Amazon SageMaker

Edo Liberty, Amazon AI Labs

Alex Smola, Zohar Karnin, Bing Xiang, Baris Cuskon, Ramesh Nallapati, Phillip Gautier, Madhav Jha, Ran Ding, Tim Januschowski, David Selinas, Bernie Wang, Jan Gasthaus, Laurence Rouesnel, Amir Sadoughi, Piali Das, Julio Delgado Mangas, Yury Astashonok, Can Balioglu, Saswata Chakravarty

1) ML Algorithms in The Cloud - New Challenges

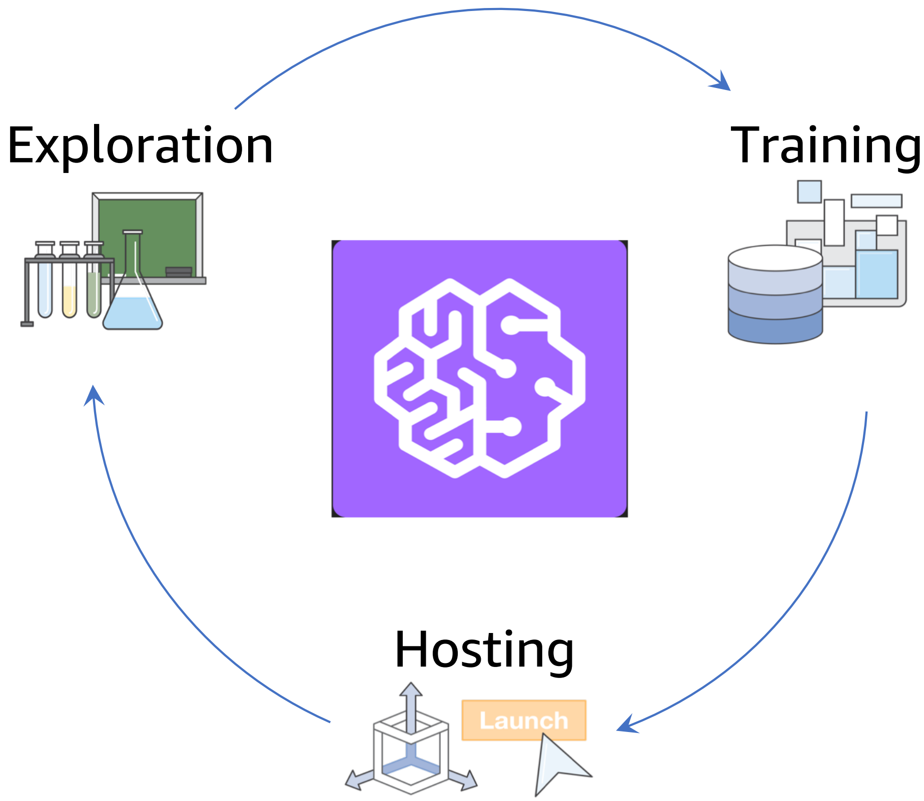
2) SageMaker Algorithms - Architecture and Data Flow

3) Science of Streaming Algorithms – Advantages and Challenges

4) SageMaker Algorithms – Accurate, Fast, Scalable, and Easy to Use.

ML Algorithms in The Cloud – New Challenges

Lifecycle of a Machine Learning Project



Small Data - Machine Learning

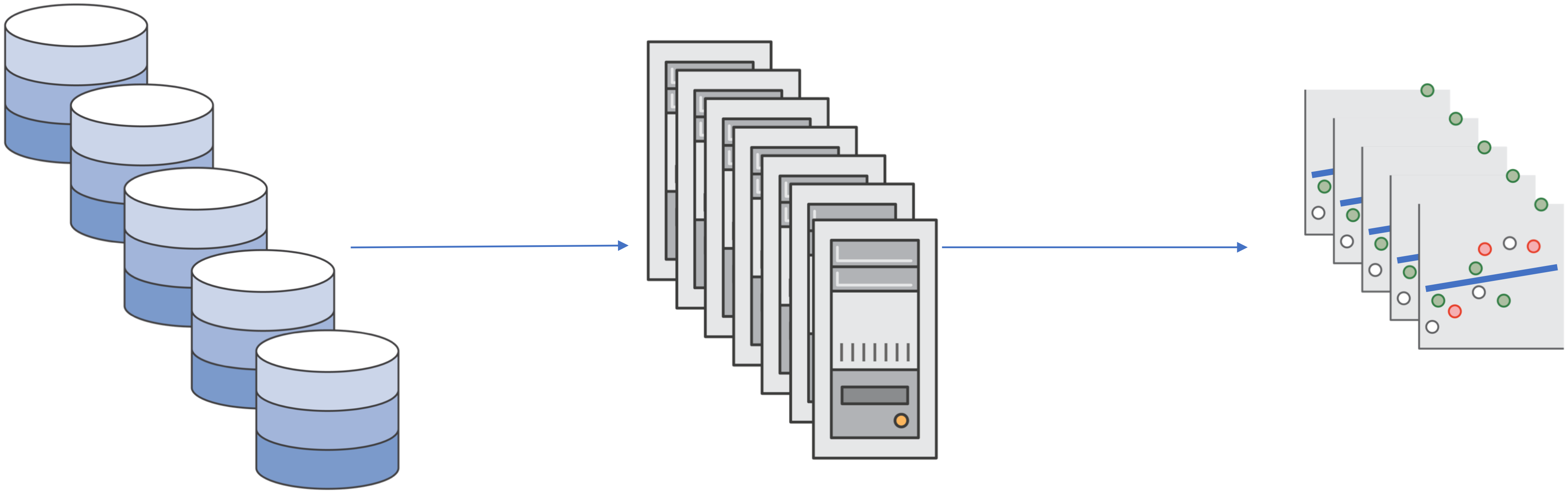
Our Customers use ML at massive scale!

“Our data warehouse is 100TB and we are processing 2TB daily. We're running mostly gradient boosting (trees), LDA and K-Means clustering and collaborative filtering.“
Shahar Cizer Kobrinsky, VP Architecture

“We process 3 million ad requests a second, 100,000 features per request. That’s 250 trillion per day. Not your run of the mill Data science problem!”
Bill Simmons, CTO

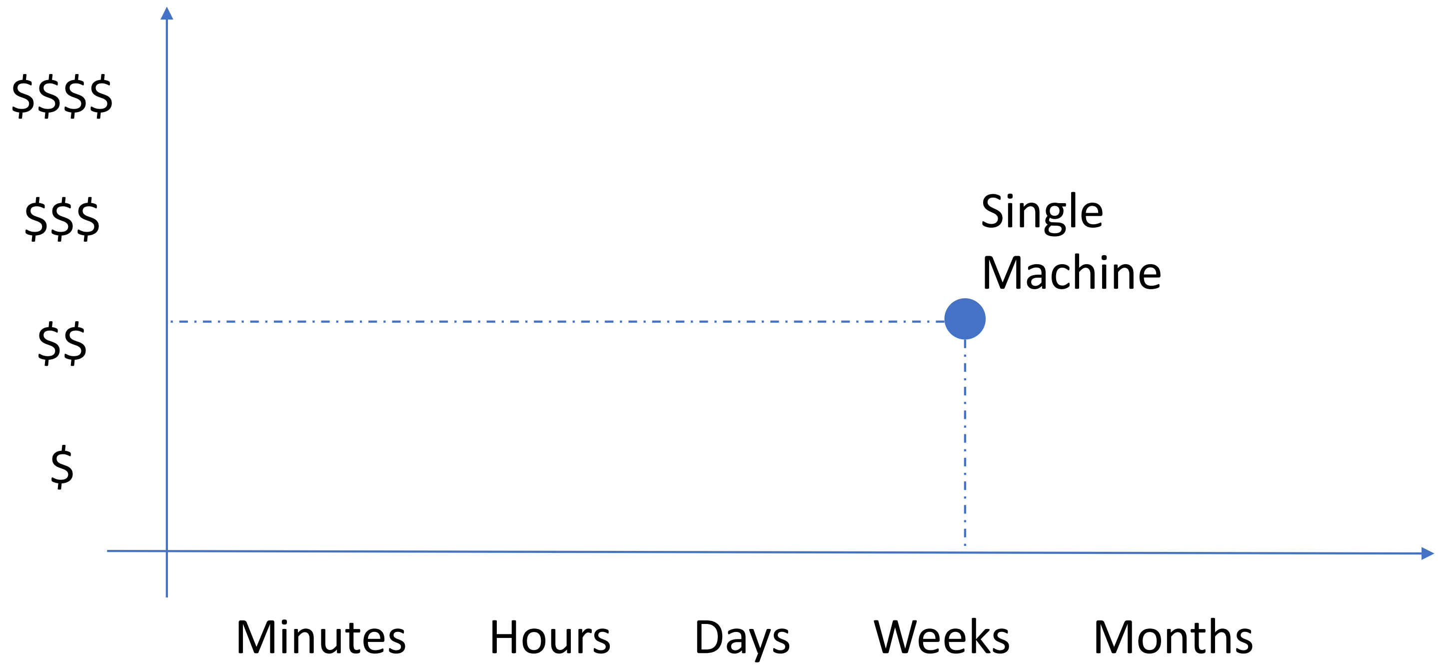
“We collect 160M events daily in the ML pipeline and run training over the last 15 days and need it to complete in one hour. Effectively there's 100M features in the model”
Valentino Volonghi, CTO

