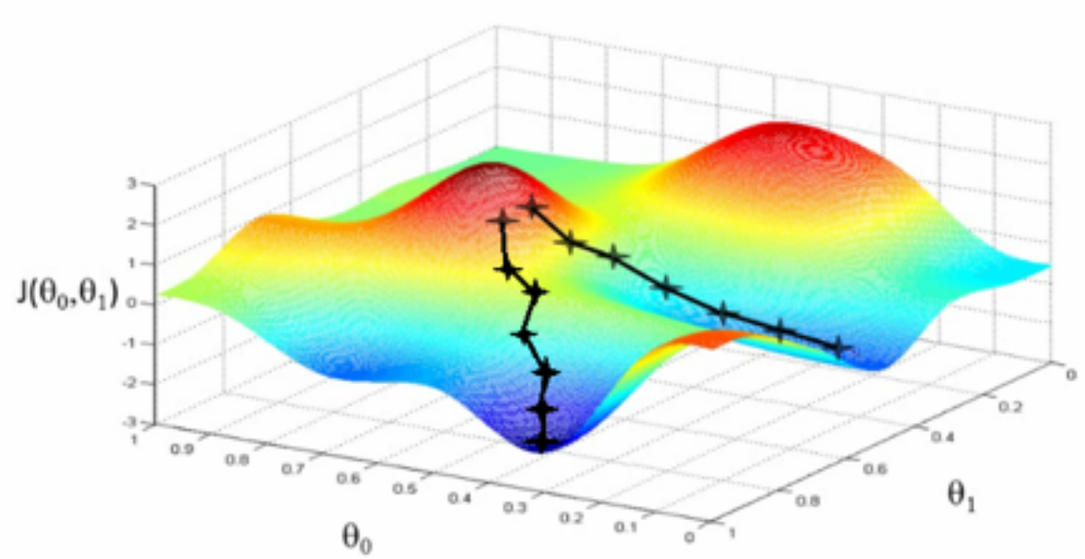


DATABRICKS

Training A Model

- For each point in dataset
  - compute gradient
  - update model
  - repeat until converged
- Requires ***multiple passes***

$$w := w - \alpha \nabla Q(w) = w - \alpha \sum_{i=1}^{n} \nabla Q_i(w),$$

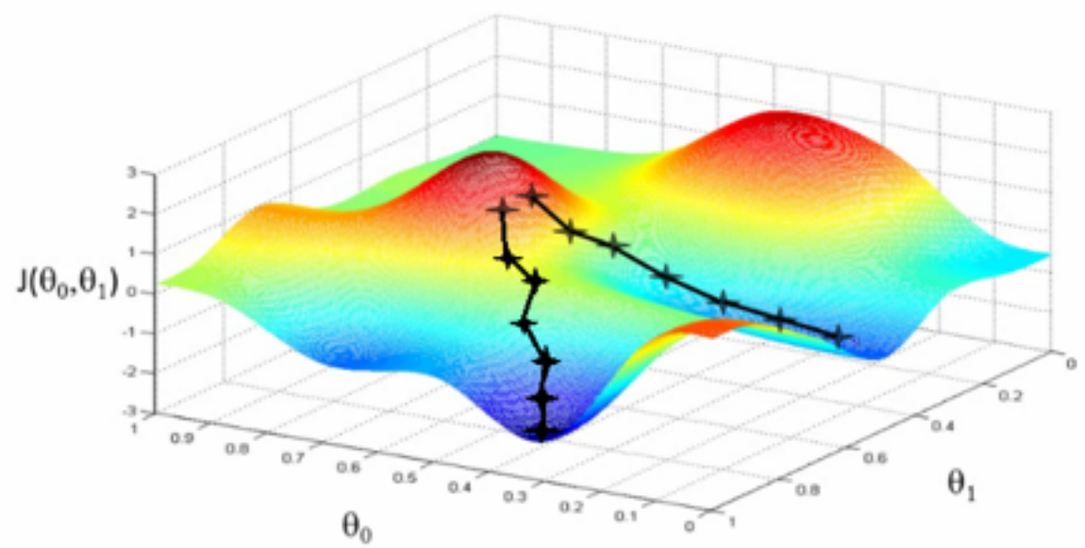


DATABRICKS

Training A Model

- For each point in dataset
  - compute gradient
  - update model
  - repeat until converged
- Requires ***multiple passes***
- Common access pattern
  - Naive Bayes, Trees, etc.

$$w := w - \alpha \nabla Q(w) = w - \alpha \sum_{i=1}^{n} \nabla Q_i(w),$$

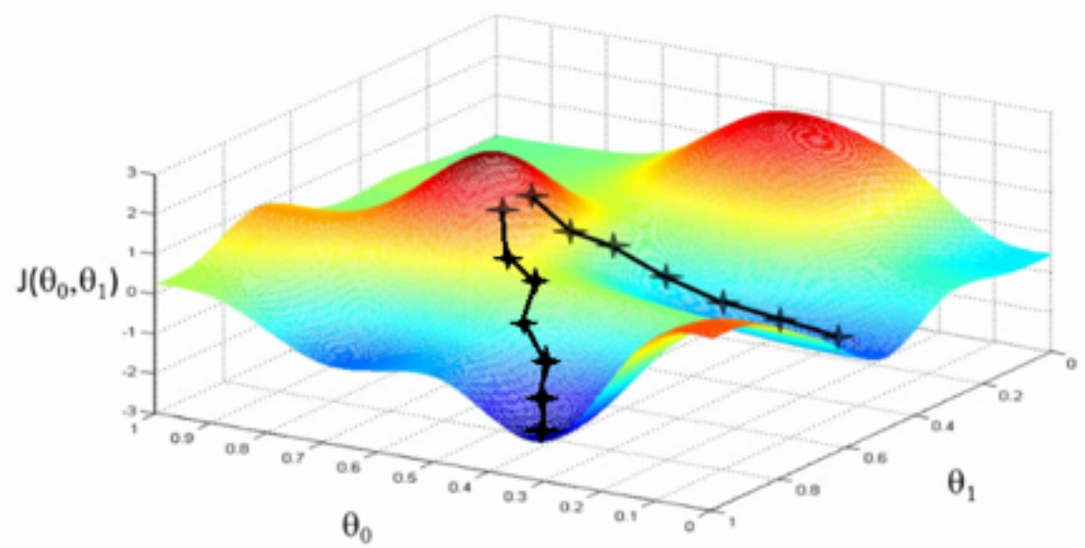


DATABRICKS

Training A Model

- For each point in dataset
  - compute gradient
  - update model
  - repeat until converged

$$w := w - \alpha \nabla Q(w) = w - \alpha \sum_{i=1}^{n} \nabla Q_i(w),$$

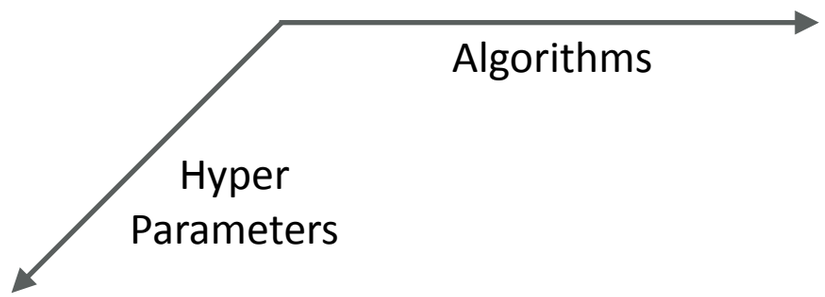


- Requires ***multiple passes***
- Common access pattern
  - Naive Bayes, Trees, etc.
- Minutes to train an SVM on 200GB of data on a 16-node cluster

DATABRICKS

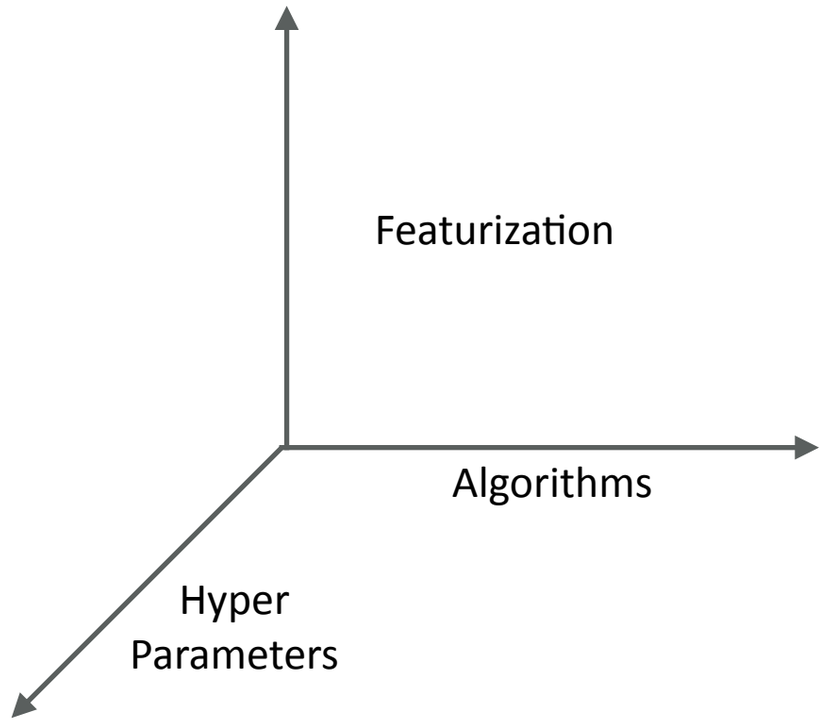
The Tricky Part

- Algorithms
  - Logistic Regression, SVM, Tree-based, etc.
- Algorithm hyper-parameters
  - Learning Rate, Regularization, etc.



The Tricky Part

- Algorithms
  - Logistic Regression, SVM, Tree-based, etc.
- Algorithm hyper-parameters
  - Learning Rate, Regularization, etc.

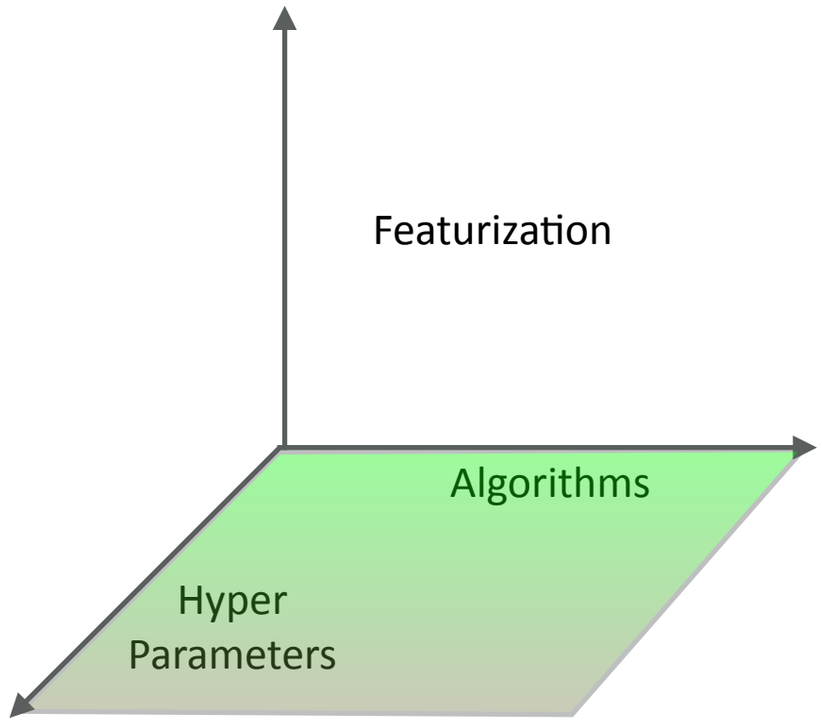


- Featurization
  - Text: n-grams, TF-IDF
  - Images: Gabor filters, random convolutions
  - Random projection? Scaling?

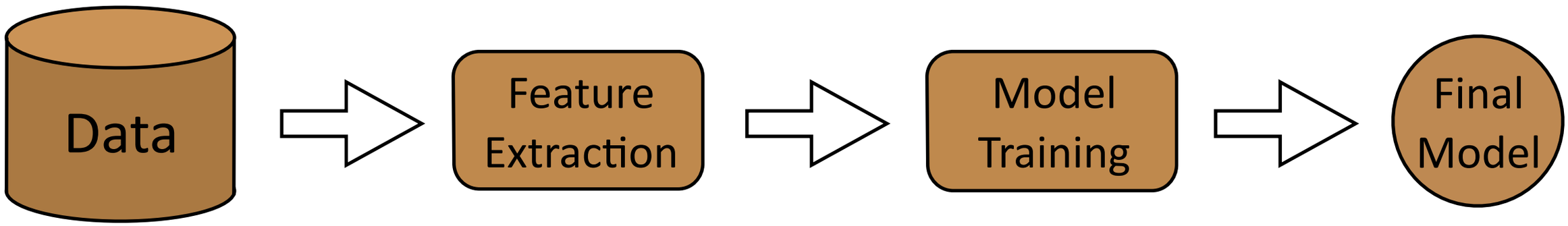
The Tricky Part

- Algorithms
  - Logistic Regression, SVM, Tree-based, etc.
- Algorithm hyper-parameters
  - Learning Rate, Regularization, etc.

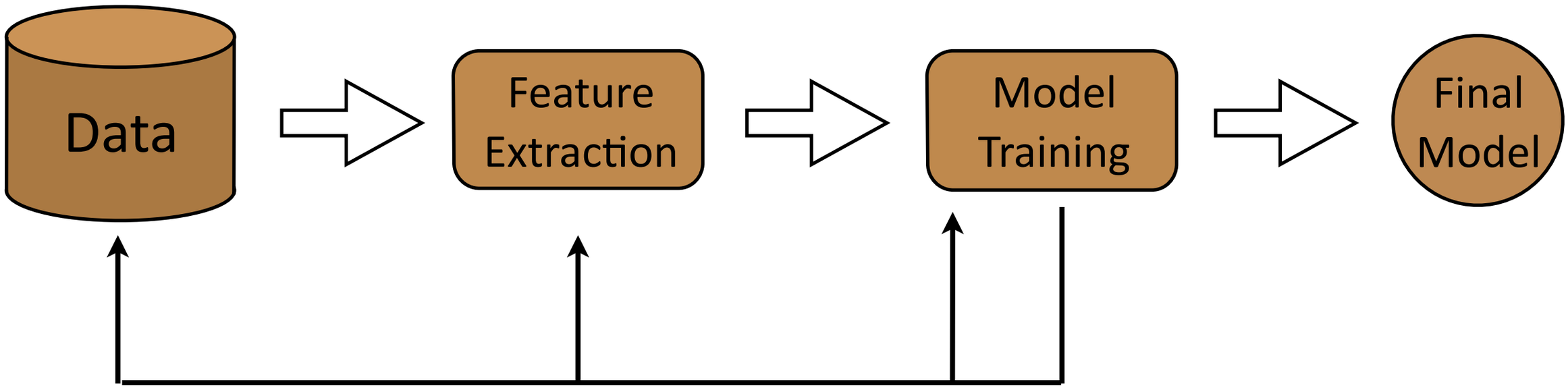
- Featurization
  - Text: n-grams, TF-IDF
  - Images: Gabor filters, random convolutions
  - Random projection? Scaling?