Large Scale Machine Learning

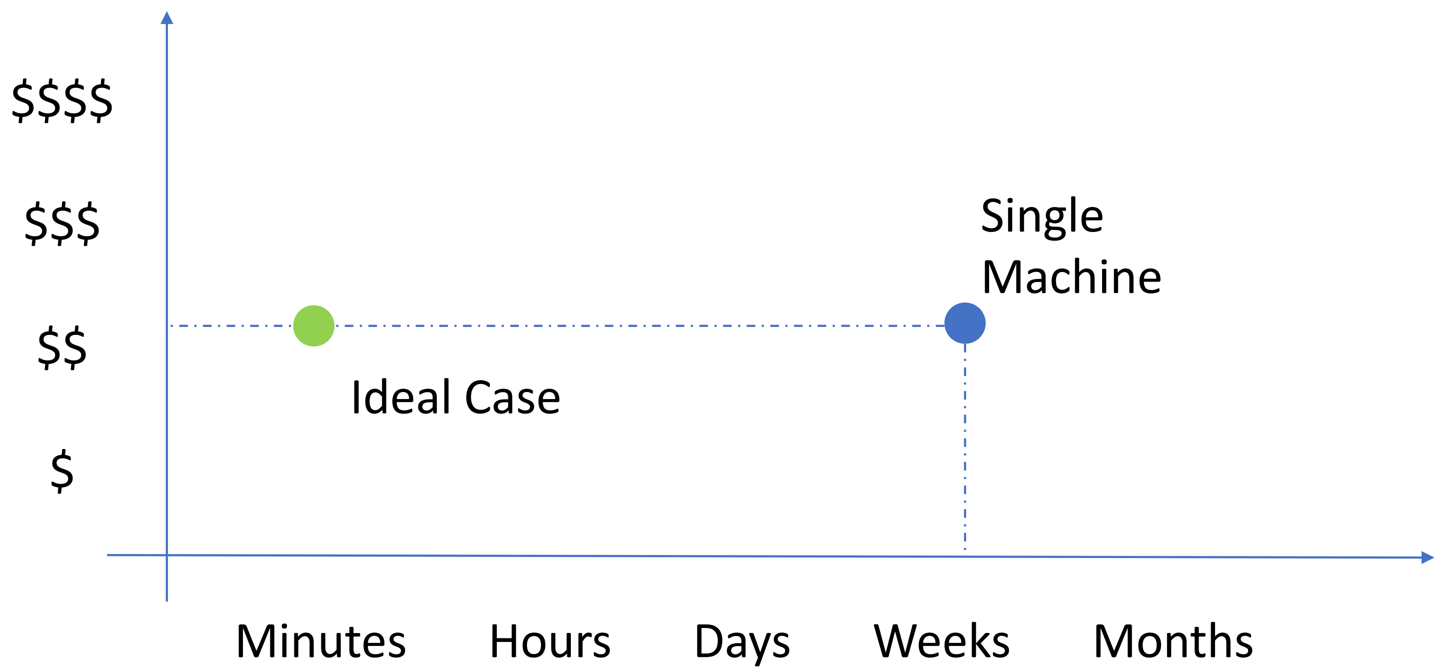


Large Scale Machine Learning

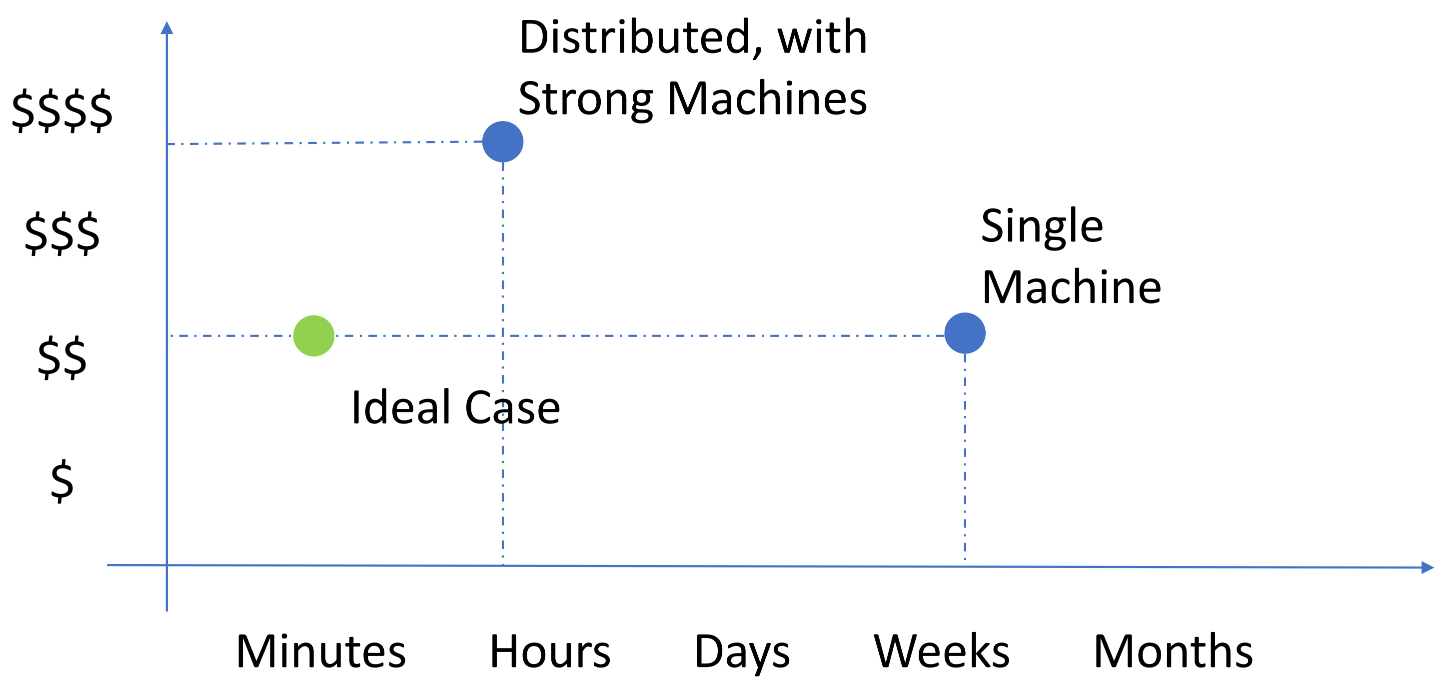
Cost vs. Time



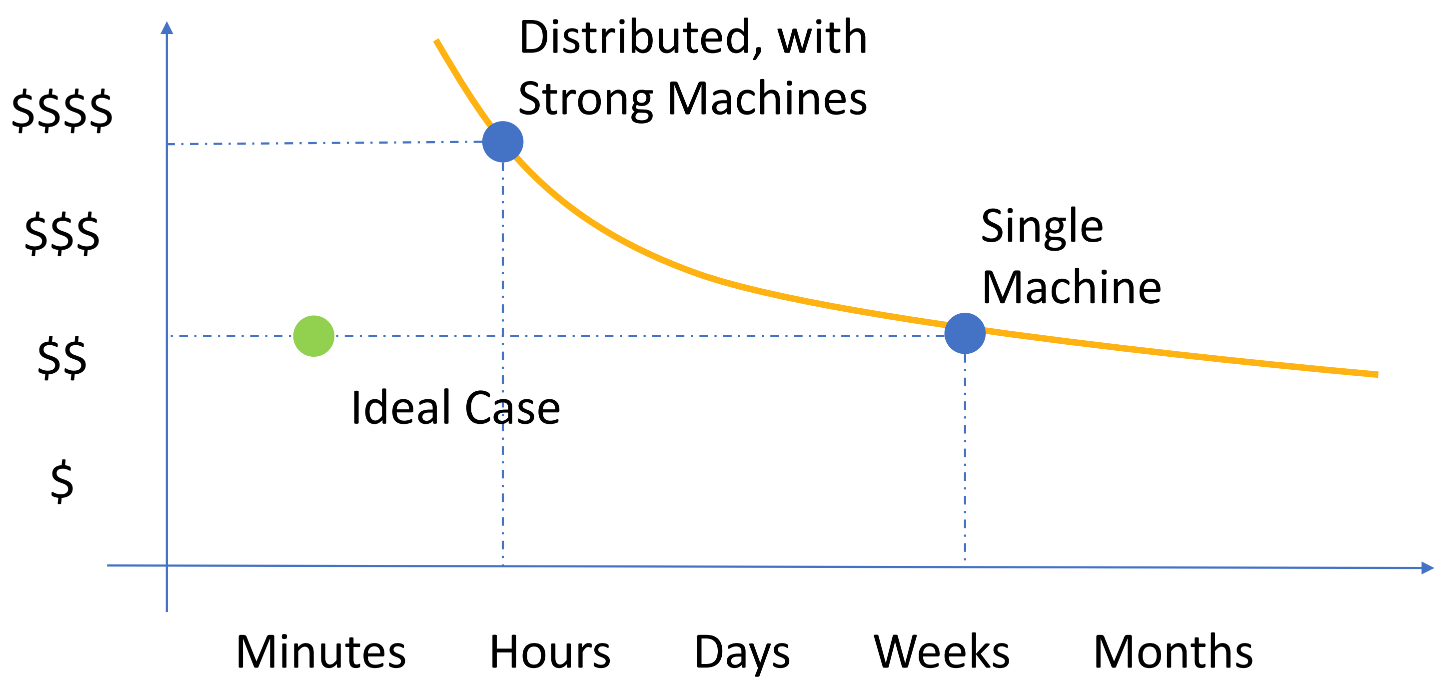
Cost vs. Time



Cost vs. Time

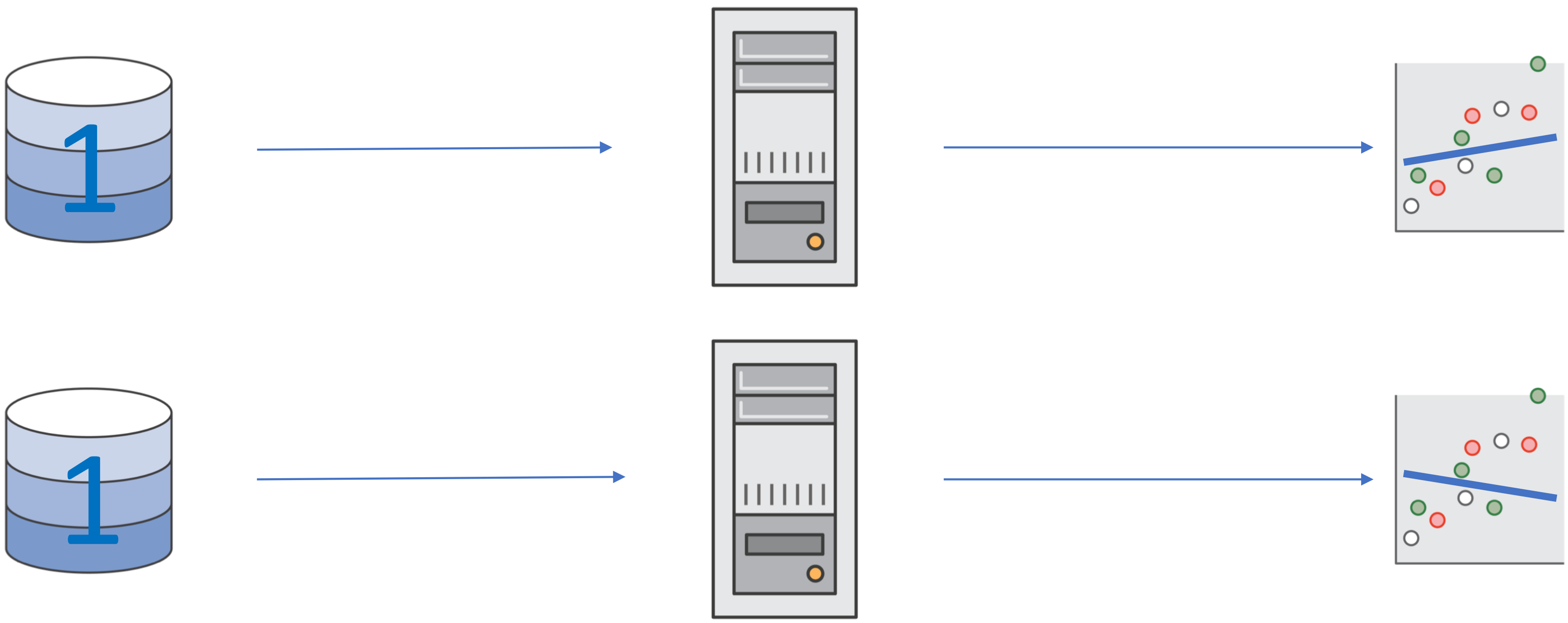

Distributed, with
Strong Machines
$$$$
$$$
$$
$
Single
Machine
Ideal Case
Minutes
Hours
Days
Weeks
Months

Cost vs. Time

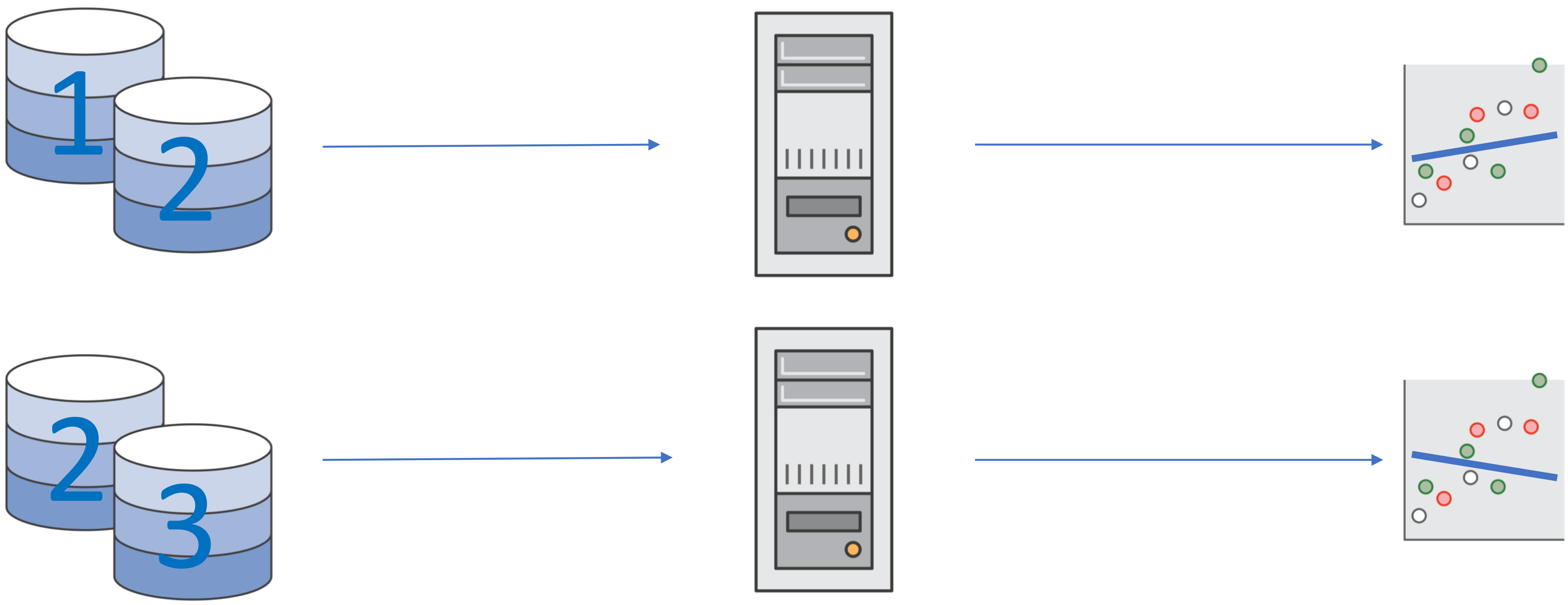


Distributed, with
Strong Machines
$$$$
$$$
$$
$
Single
Machine
Ideal Case
Minutes
Hours
Days
Weeks
Months