A Standard ML Pipeline

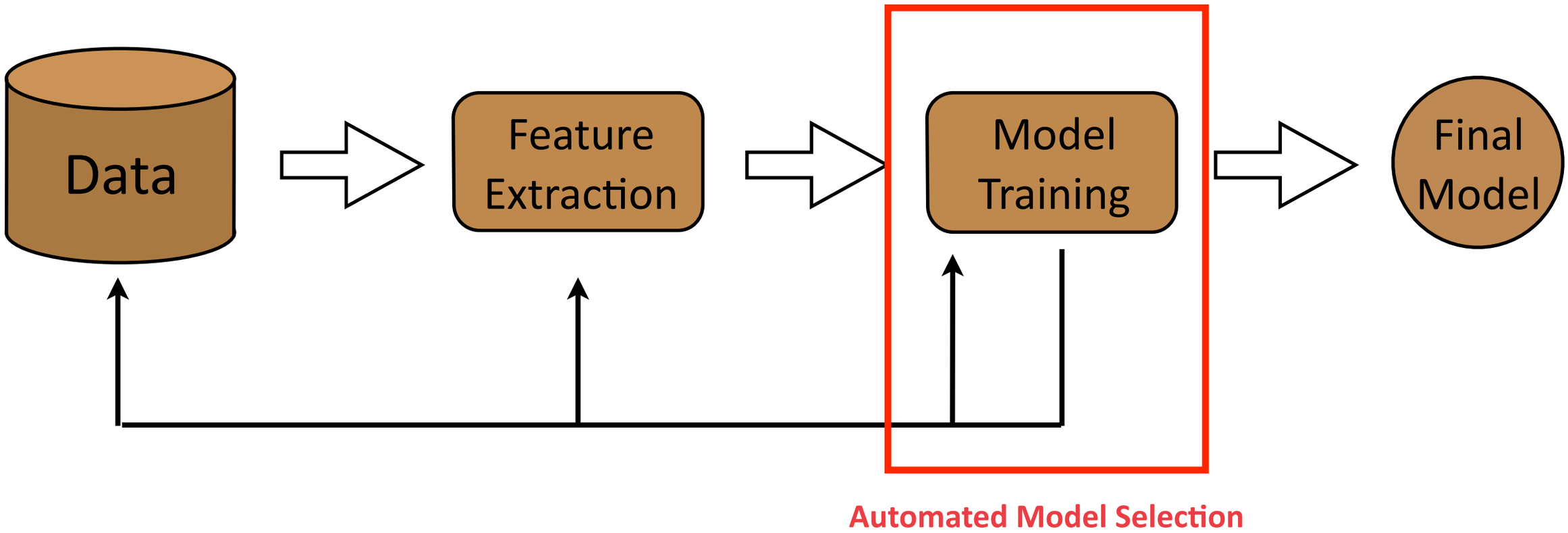


A Standard ML Pipeline



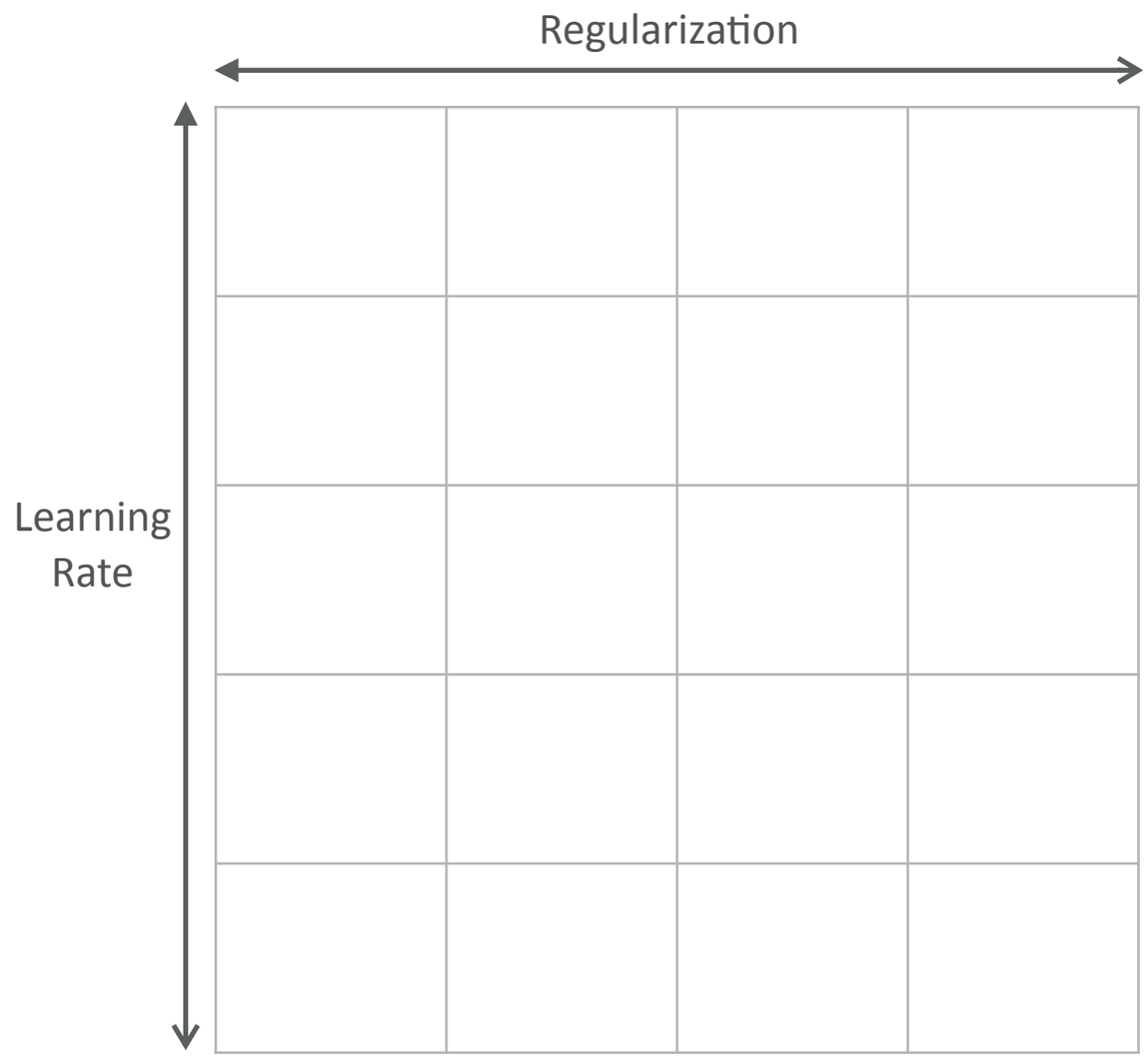
- In practice, model building is an iterative process of continuous refinement
- Our grand vision is to automate the construction of these pipelines

A Standard ML Pipeline



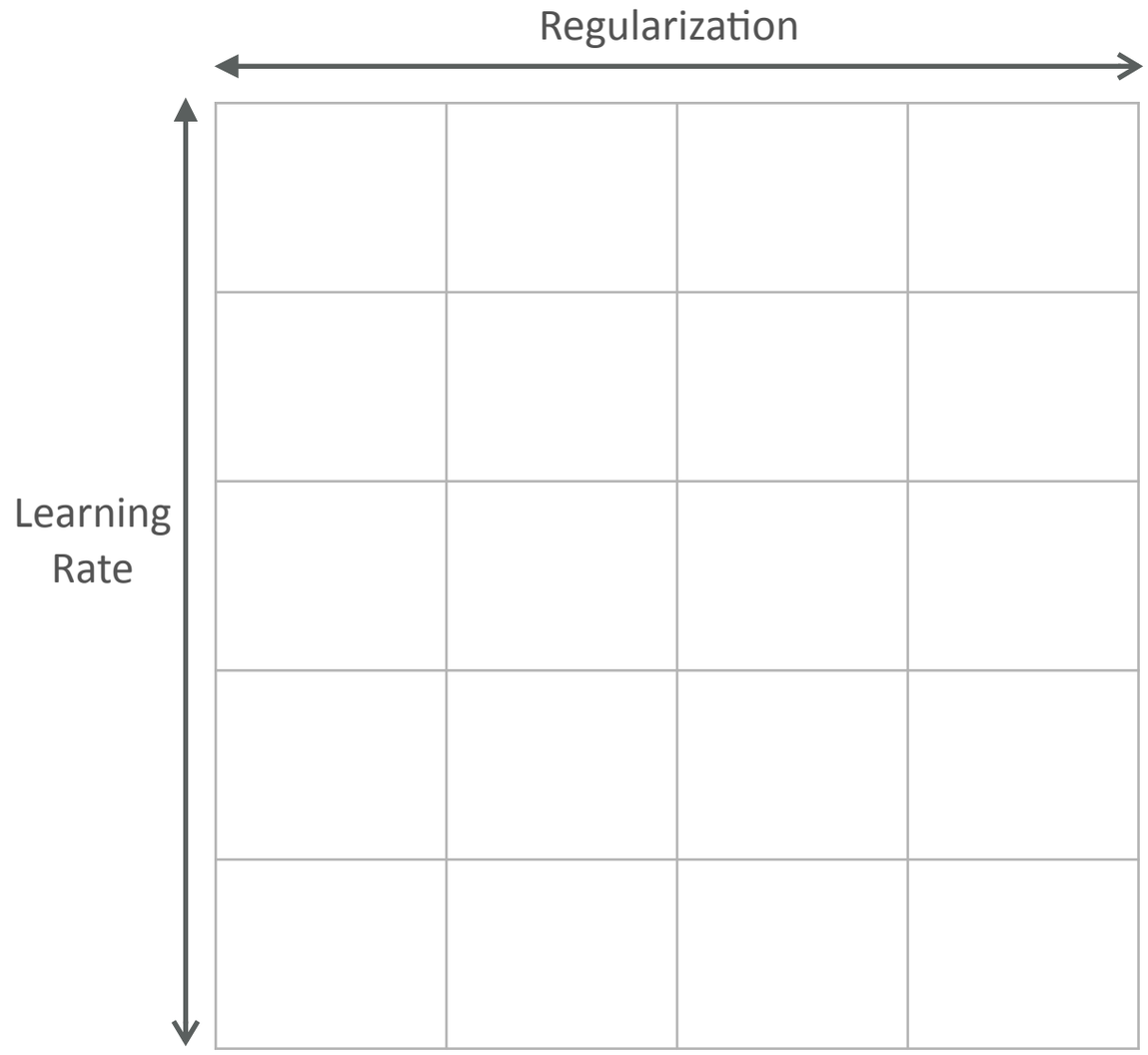
- In practice, model building is an iterative process of continuous refinement
- Our grand vision is to automate the construction of these pipelines
- Start with one aspect of the pipeline - model selection

One Approach



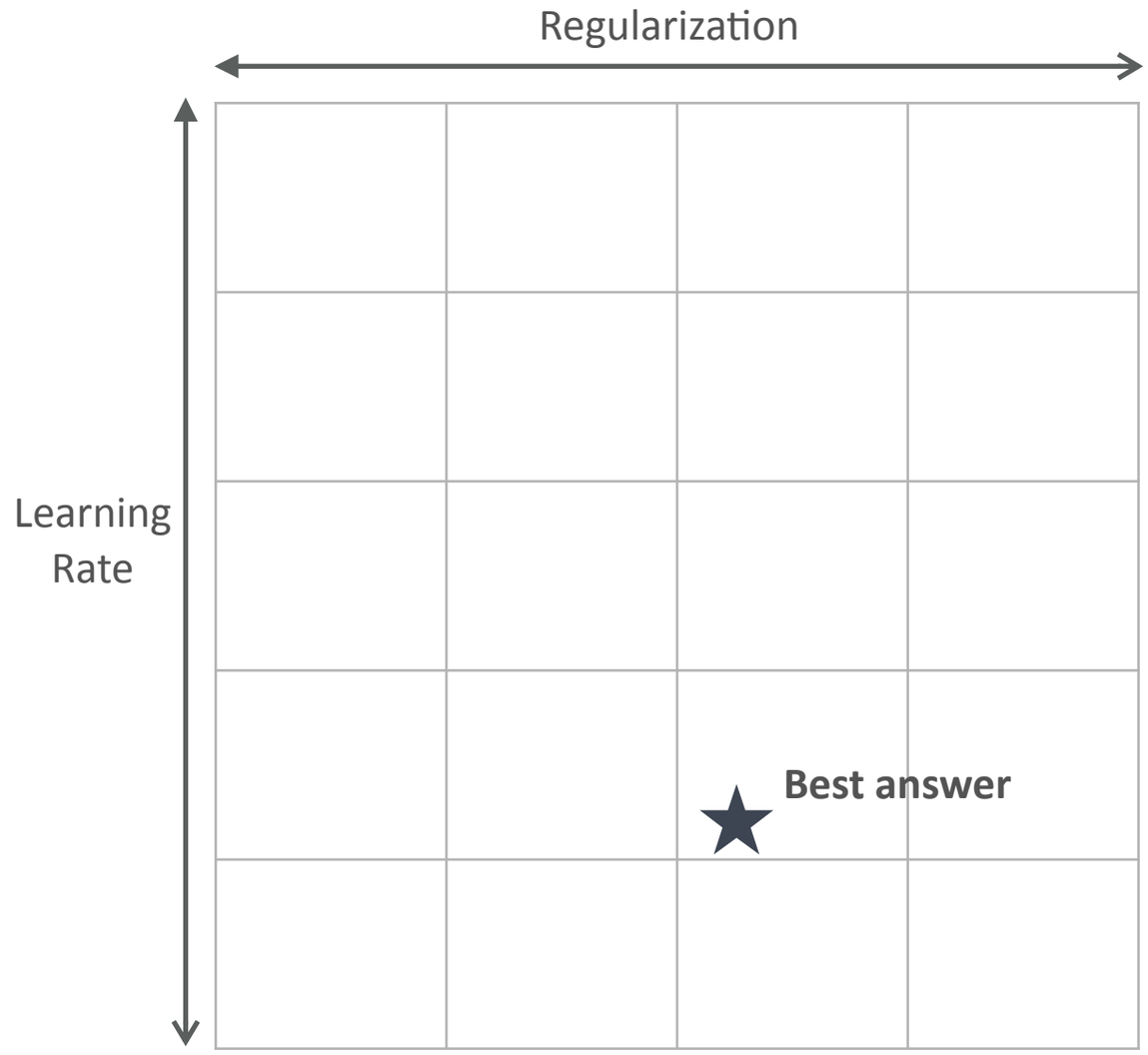
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



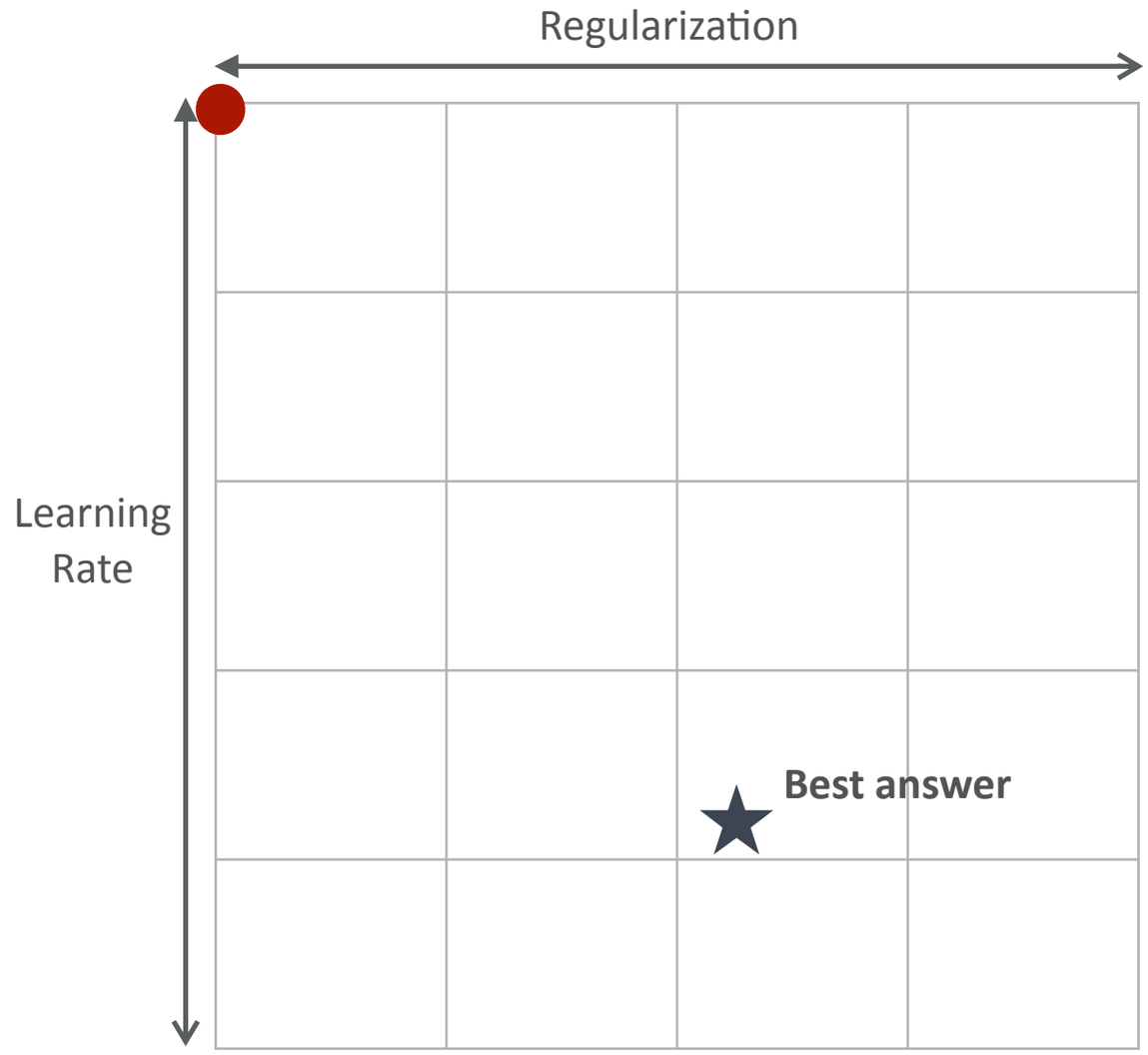
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



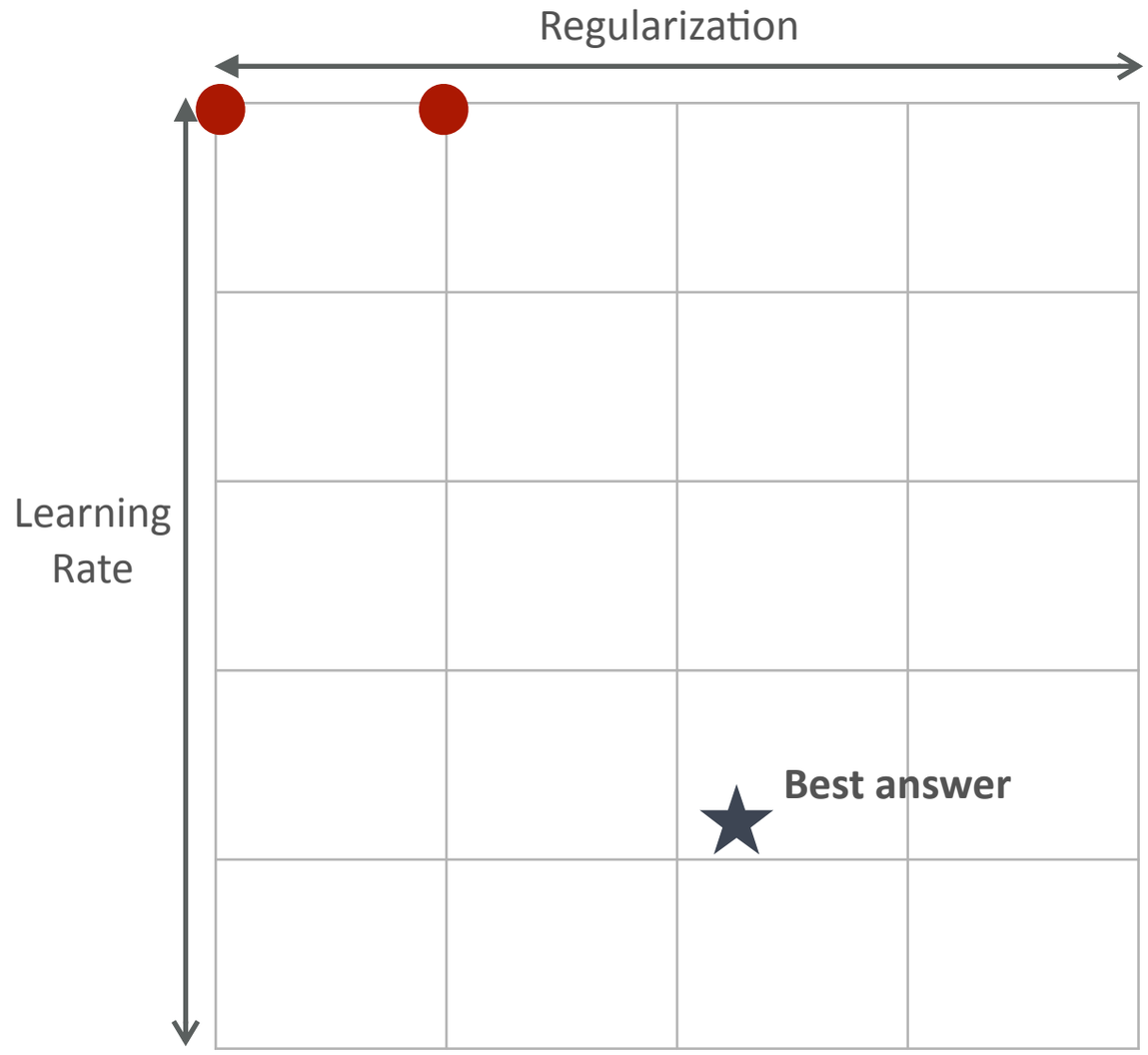
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