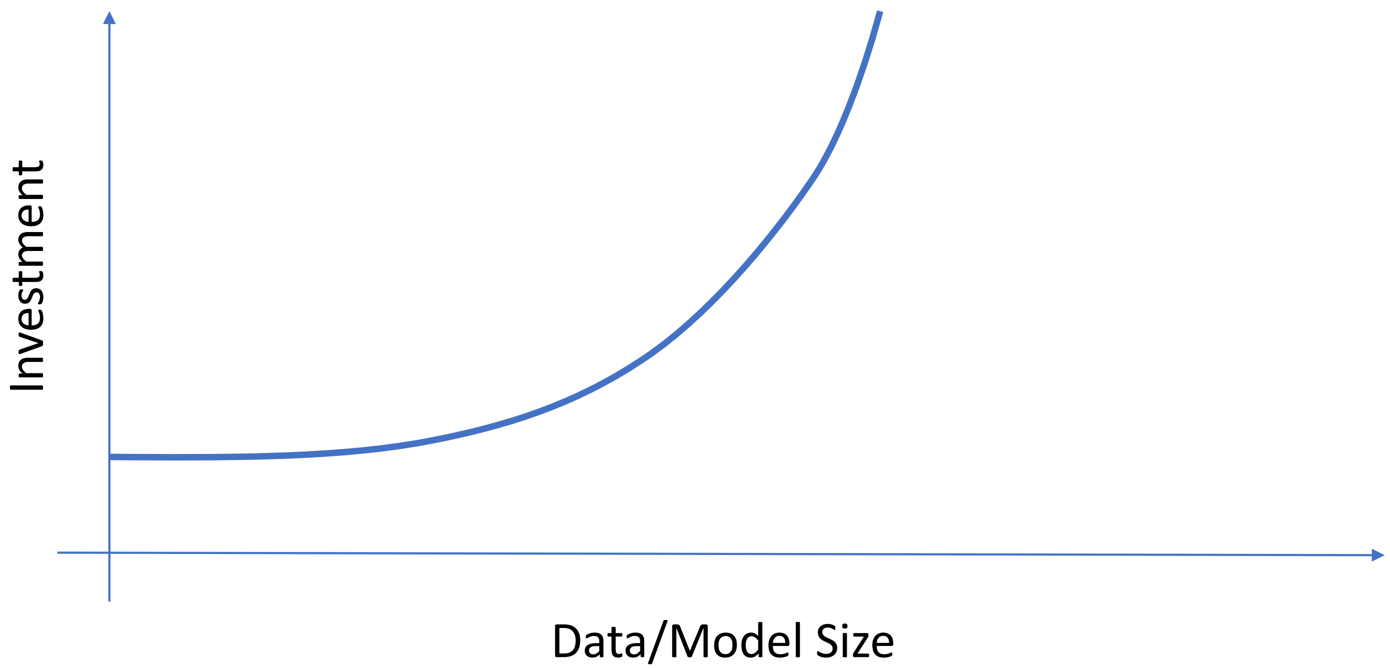
Model Selection



Incremental Training

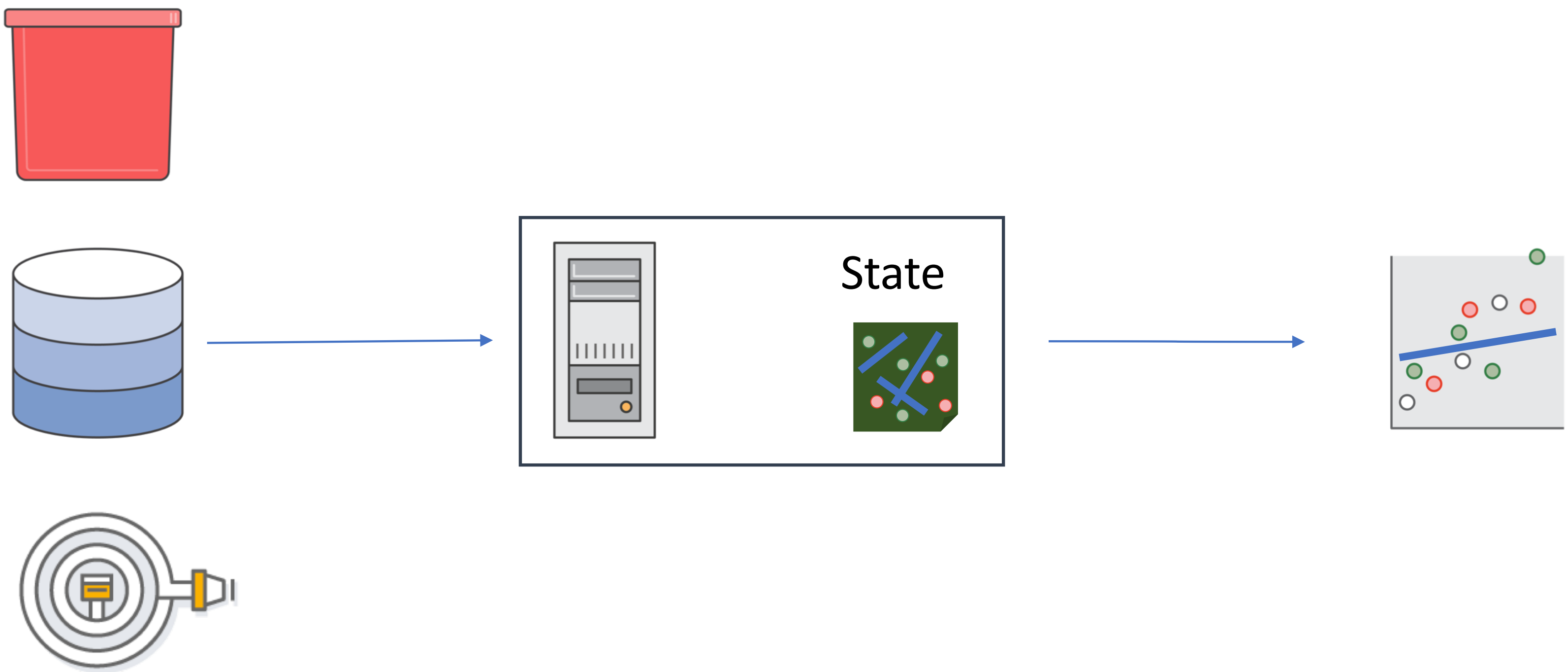


Production Readiness

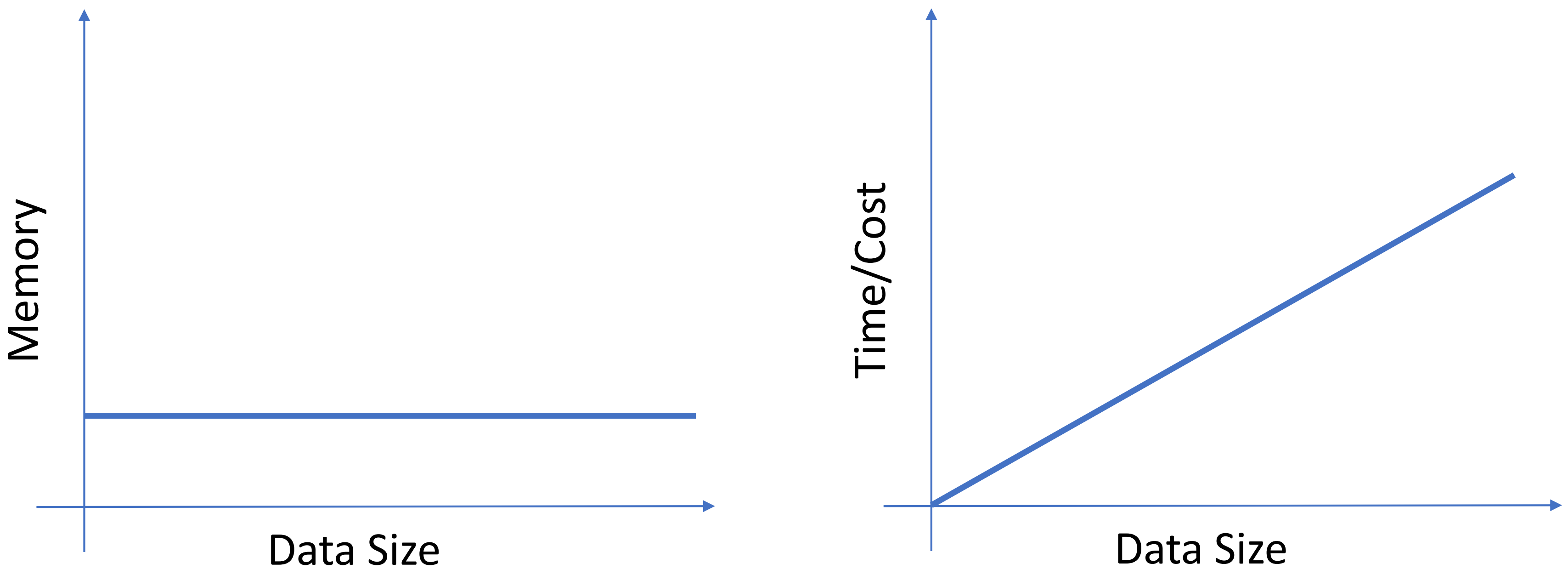


SageMaker Algorithms - Architecture and Data Flow

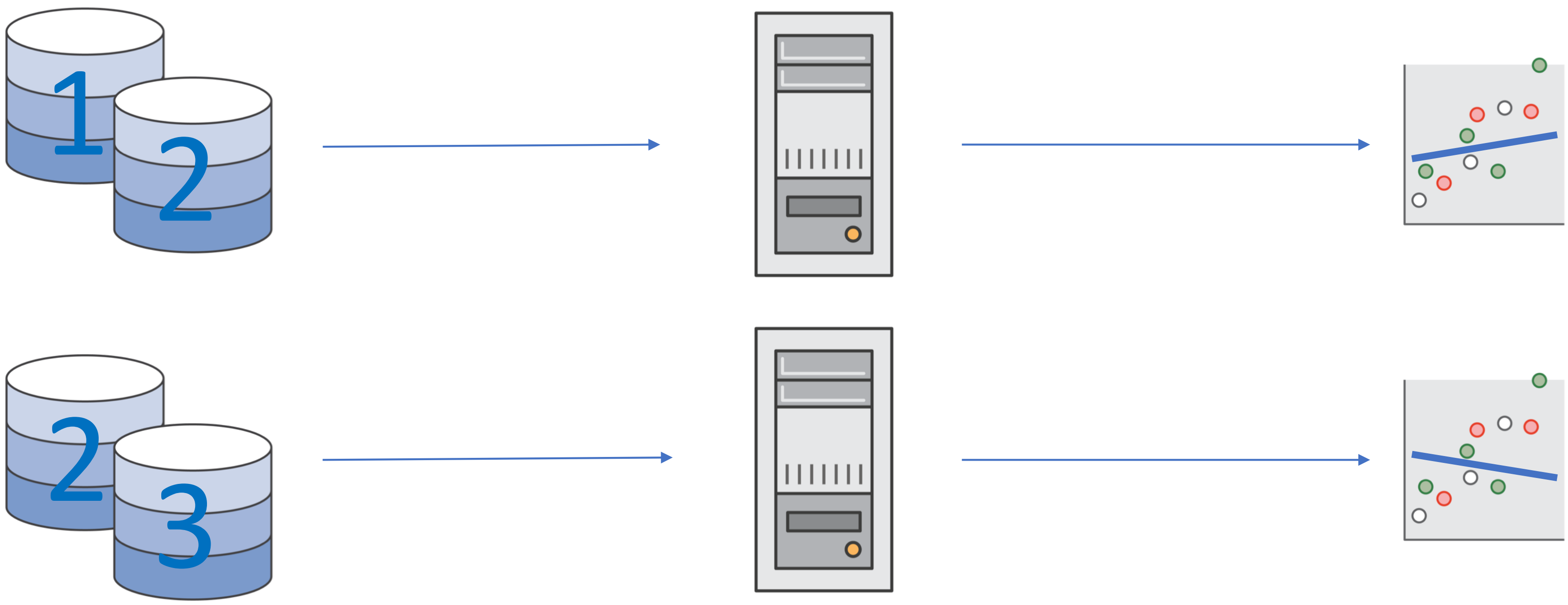
Streaming



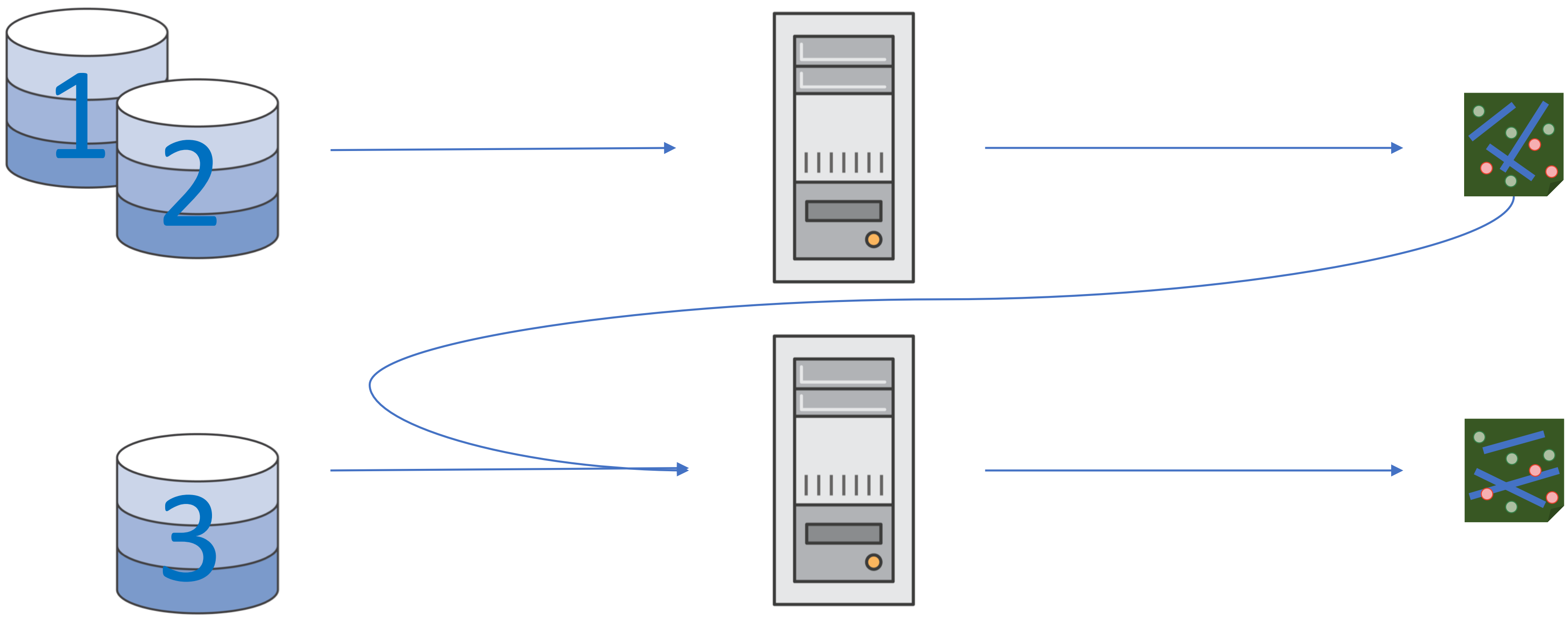