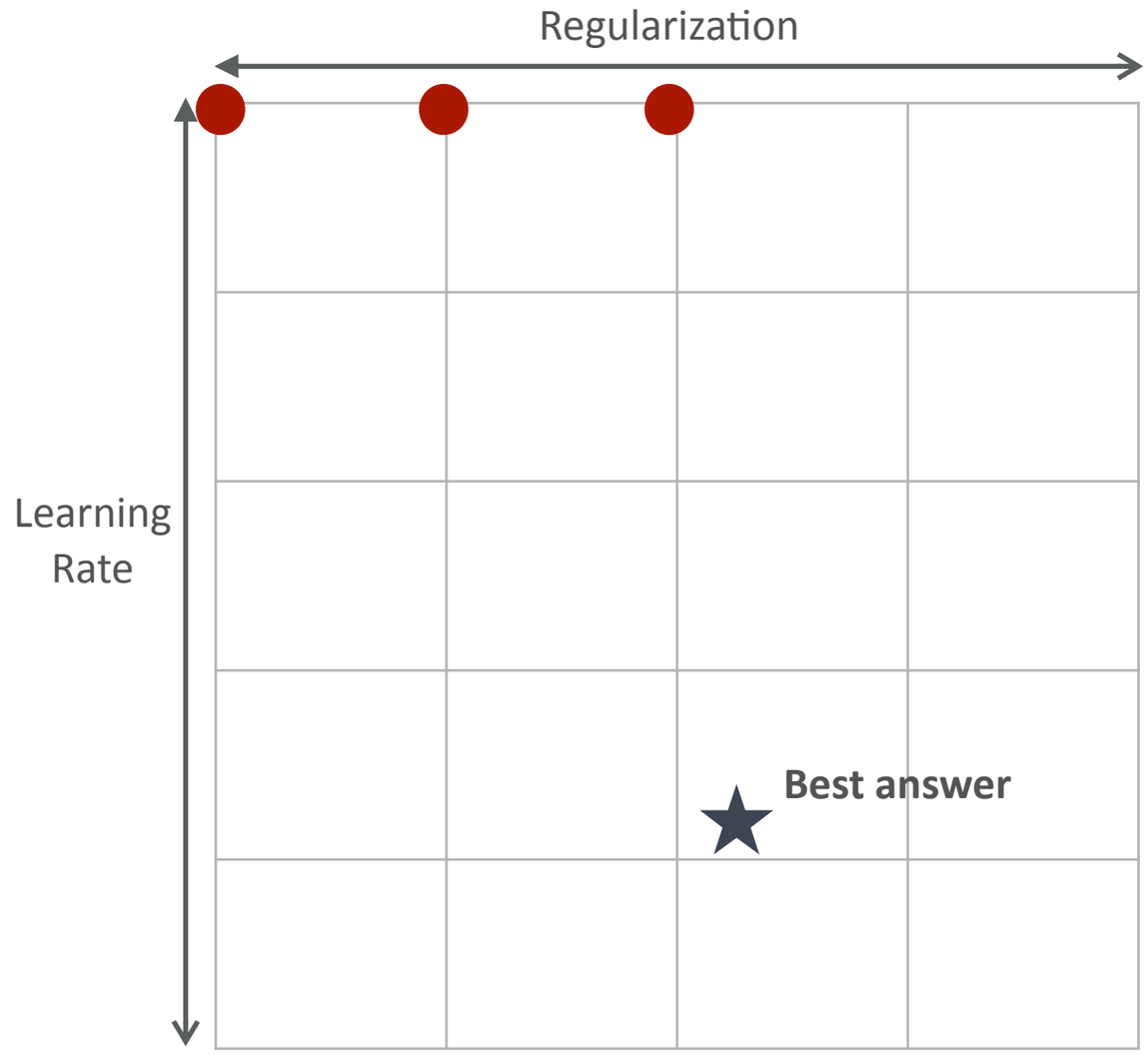
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



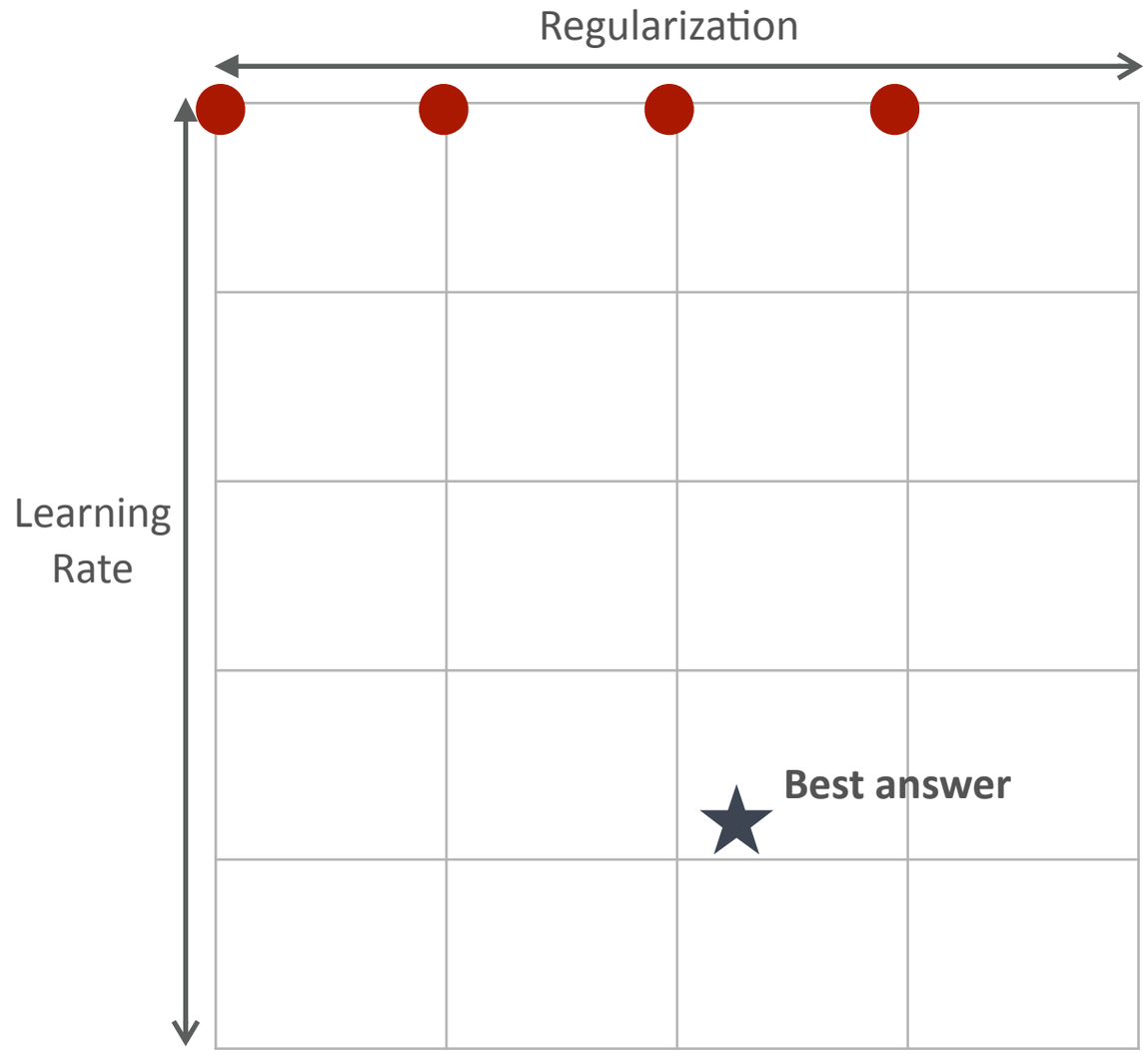
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



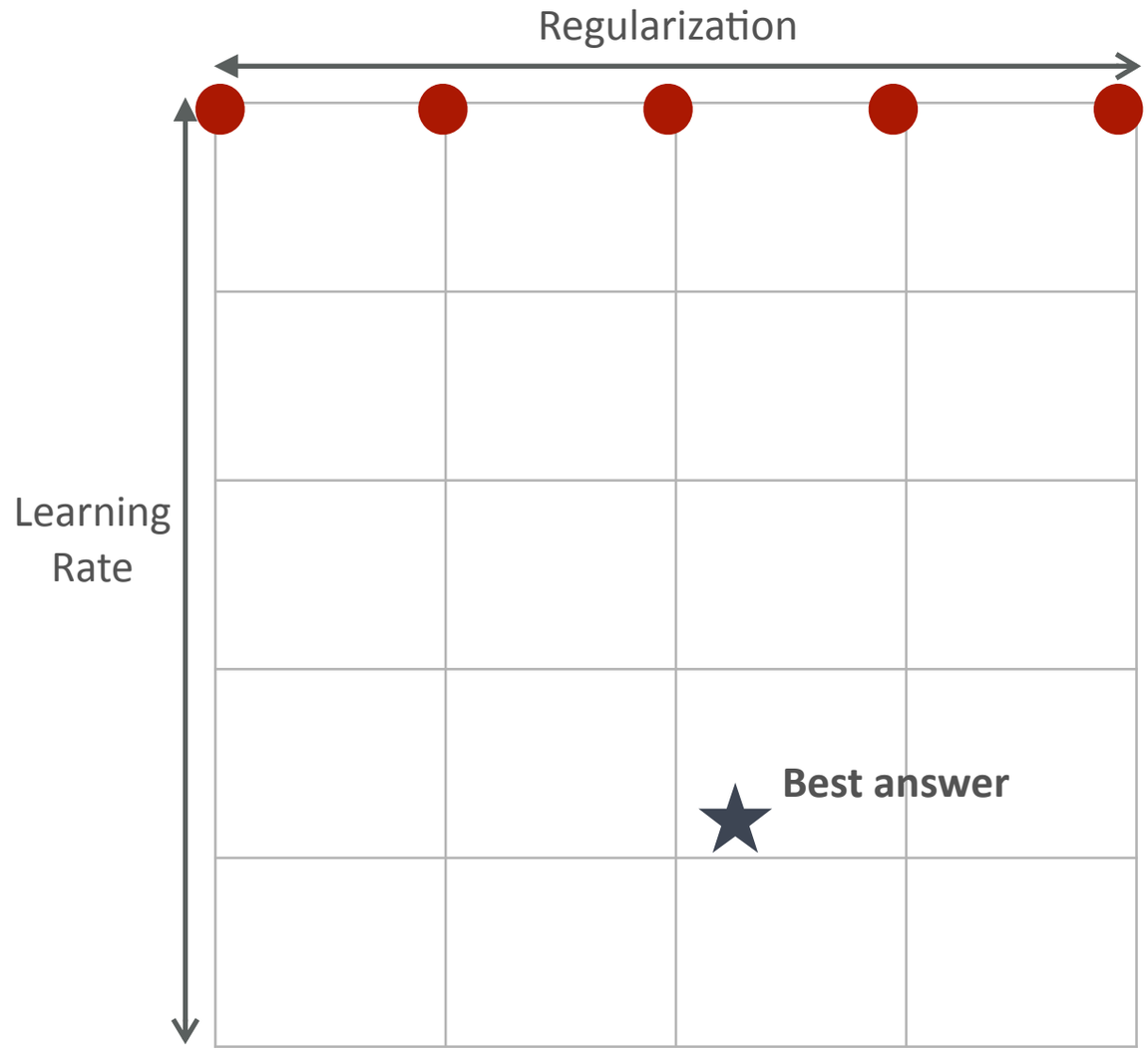
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



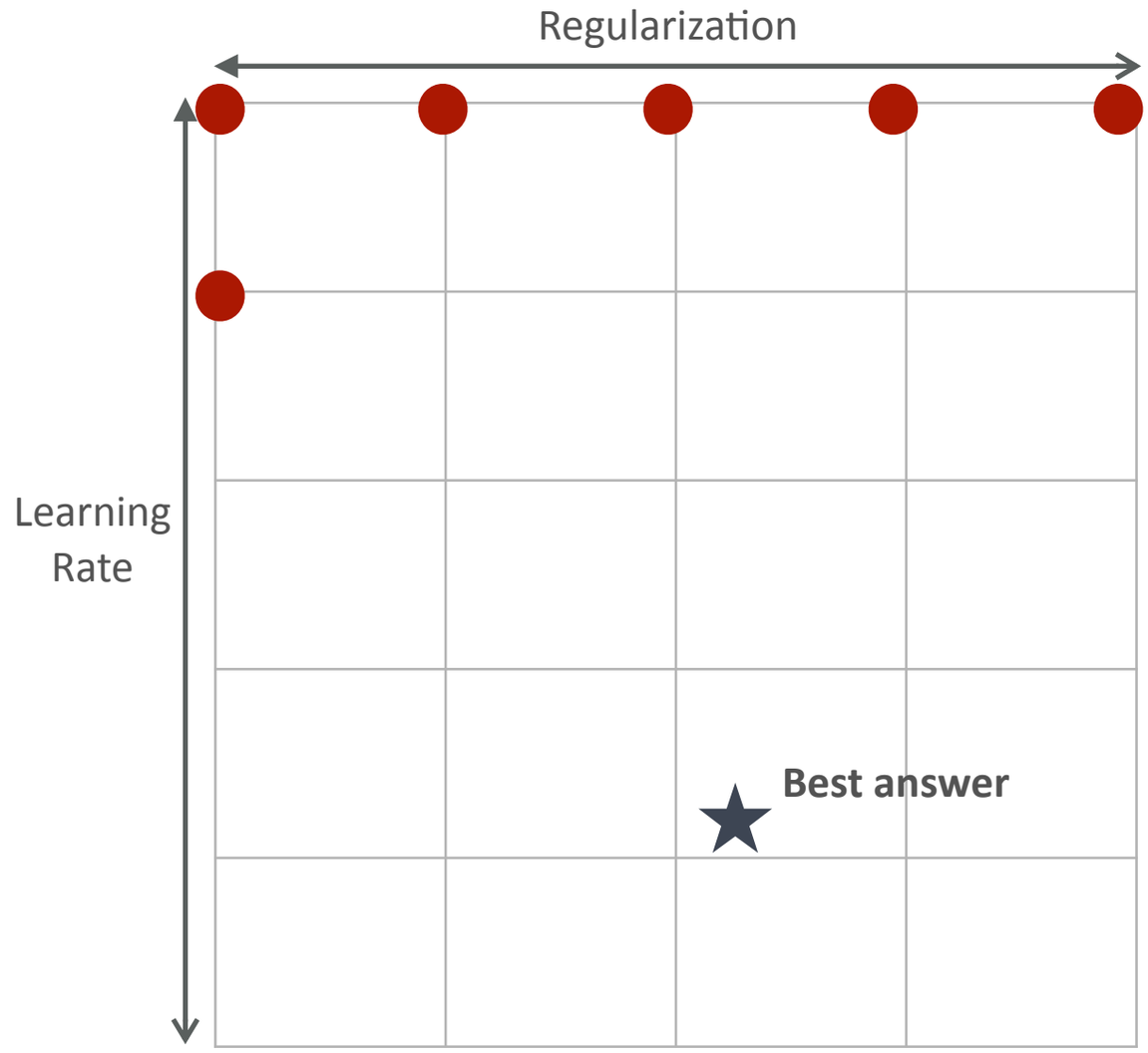
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



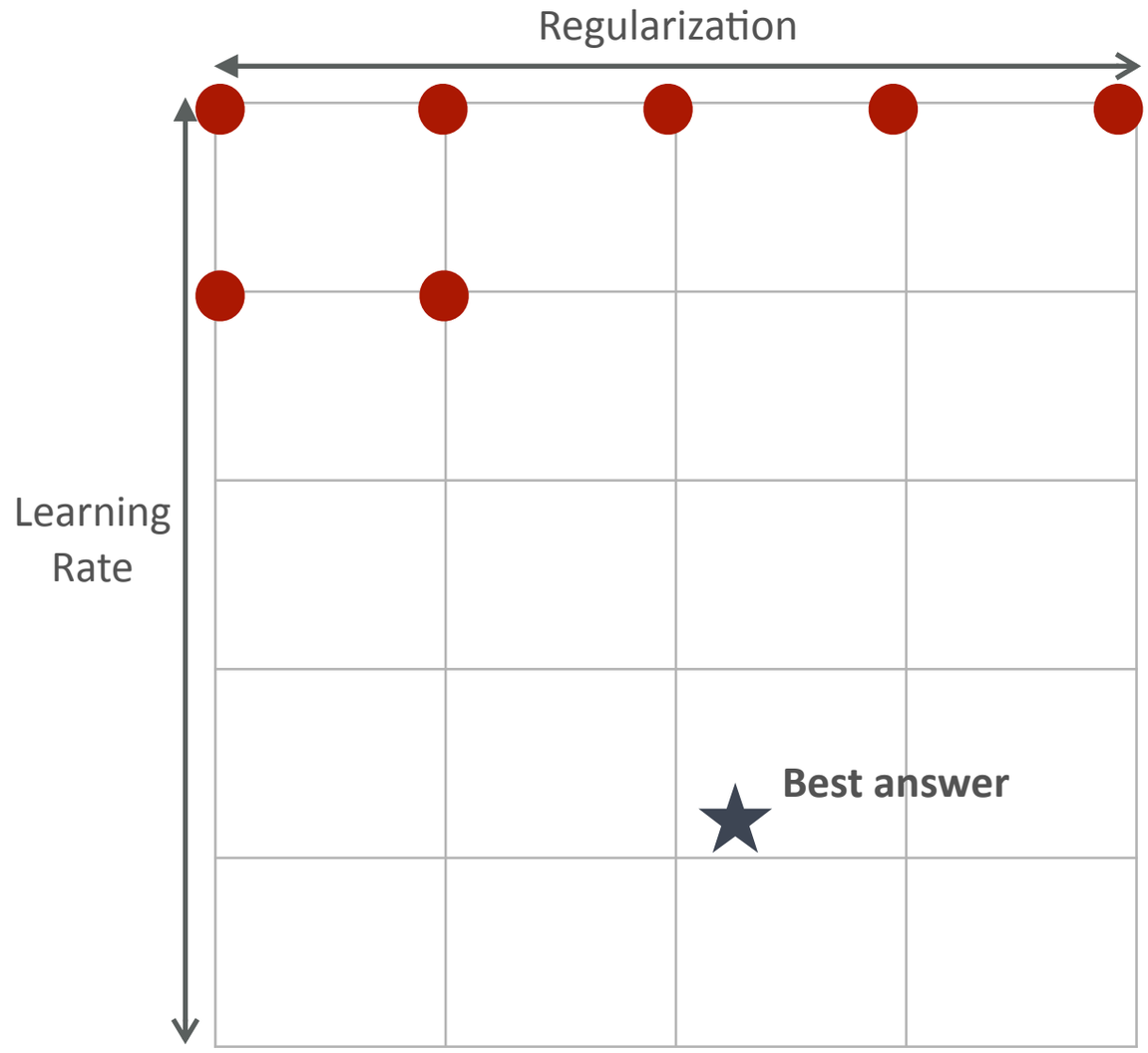
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