Streaming



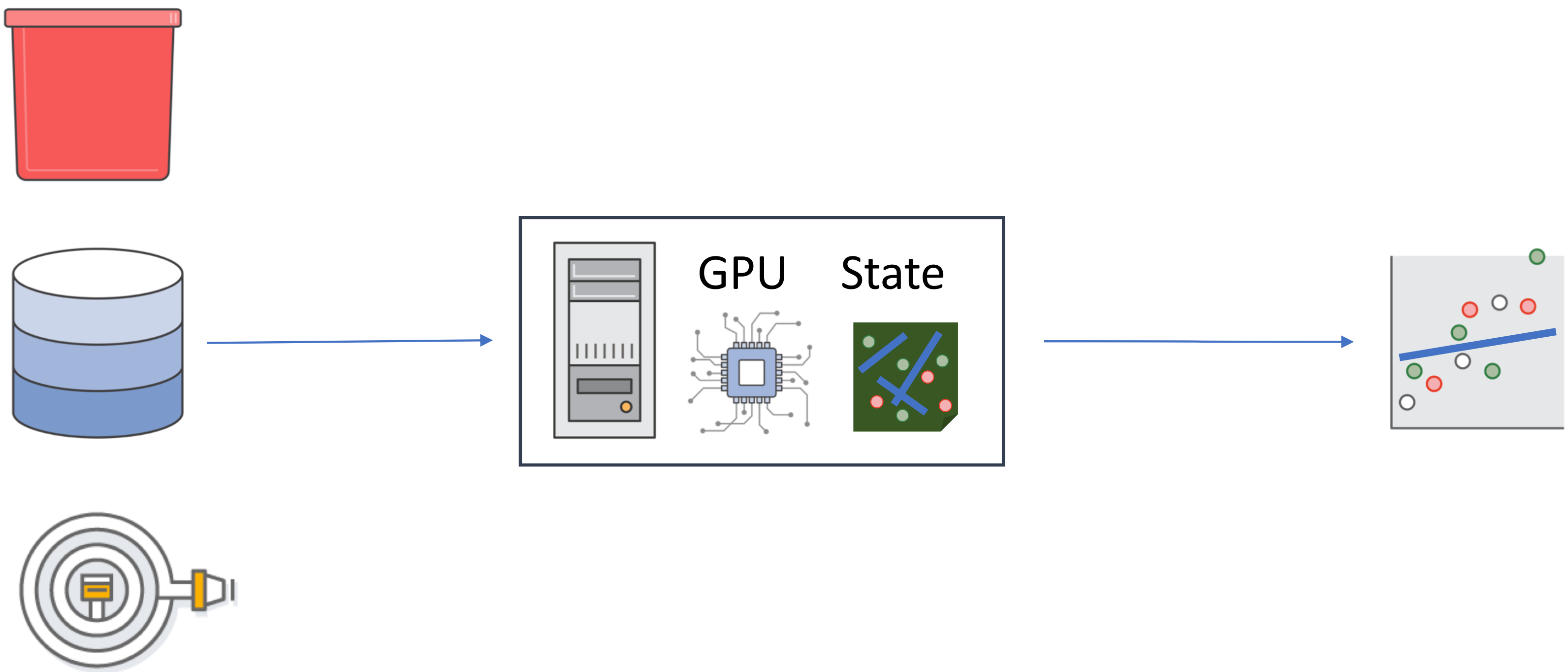
Incremental Training



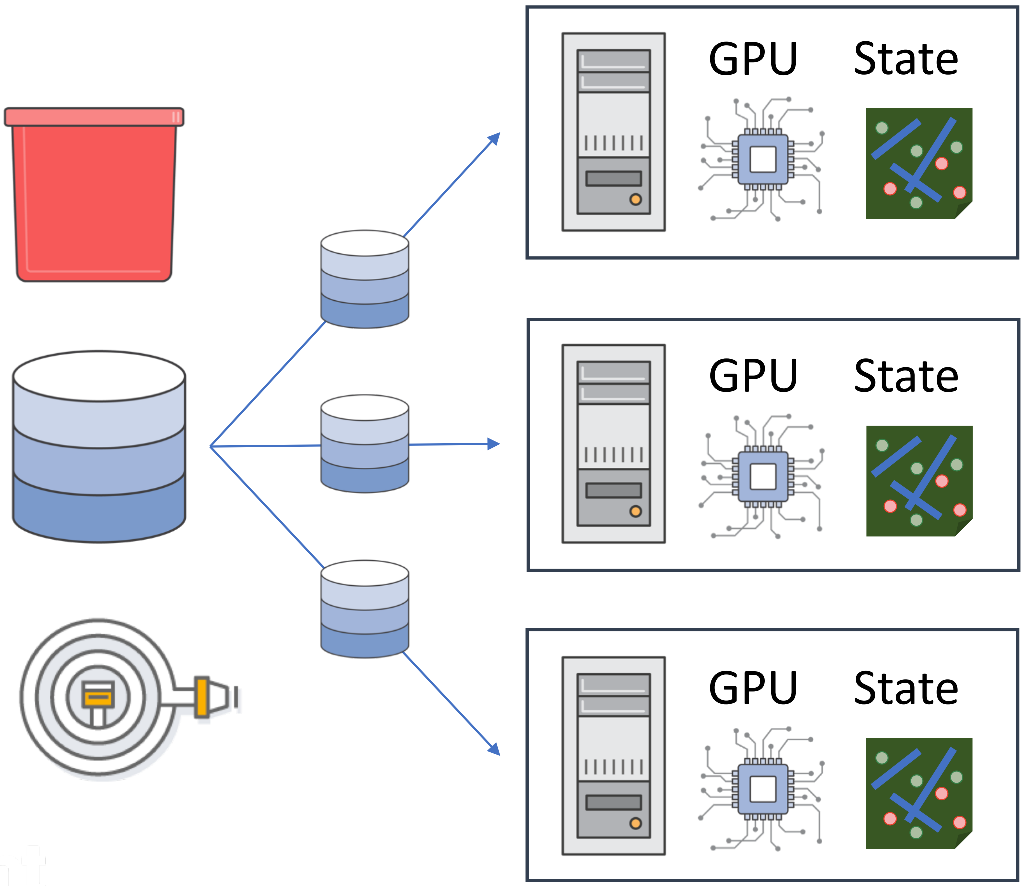
Incremental Training



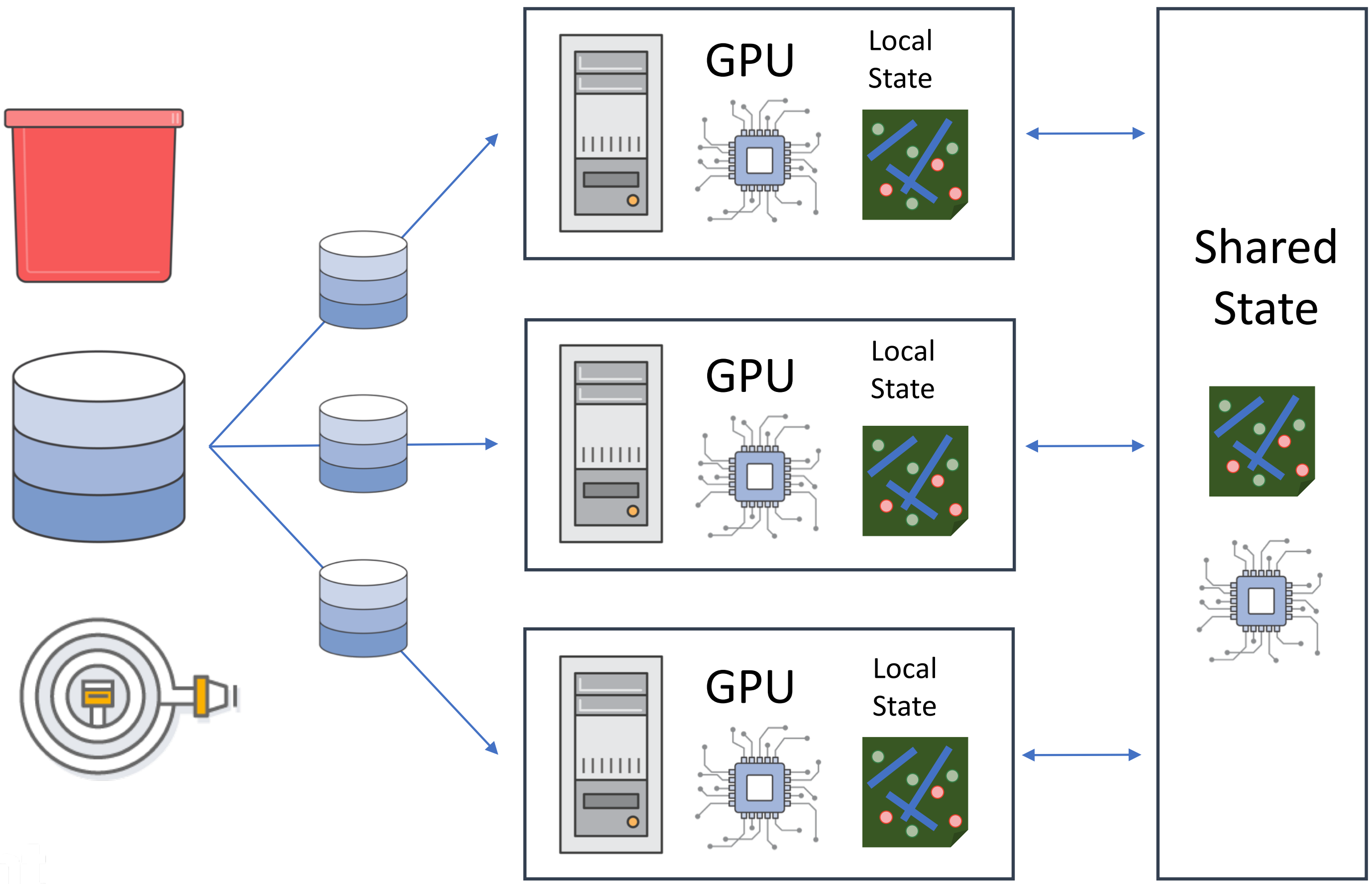
GPU/CPU



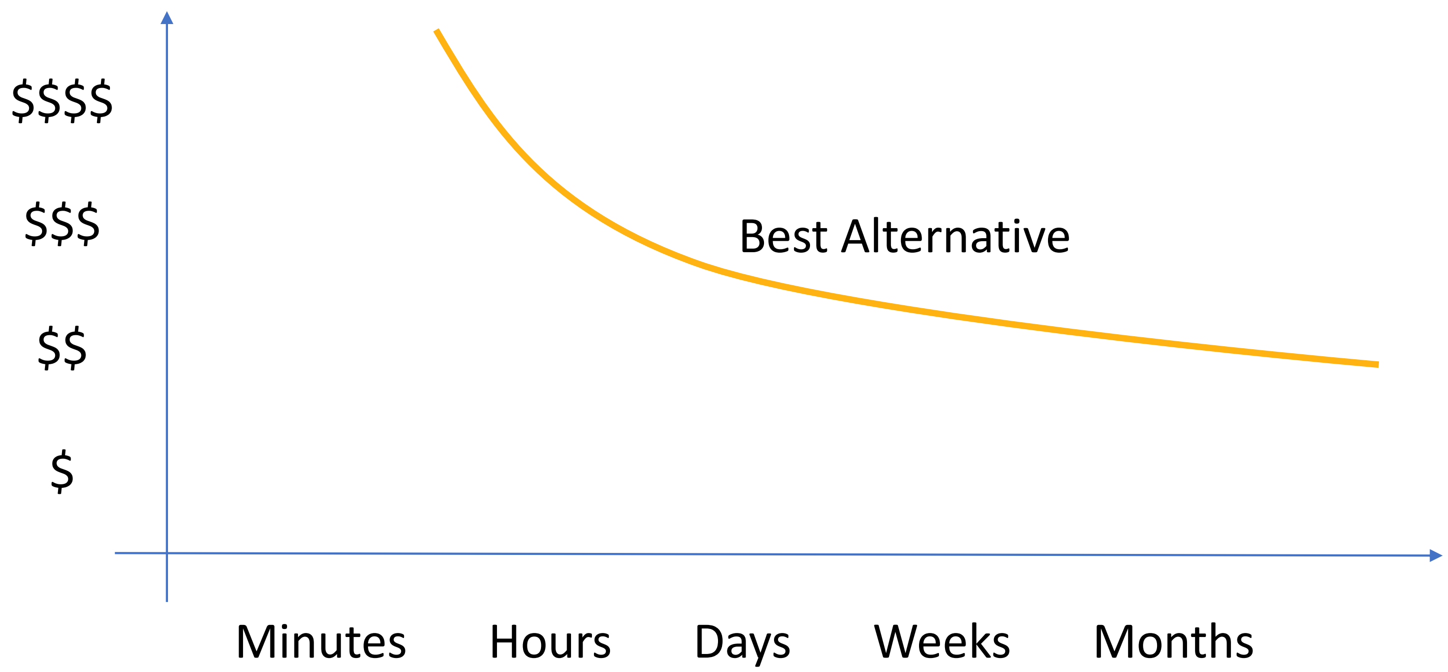
Distributed



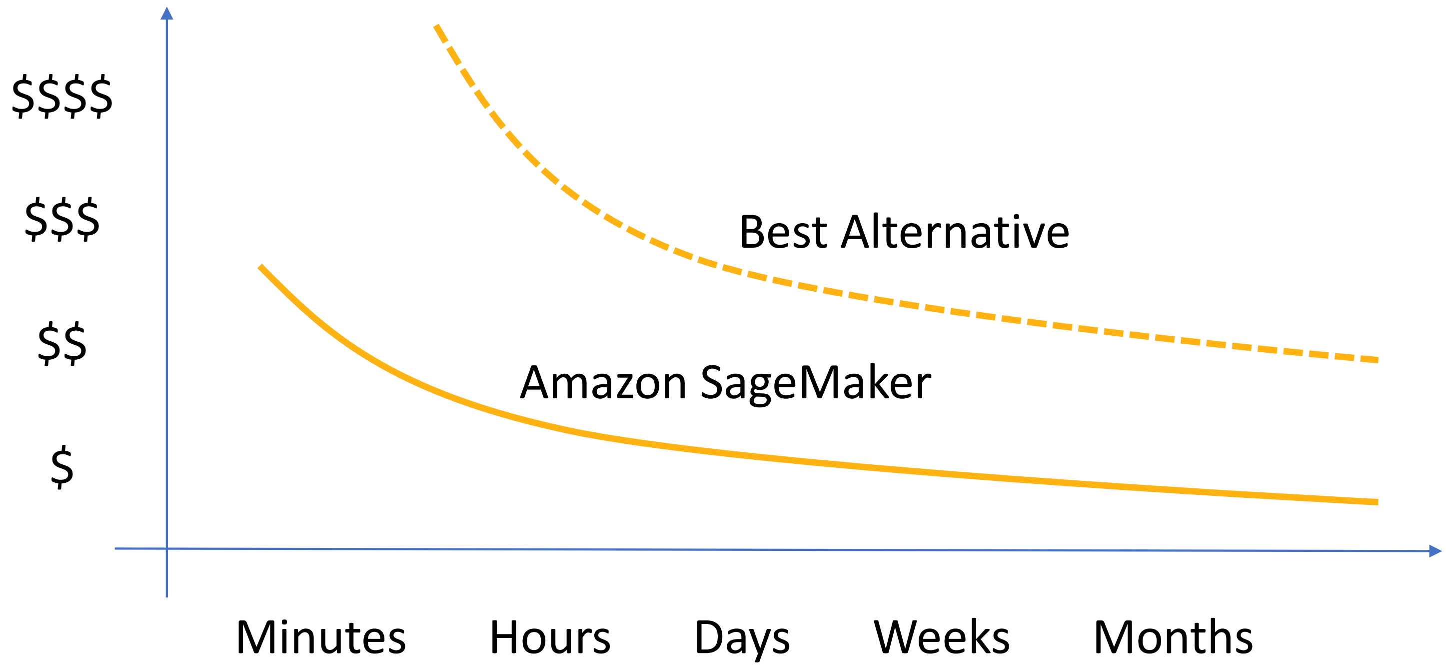
Parameter Server – distributed (k,v) store.