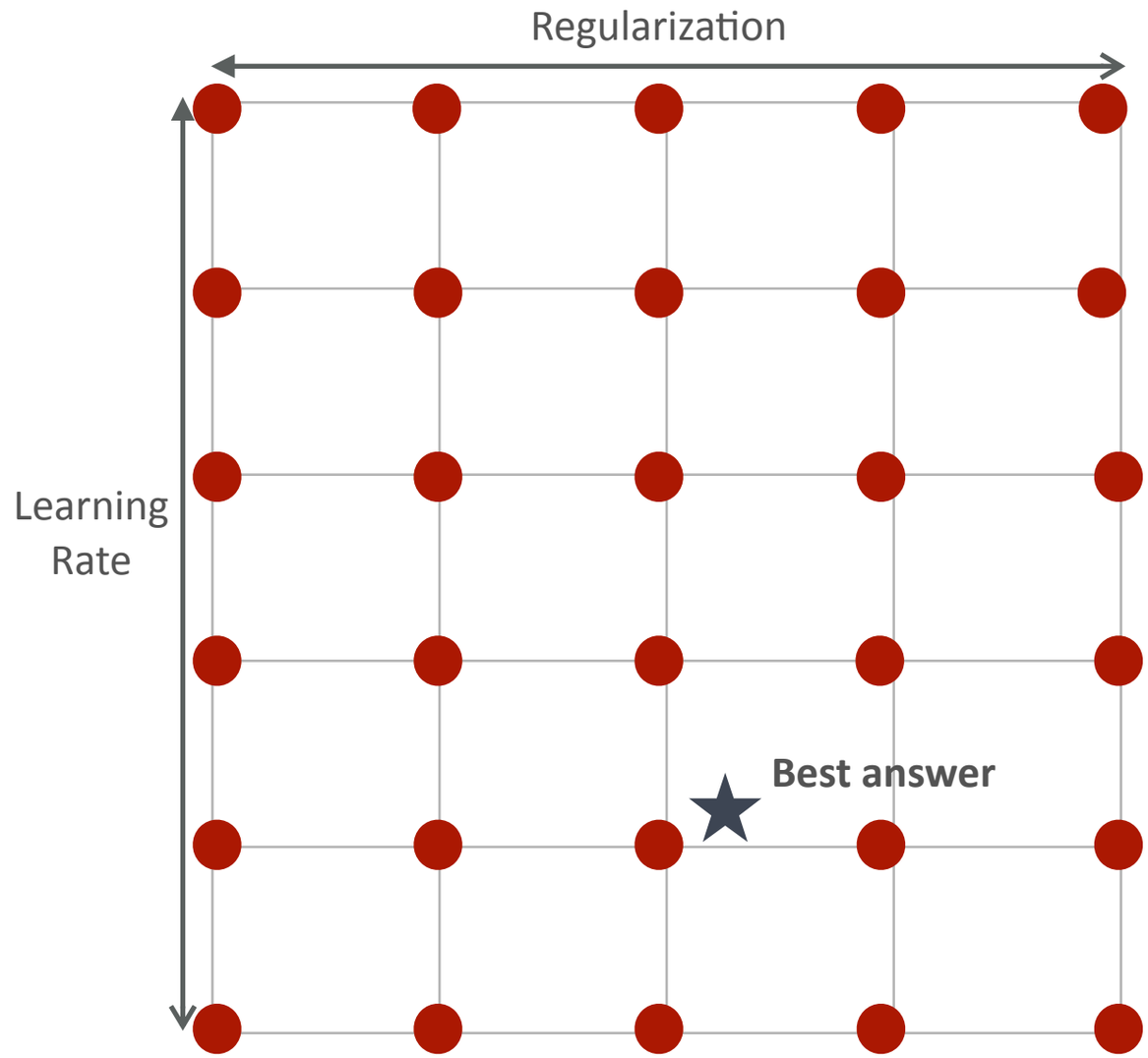
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



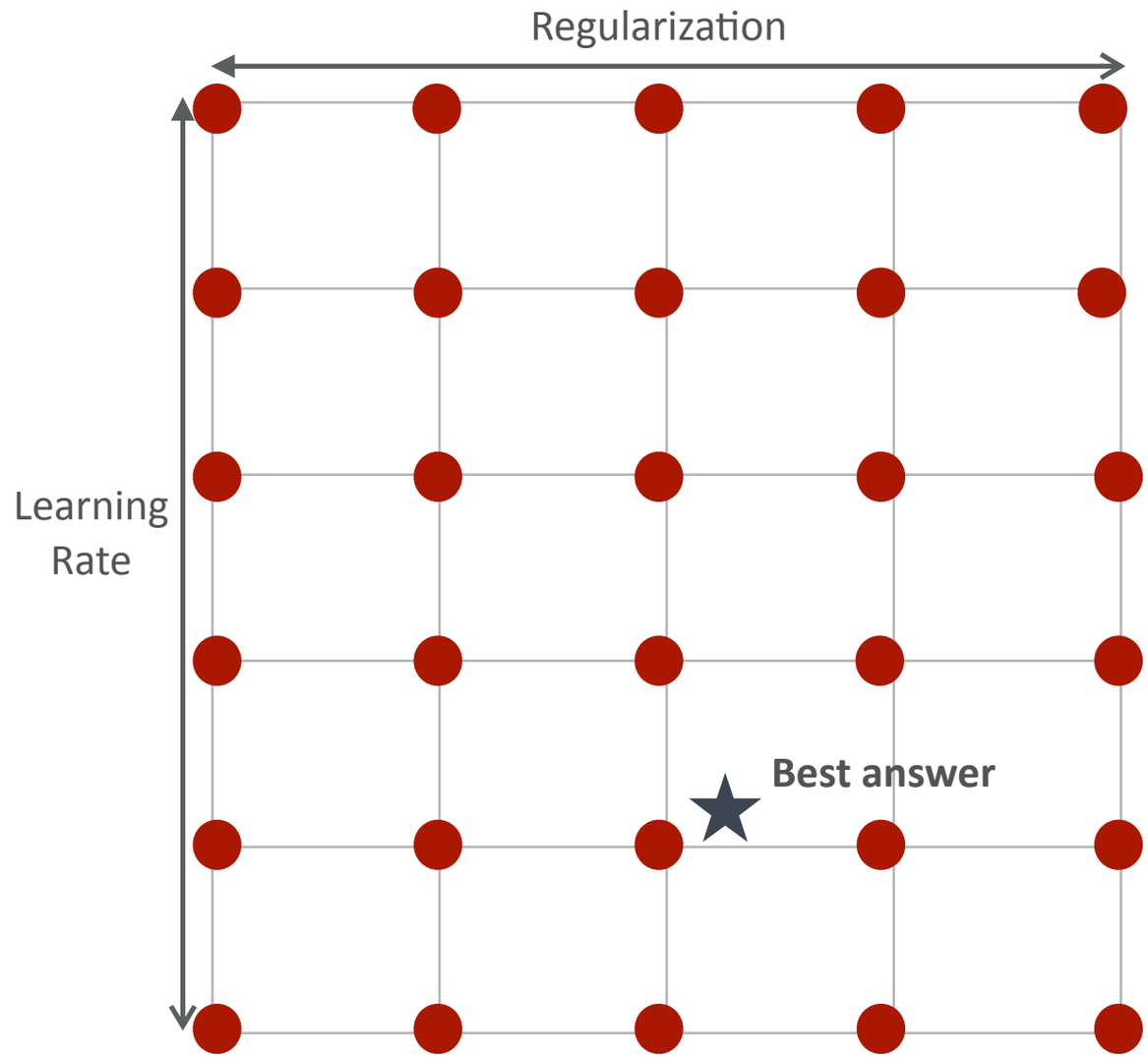
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.



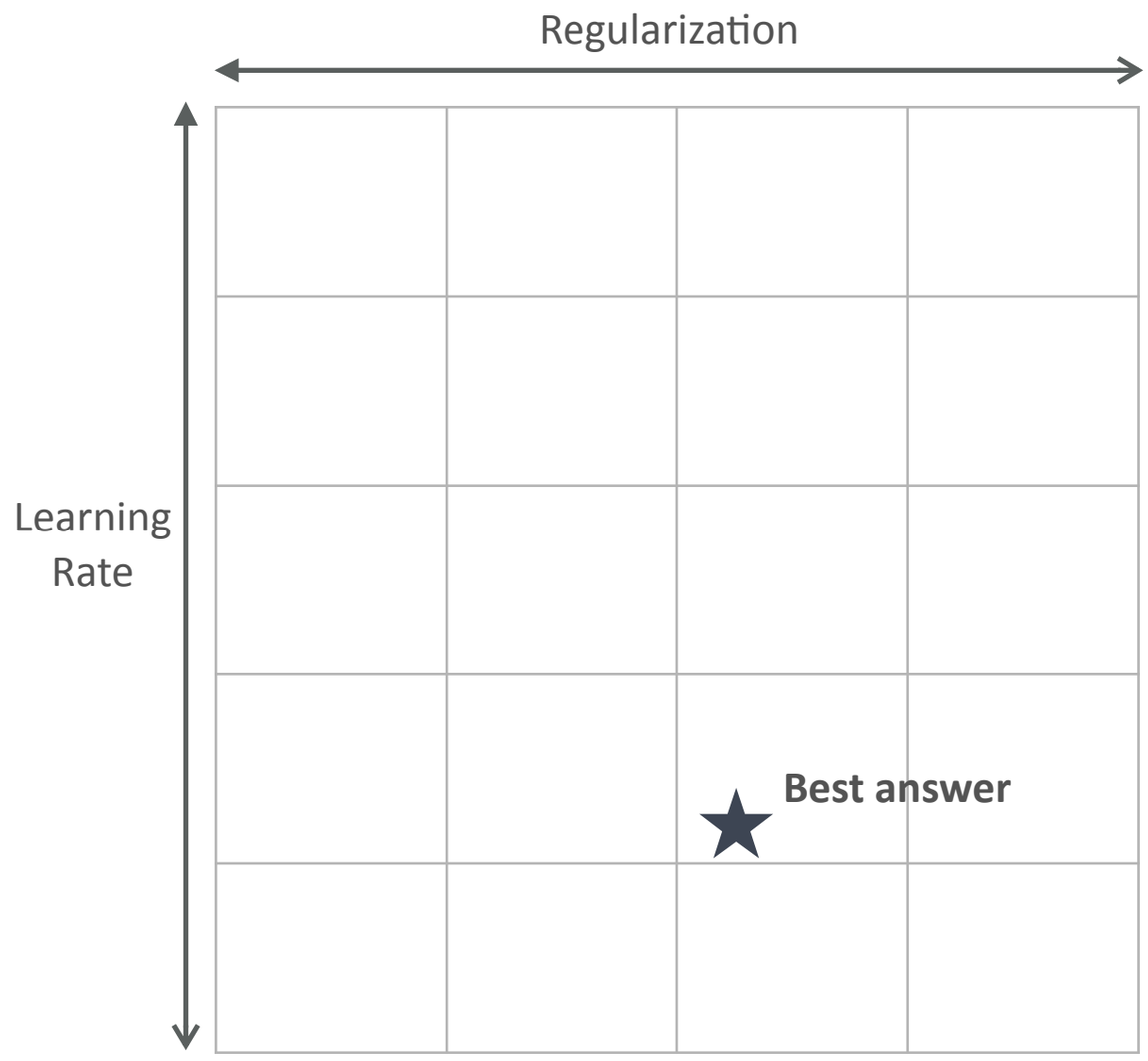
One Approach

- Try it all!
  - Search over all hyperparameters, algorithms, features, etc.
- Drawbacks
  - Expensive to compute models
  - Hyperparameter space is large
- Some version of this still often done in practice!



A Better Approach

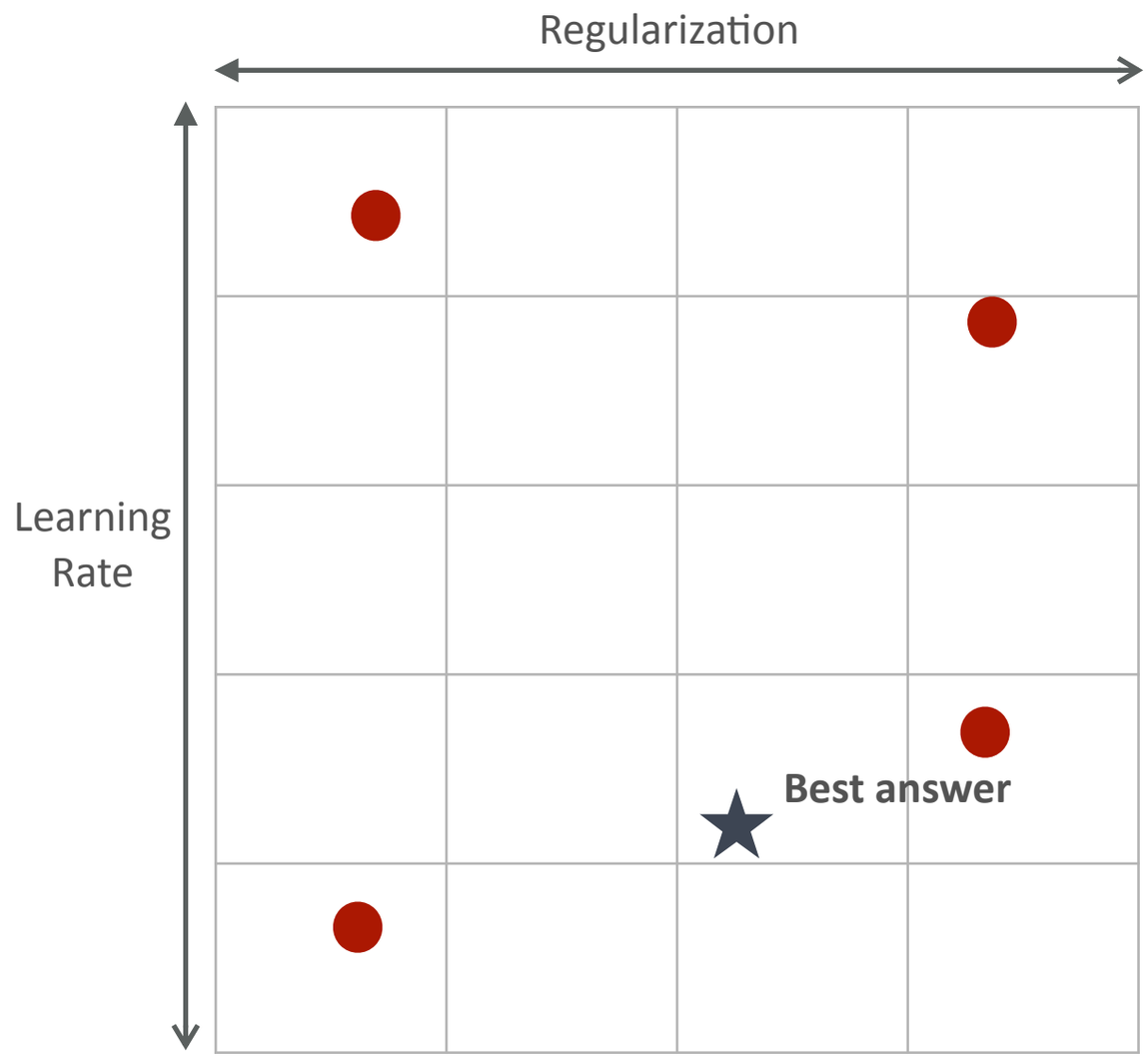
- Better resource utilization
  - through batching
- Algorithmic Speedups
  - via early stopping
- Improved Search
  - e.g., via randomization



DATABRICKS

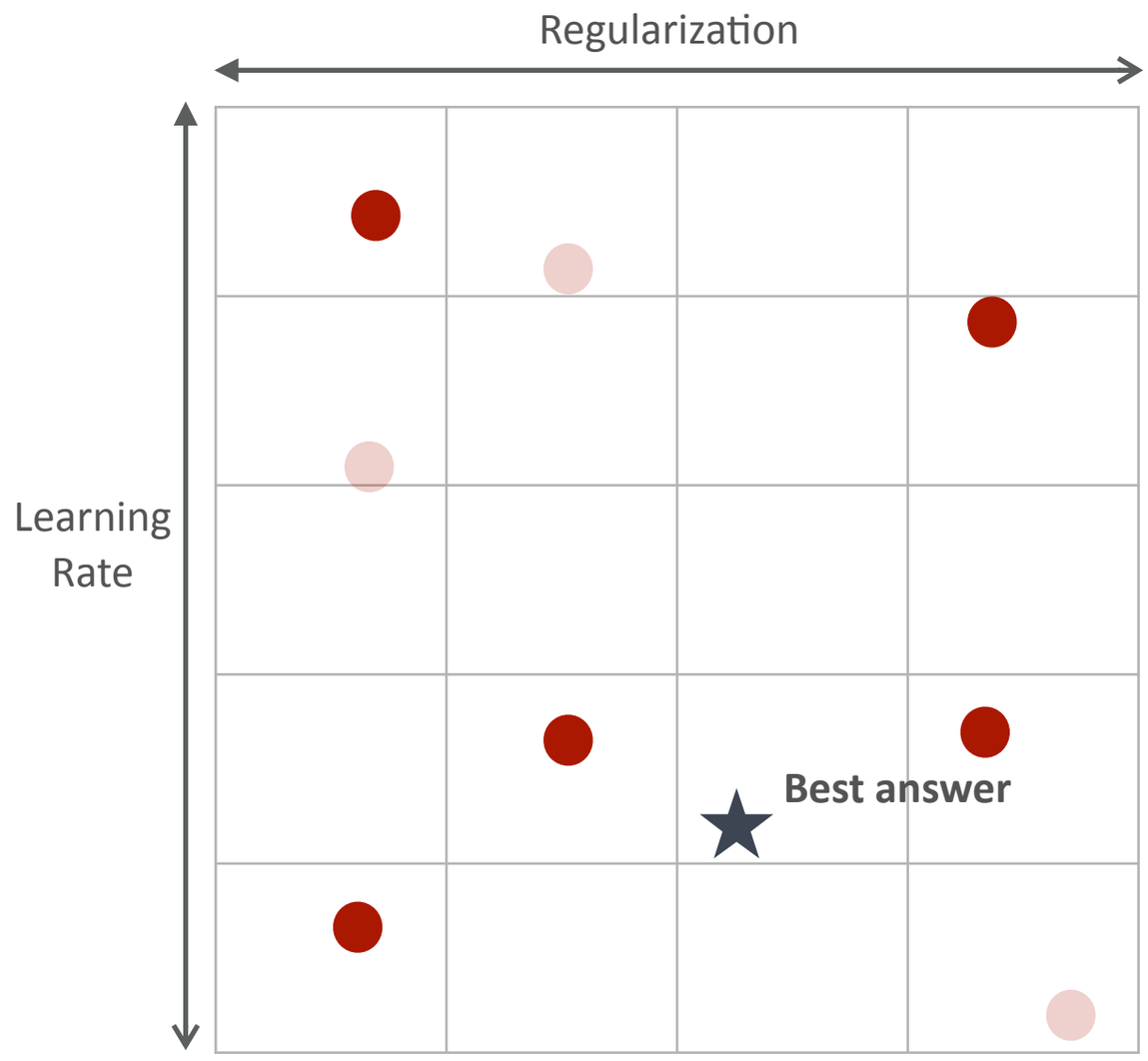
A Better Approach

- Better resource utilization
  - through batching
- Algorithmic Speedups
  - via early stopping
- Improved Search
  - e.g., via randomization



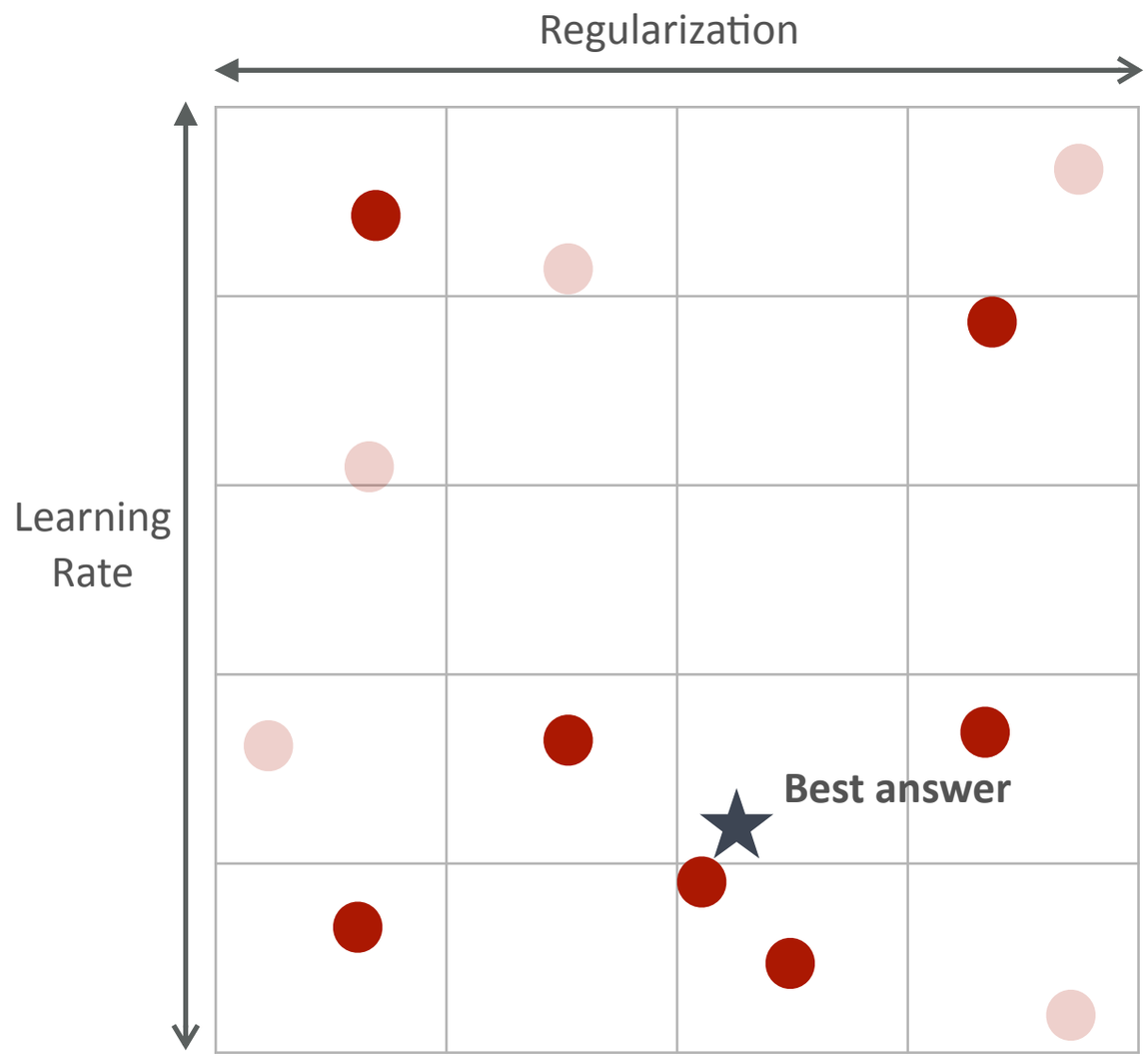
A Better Approach

- Better resource utilization
  - through batching
- Algorithmic Speedups
  - via early stopping
- Improved Search
  - e.g., via randomization



A Better Approach

- Better resource utilization
  - through batching
- Algorithmic Speedups
  - via early stopping
- Improved Search
  - e.g., via randomization



A Tale Of 3 Optimizations

Better Resource Utilization

Algorithmic Speedups

Improved Search

DATABRICKS

Better Resource Utilization

Better Resource Utilization

- ✦ Modern memory slower than processors
- ✦ Can read: 0.6b doubles/sec/core (4.8 GB/s)
- ✦ Can compute: 15b flops/sec/core

DATABRICKS

Better Resource Utilization

- ✦ Modern memory slower than processors
- ✦ Can read: 0.6b doubles/sec/core (4.8 GB/s)
- ✦ Can compute: 15b flops/sec/core
- ✦ **We can do 25 flops/double read**

DATABRICKS

What Does This Mean For Modeling?

A	B	C
1	a	Dog
1	b	Cat
2	c	Cat
2	d	Cat
3	e	Dog
3	f	Horse
4	g	Doge

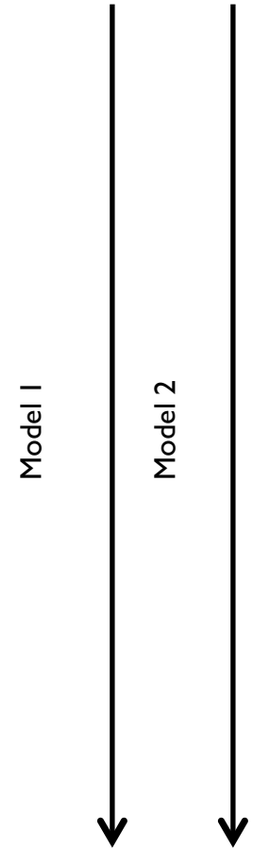
What Does This Mean For Modeling?

- Typical model update requires 2-4 flops/double
  - recall: 25 flops / double read

A	B	C
1	a	Dog
1	b	Cat
2	c	Cat
2	d	Cat
3	e	Dog
3	f	Horse
4	g	Doge

What Does This Mean For Modeling?

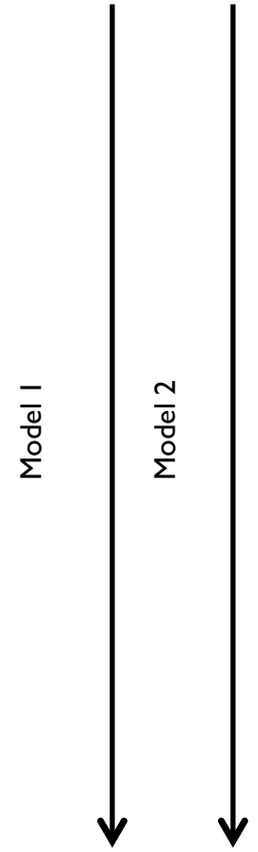
- Typical model update requires 2-4 flops/double
  - recall: 25 flops / double read
- Can do 7-10 model updates per double we read
  - Assuming that models fit in cache



A	B	C
1	a	Dog
1	b	Cat
2	c	Cat
2	d	Cat
3	e	Dog
3	f	Horse
4	g	Doge

What Does This Mean For Modeling?

- Typical model update requires 2-4 flops/double
  - recall: 25 flops / double read
- Can do 7-10 model updates per double we read
  - Assuming that models fit in cache
- **Train multiple models simultaneously**



A	B	C
1	a	Dog
1	b	Cat
2	c	Cat
2	d	Cat
3	e	Dog
3	f	Horse
4	g	Doge

What Do We See In Spark?

- 2x and 5x increase in models trained/sec with batching
- Overhead from virtualization, network, etc.

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	1.91	1.50	1.51	1.38
5	4.05	2.53	1.93	1.36
10	5.31	3.40	2.37	1.14

What Do We See In Spark?

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	1.91	1.50	1.51	1.38
5	4.05	2.53	1.93	1.36
10	5.31	3.40	2.37	1.14

What Do We See In Spark?

- These numbers are with vector-matrix multiplies

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	1.91	1.50	1.51	1.38
5	4.05	2.53	1.93	1.36
10	5.31	3.40	2.37	1.14

What Do We See In Spark?

- These numbers are with vector-matrix multiplies

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	1.91	1.50	1.51	1.38
5	4.05	2.53	1.93	1.36
10	5.31	3.40	2.37	1.14

- Can do better when rewriting in terms of matrix-matrix multiplies

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	0.97	1.51	0.92	1.28
5	2.67	2.95	2.26	3.18
10	4.54	7.36	6.39	5.40

What Do We See In Spark?

- These numbers are with vector-matrix multiplies

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	1.91	1.50	1.51	1.38
5	4.05	2.53	1.93	1.36
10	5.31	3.40	2.37	1.14

- Can do better when rewriting in terms of matrix-matrix multiplies

Batch Size \ D	100	500	1000	10000
1	1.00	1.00	1.00	1.00
2	0.97	1.51	0.92	1.28
5	2.67	2.95	2.26	3.18
10	4.54	7.36	6.39	5.40

A Tale Of 3 Optimizations

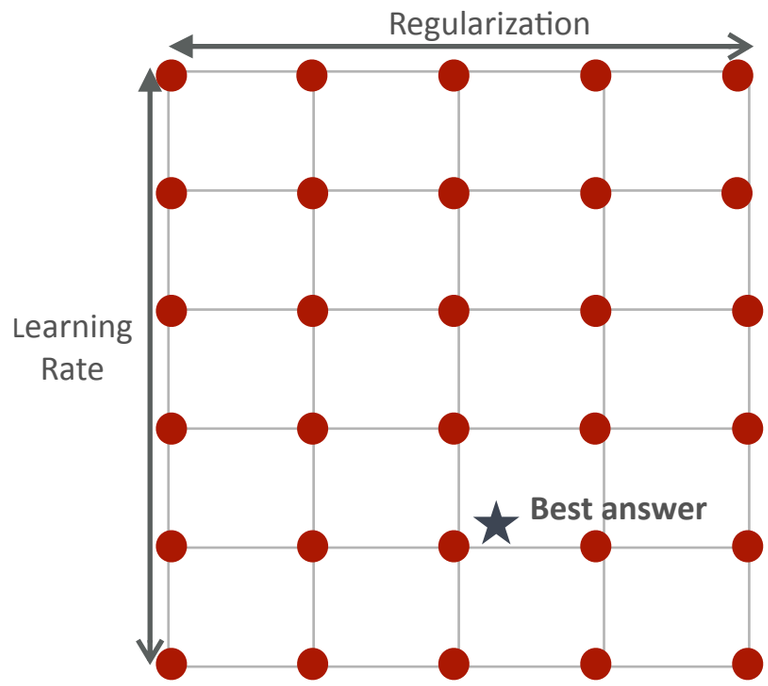
Better Resource Utilization

Algorithmic Speedups

Improved Search

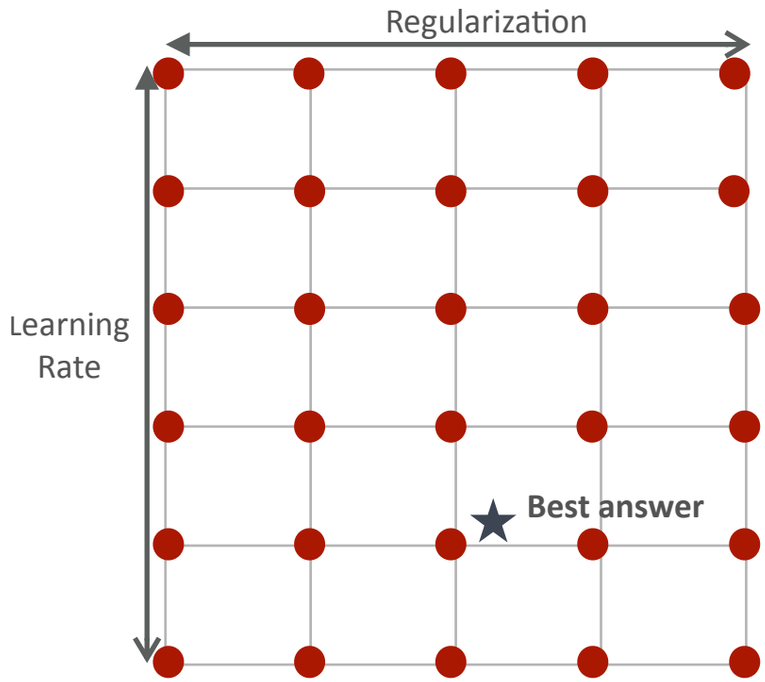
DATABRICKS

Algorithmic Speedups



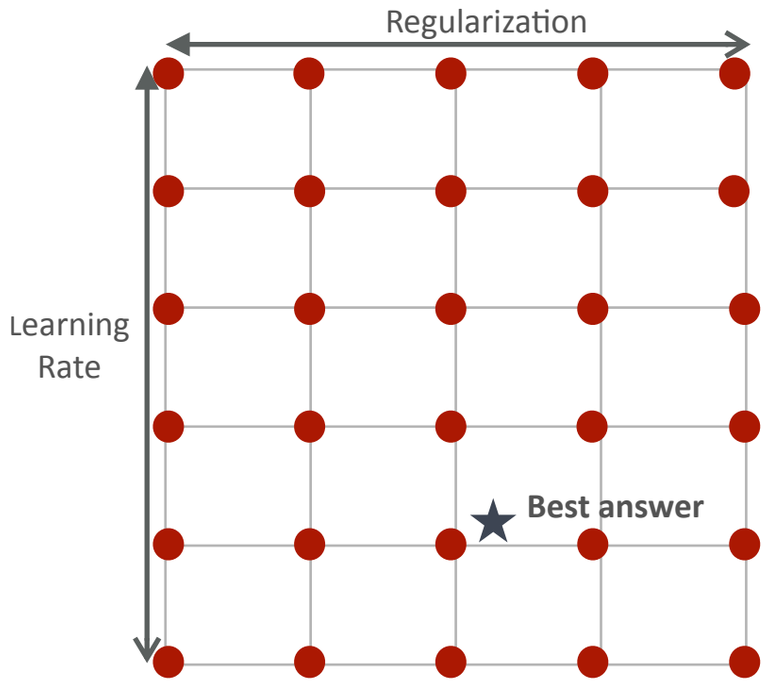
Algorithmic Speedups

- Each point in hyper-parameter space represents trained model



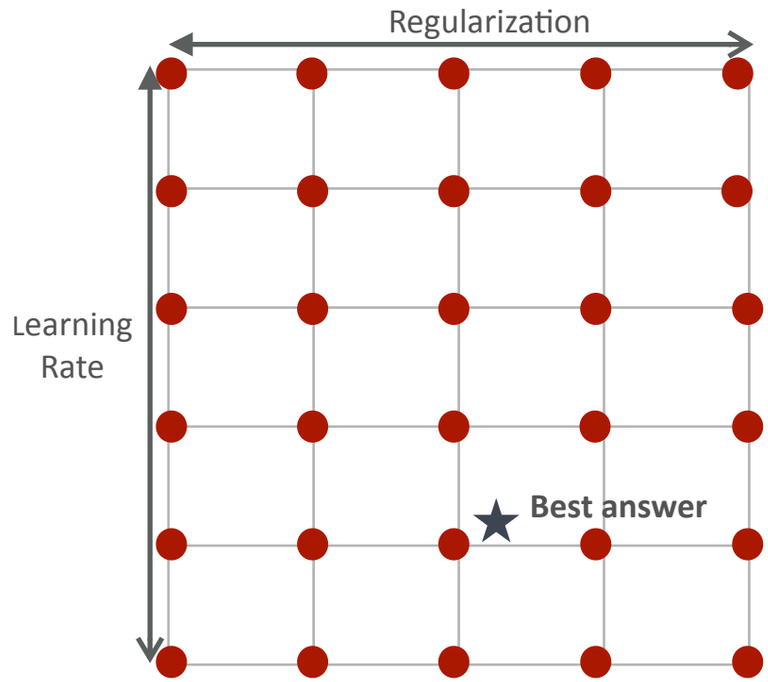
Algorithmic Speedups

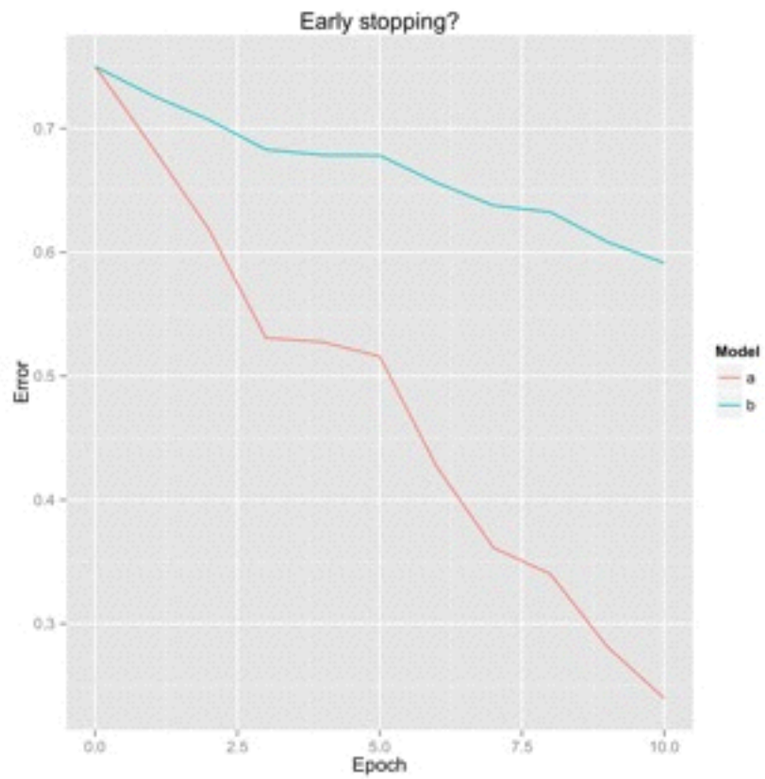
- Each point in hyper-parameter space represents trained model
- Sometimes we see early on that a model is no good



Algorithmic Speedups

- Each point in hyper-parameter space represents trained model
- Sometimes we see early on that a model is no good
- So we stop early - give up on models that are not progressing

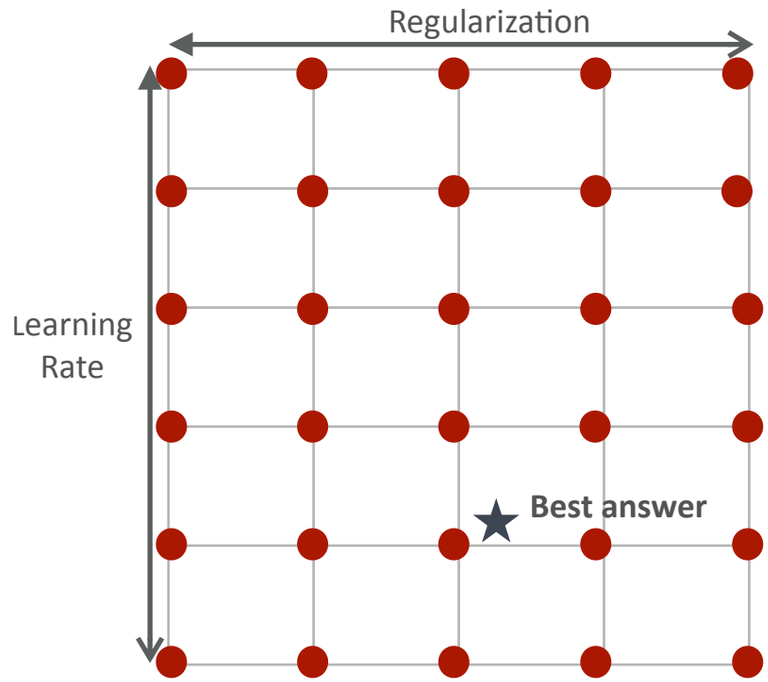


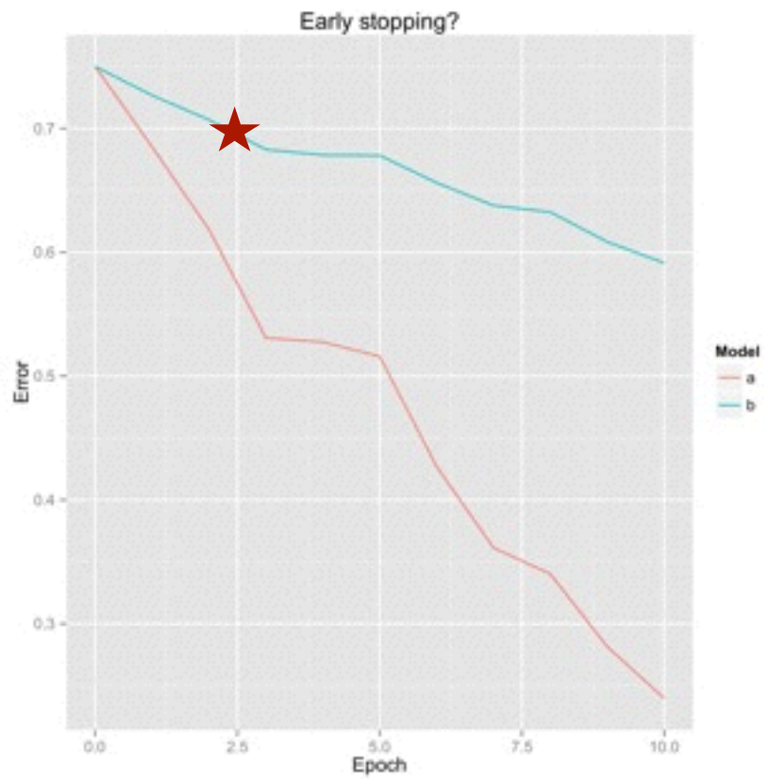


DATABRICKS

Algorithmic Speedups

- Each point in hyper-parameter space represents trained model
- Sometimes we see early on that a model is no good
- So we stop early - give up on models that are not progressing

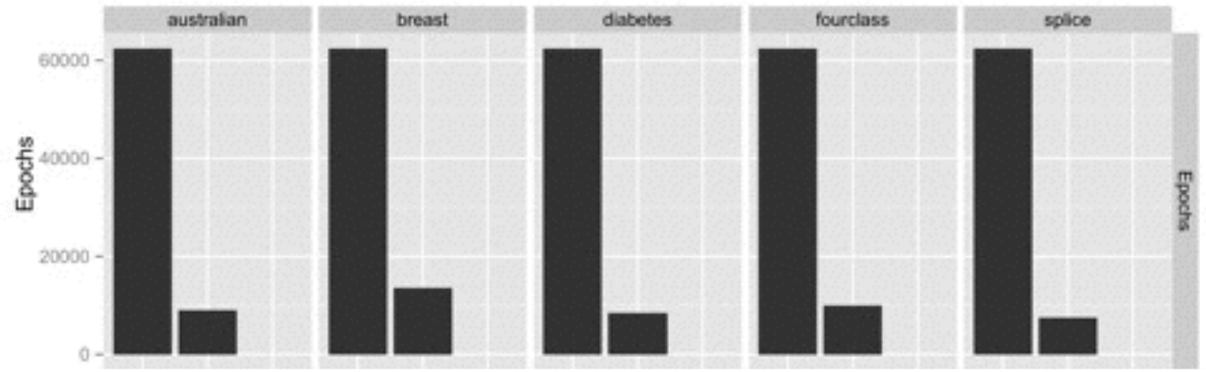




DATABRICKS

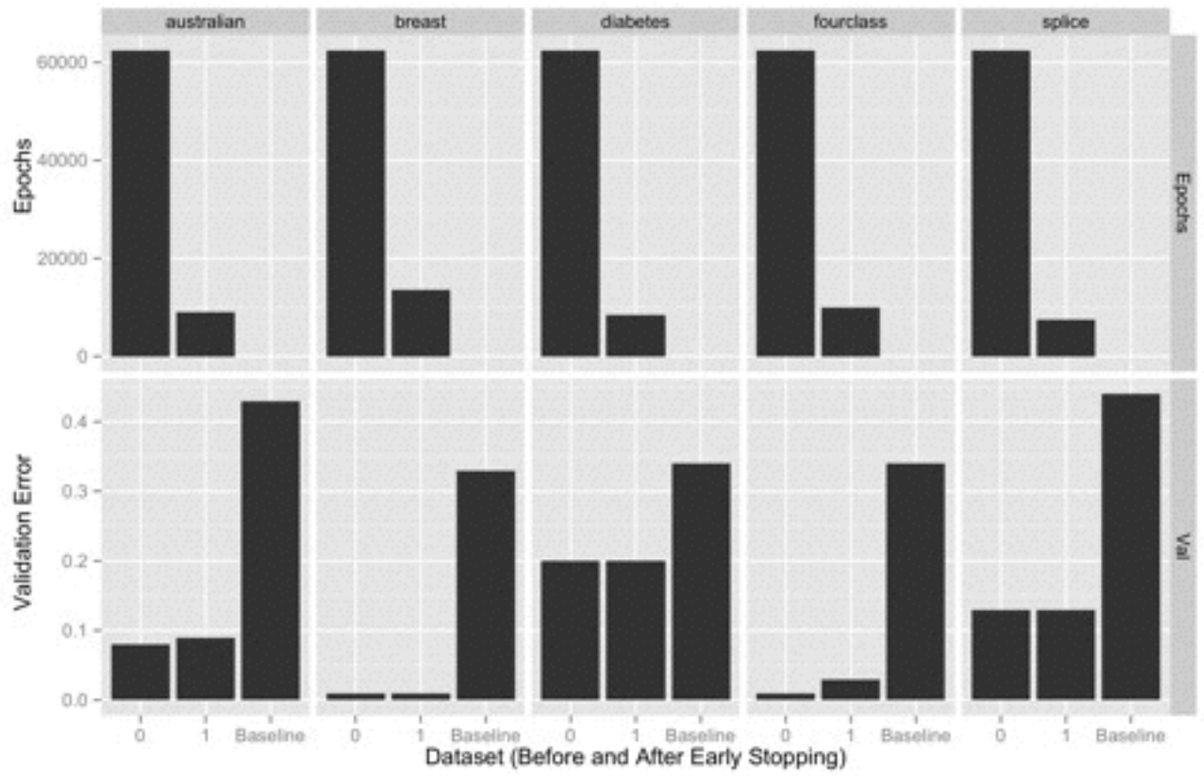
Algorithmic Speedups

- Each point in hyper-parameter space represents trained model
- Sometimes we see early on that a model is no good
- So we stop early - give up on models that are not progressing



Algorithmic Speedups

- Each point in hyper-parameter space represents trained model
- Sometimes we see early on that a model is no good
- So we stop early - give up on models that are not progressing



DATABRICKS

A Tale Of 3 Optimizations

Better Resource Utilization

Algorithmic Speedups

Improved Search

DATABRICKS

What Search Method?

What Search Method?

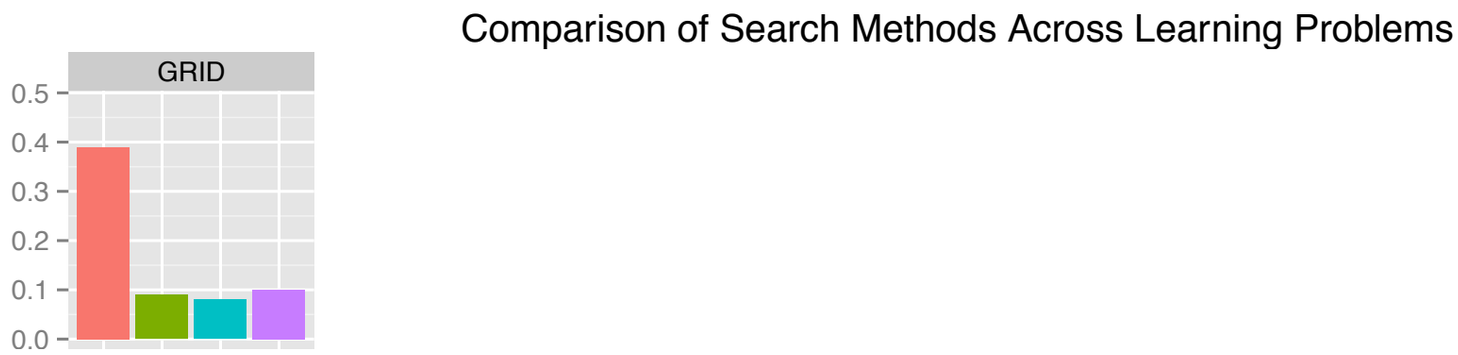
- Various derivative-free optimization techniques
- Simple ones (Grid, Random)
- Classic Derivative-Free (Nelder-Mead, Powell's method)
- Bayesian (SMAC, TPE, Spearmint)

What Search Method?

- Various derivative-free optimization techniques
  - Simple ones (Grid, Random)
  - Classic Derivative-Free (Nelder-Mead, Powell's method)
  - Bayesian (SMAC, TPE, Spearmint)

- What should we do?
  - Tried on 5 datasets, optimized over 4 hyperparameters!

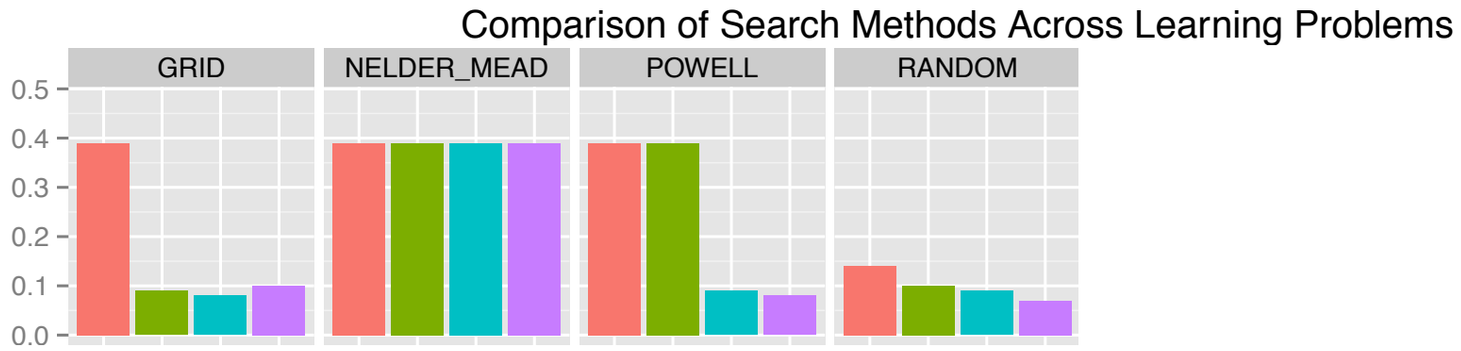
What Search Method?



- Various derivative-free optimization techniques
  - Simple ones (Grid, Random)
  - Classic Derivative-Free (Nelder-Mead, Powell's method)
  - Bayesian (SMAC, TPE, Spearmint)

- What should we do?
  - Tried on 5 datasets, optimized over 4 hyperparameters!

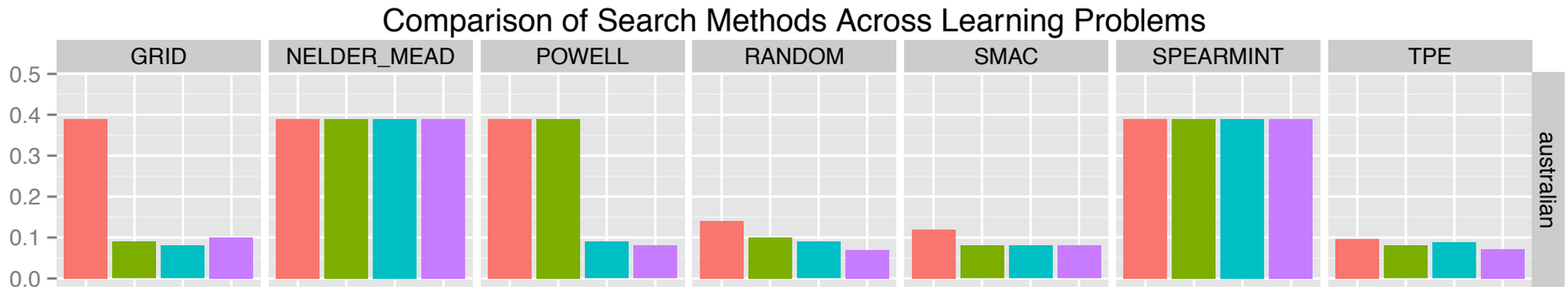
What Search Method?



- Various derivative-free optimization techniques
  - Simple ones (Grid, Random)
  - Classic Derivative-Free (Nelder-Mead, Powell's method)
  - Bayesian (SMAC, TPE, Spearmint)

- What should we do?
  - Tried on 5 datasets, optimized over 4 hyperparameters!

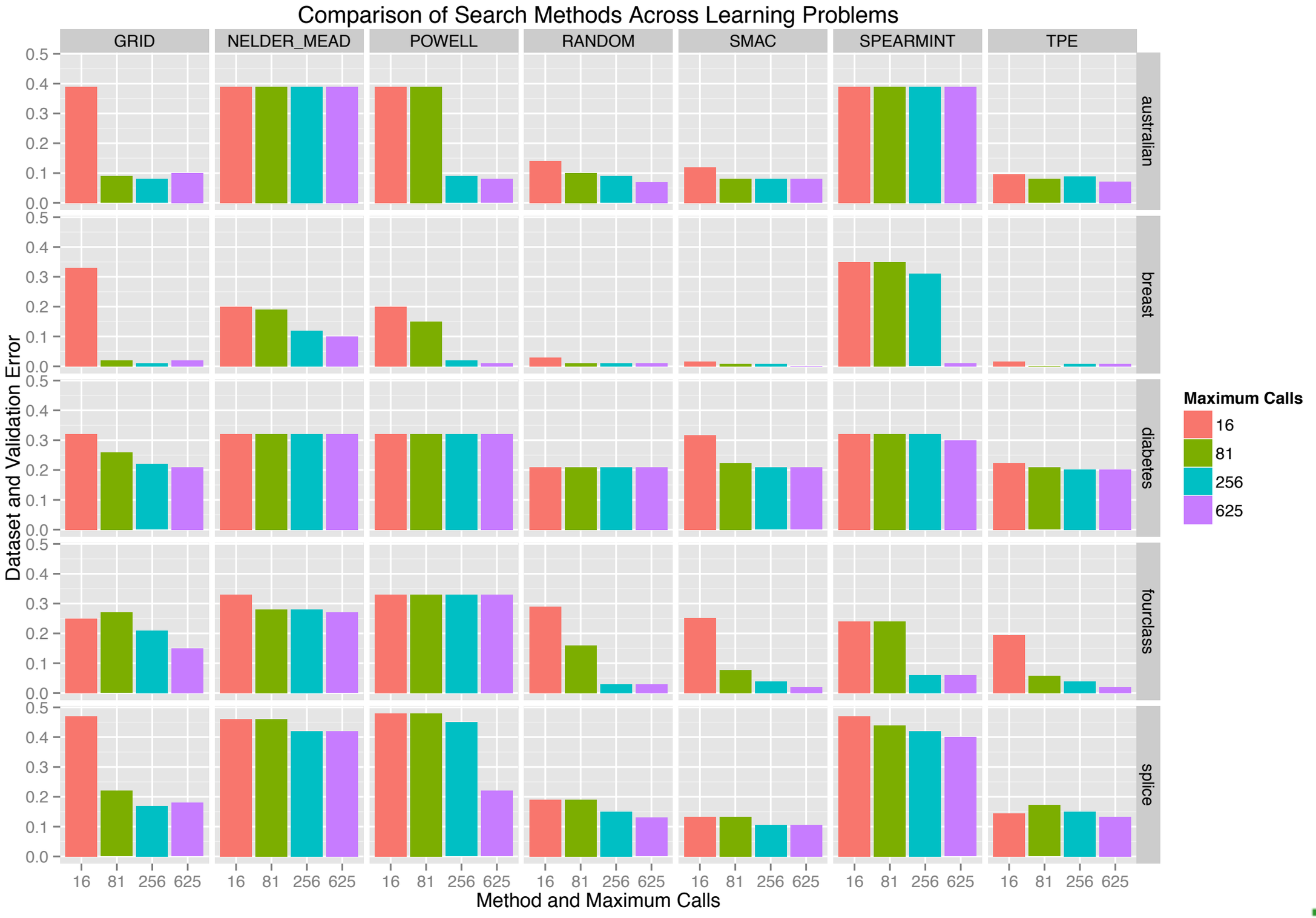
What Search Method?



- Various derivative-free optimization techniques
  - Simple ones (Grid, Random)
  - Classic Derivative-Free (Nelder-Mead, Powell's method)
  - Bayesian (SMAC, TPE, Spearmint)

- What should we do?
  - Tried on 5 datasets, optimized over 4 hyperparameters!

What Search Method?



DATABRICKS

Putting It All Together

Putting It All Together

- First version of MLbase optimizer

Putting It All Together

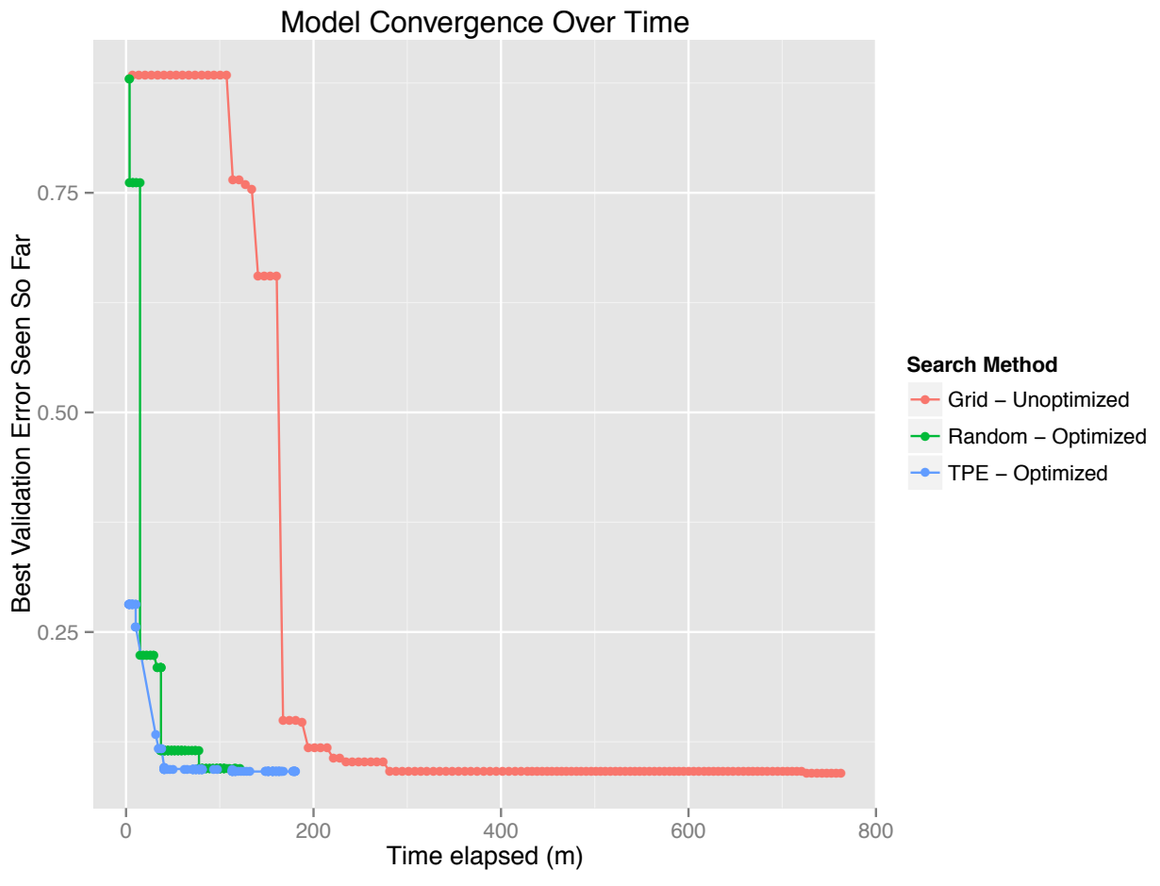
- First version of MLbase optimizer
- 30GB dense images (240K x 16K)
- 2 model families, 5 hyperparams

Putting It All Together

- First version of MLbase optimizer
- 30GB dense images (240K x 16K)
- 2 model families, 5 hyperparams
- Baseline: grid search
- Our method: combination of
  - Batching
  - Early stopping
  - Random or TPE

Putting It All Together

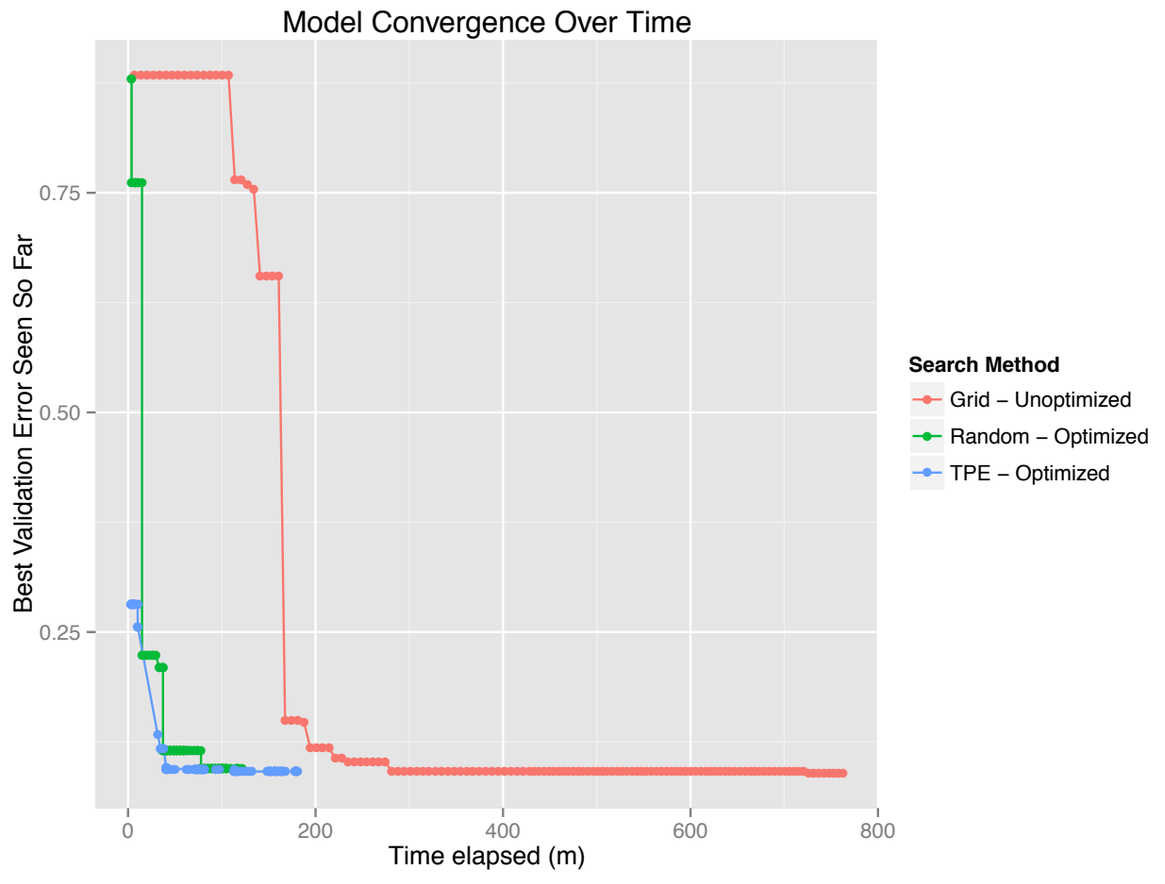
- First version of MLbase optimizer
- 30GB dense images (240K x 16K)
- 2 model families, 5 hyperparams
- Baseline: grid search
- Our method: combination of
  - Batching
  - Early stopping
  - Random or TPE



DATABRICKS

Putting It All Together

- First version of MLbase optimizer
- 30GB dense images (240K x 16K)
- 2 model families, 5 hyperparams
- Baseline: grid search
- Our method: combination of
  - Batching
  - Early stopping
  - Random or TPE

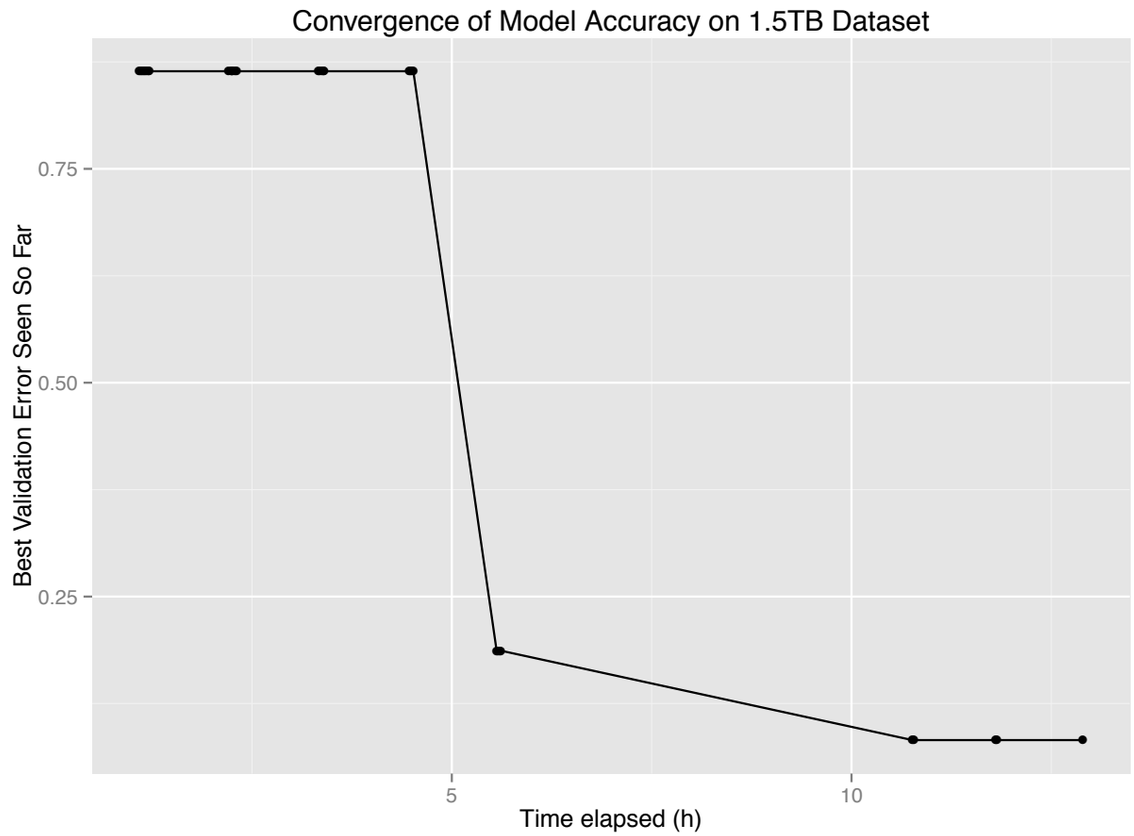


20x speedup compared to grid search

15 minutes vs 5 hours!

DATABRICKS

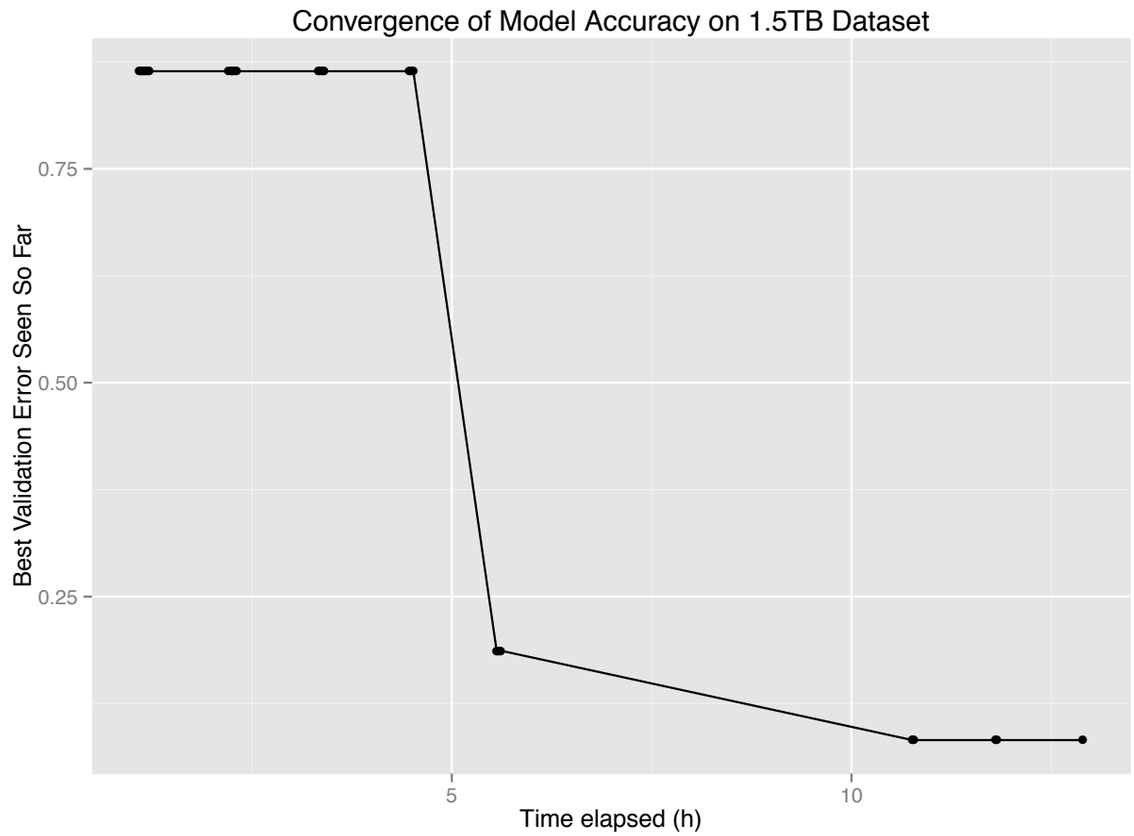
Does It Scale?



DATABRICKS

Does It Scale?

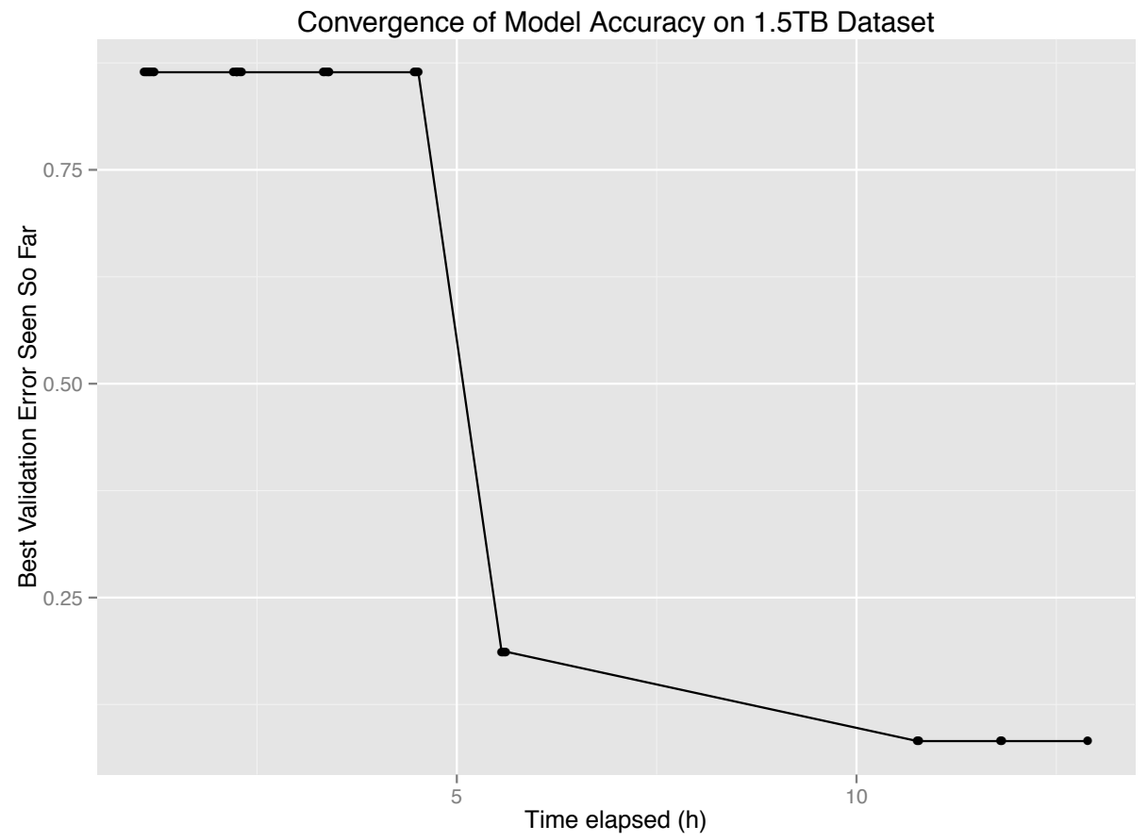
- 1.5TB dataset (1.2M x 160K)



DATABRICKS

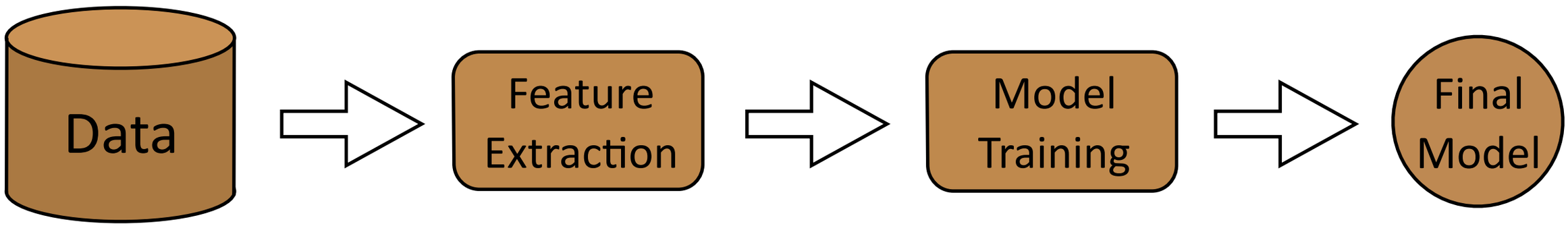
Does It Scale?

- 1.5TB dataset (1.2M x 160K)
- 128 nodes, thousands of passes over data
- Tried 32 models in 15 hours
  - Good answer after 11 hours

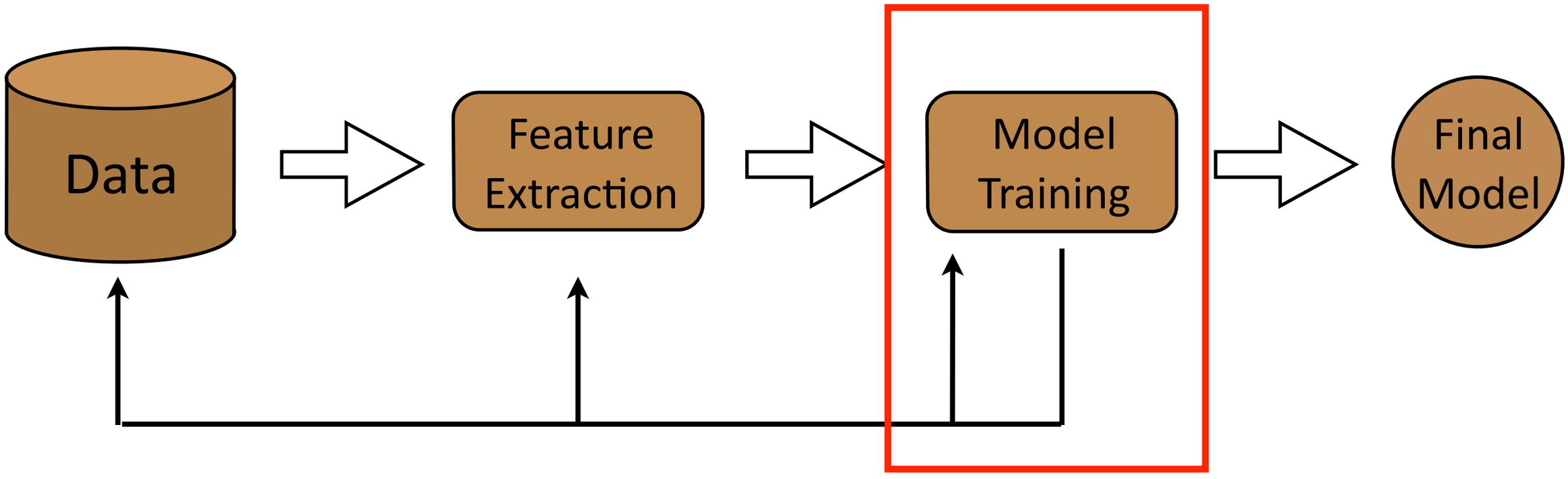


DATABRICKS

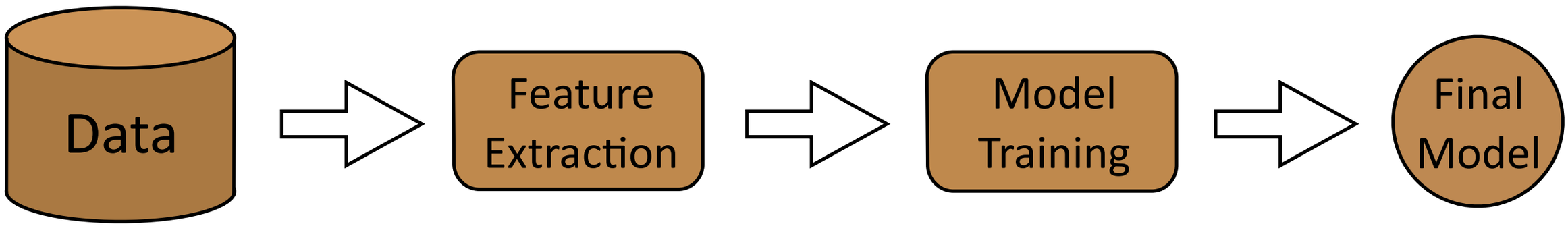
Future Work



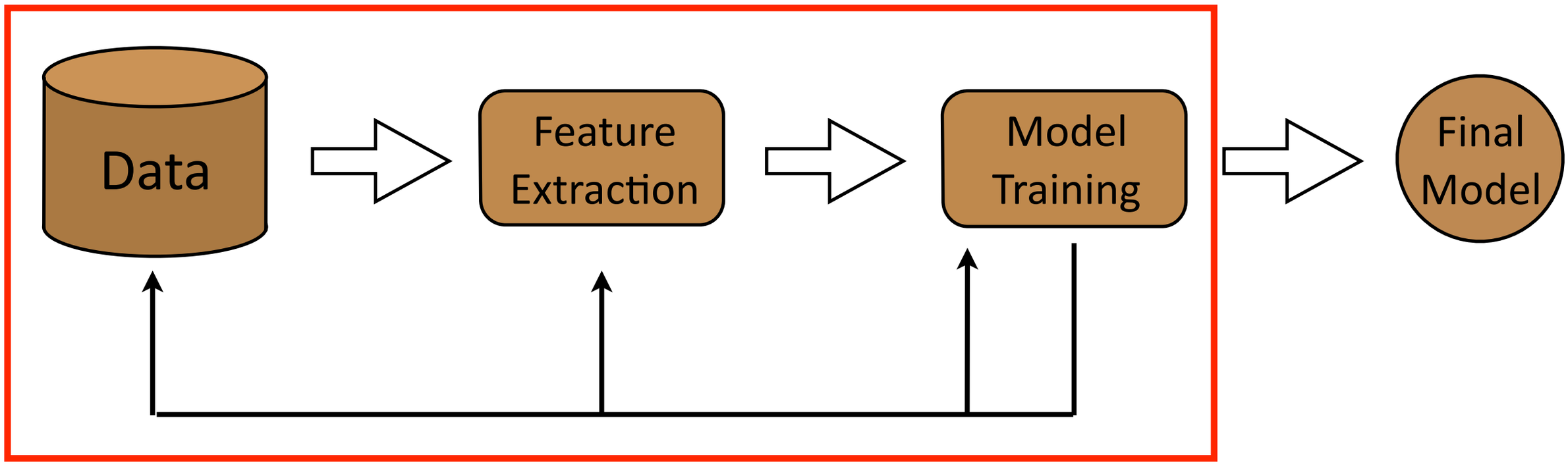
Future Work



Future Work



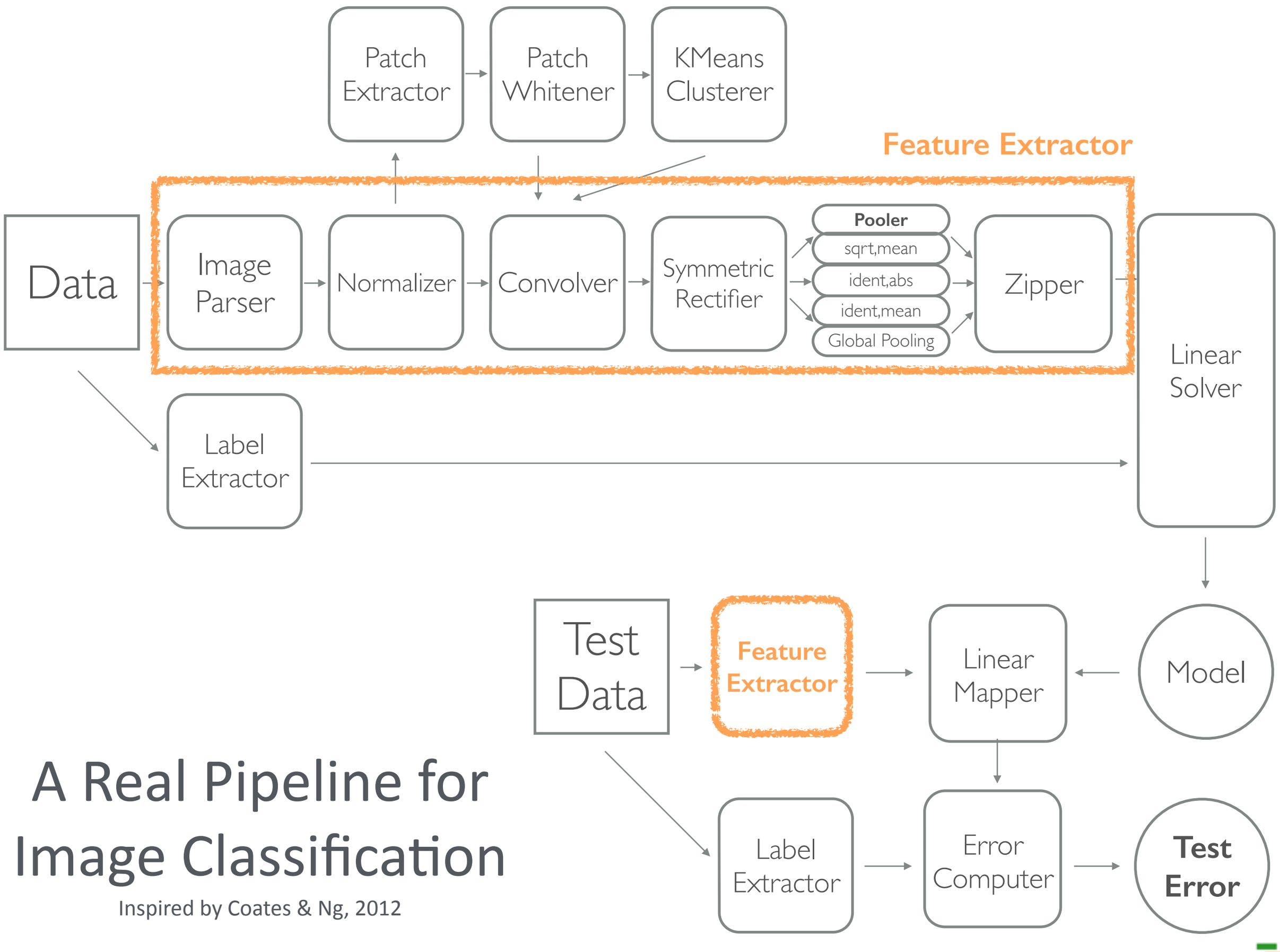
Future Work



Automated ML Pipeline Construction

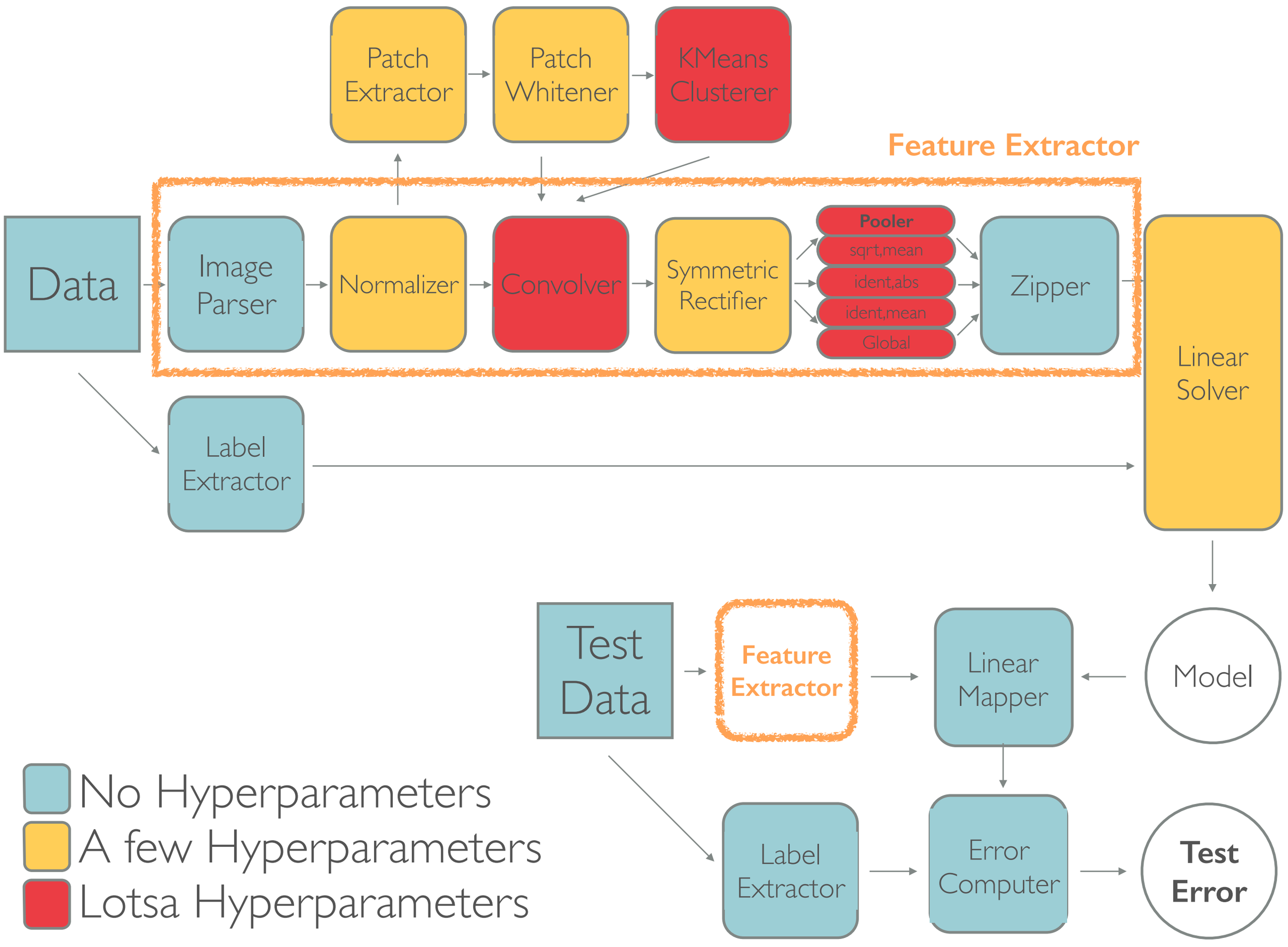
A Real Pipeline for Image Classification

Inspired by Coates & Ng, 2012



Slide courtesy of Evan Sparks

DATABRICKS



Slide courtesy of Evan Sparks

DATABRICKS

Other Future Work

Other Future Work

- ✦ Ensembling

Other Future Work

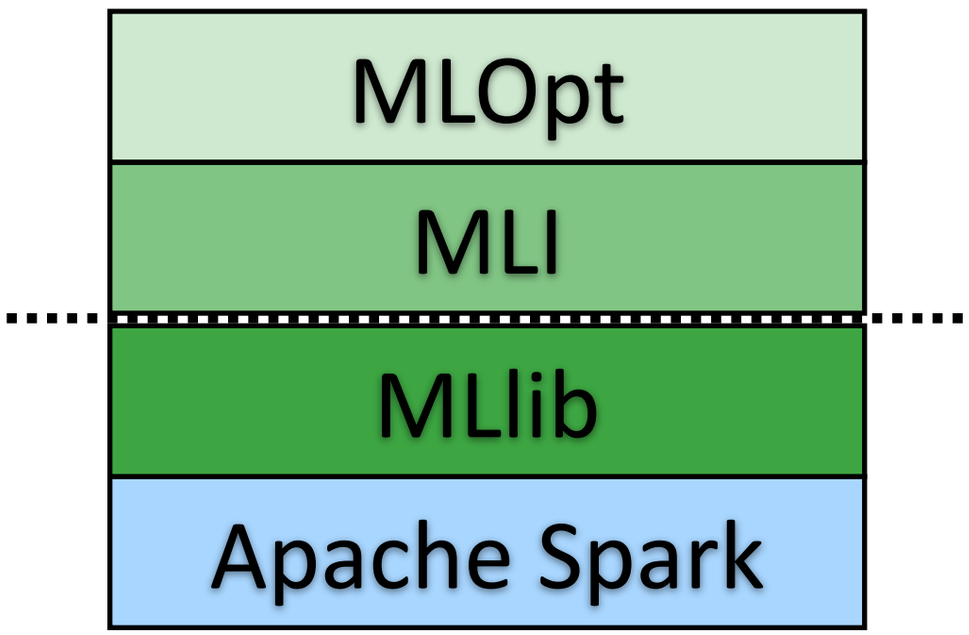
- Ensembling
- Leverage sampling

Other Future Work

- Ensembling
- Leverage sampling
- Better parallelism for smaller datasets

Other Future Work

- Ensembling
- Leverage sampling
- Better parallelism for smaller datasets
- Multiple hypothesis testing issues



MLOpt: Declarative layer that aims to automate ML pipeline construction

MLlib: Spark's core ML library

MLI: API to simplify ML development

Spark: Cluster computing system designed for iterative computation

THANKS!
QUESTIONS?

www.mlbase.org

DATABRICKS