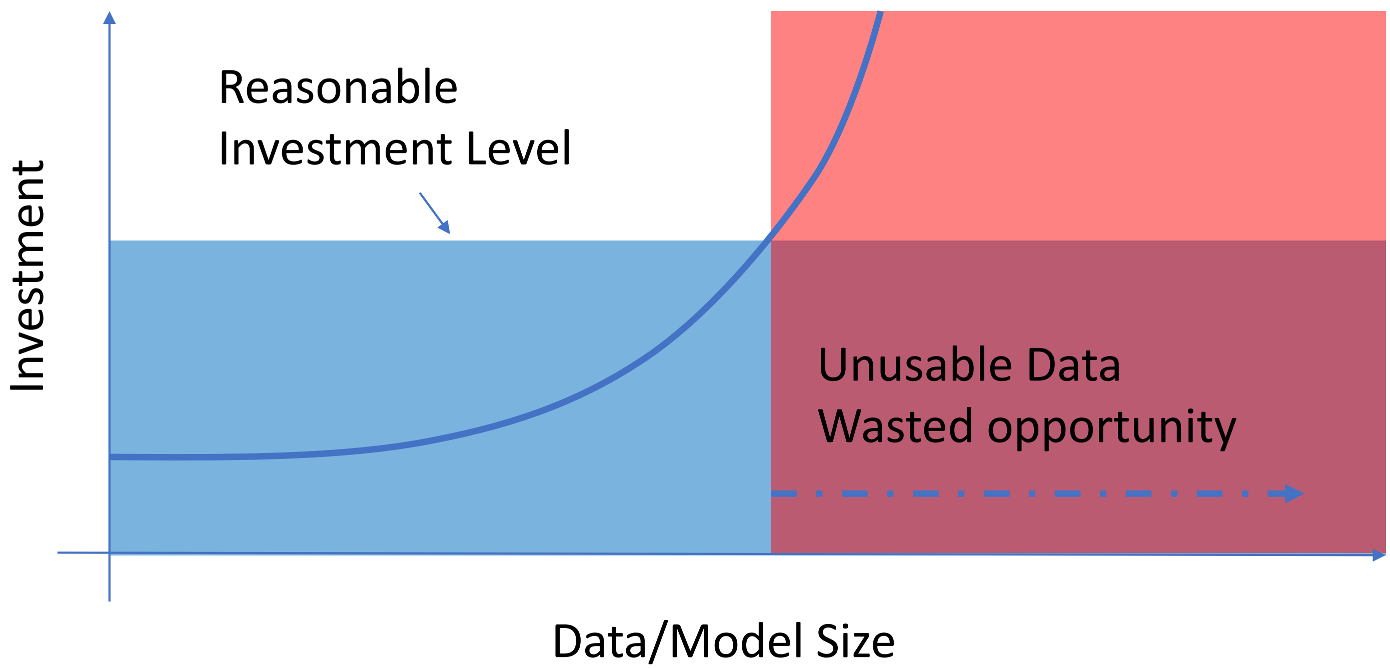
Cost vs. Time



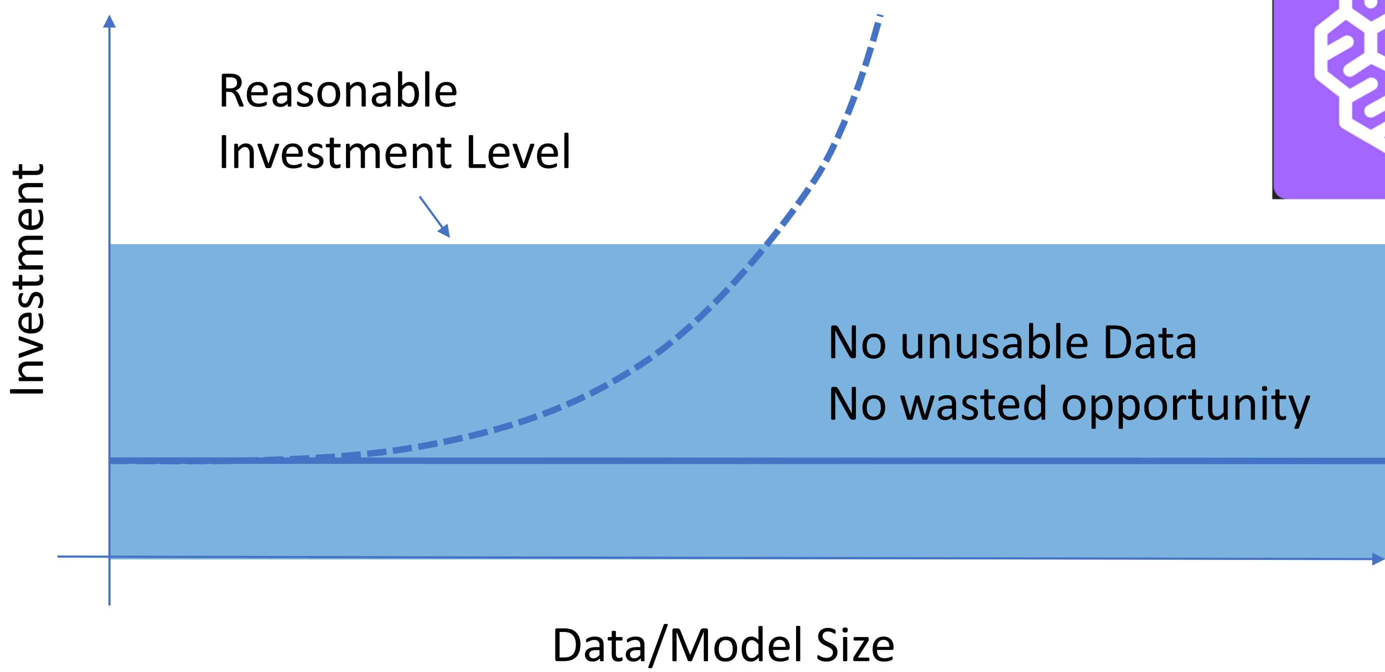
Cost vs. Time


$$$$
$$$
$$
$
Best Alternative
Amazon SageMaker
Minutes
Hours
Days
Weeks
Months

Production Readiness

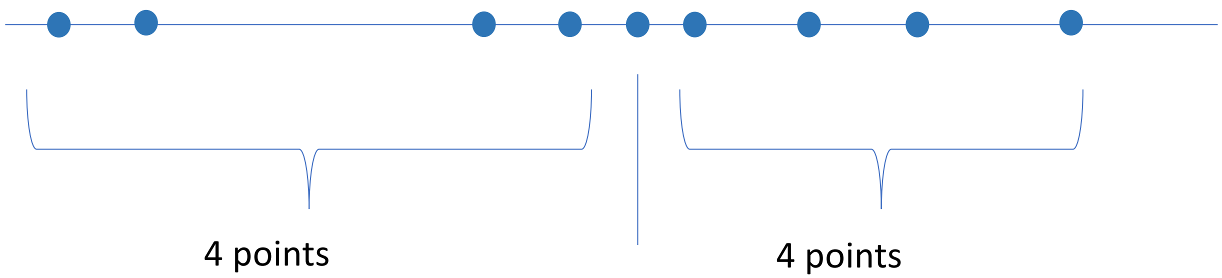


Production Readiness



Science of Streaming Algorithms – Advantages and Challenges

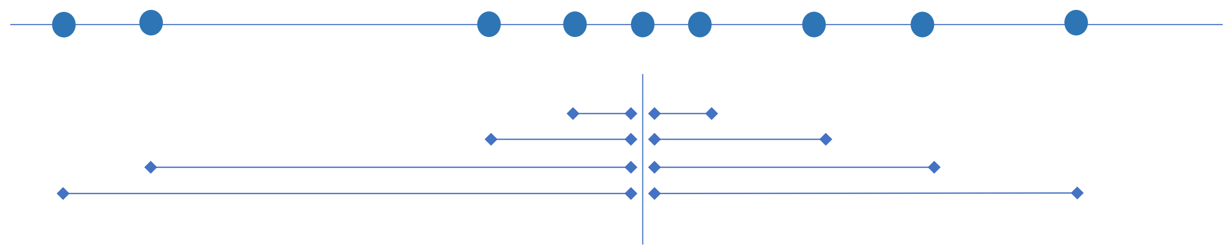
Simple Problems Are Unsolvable



Finding the exact median in a stream is impossible!

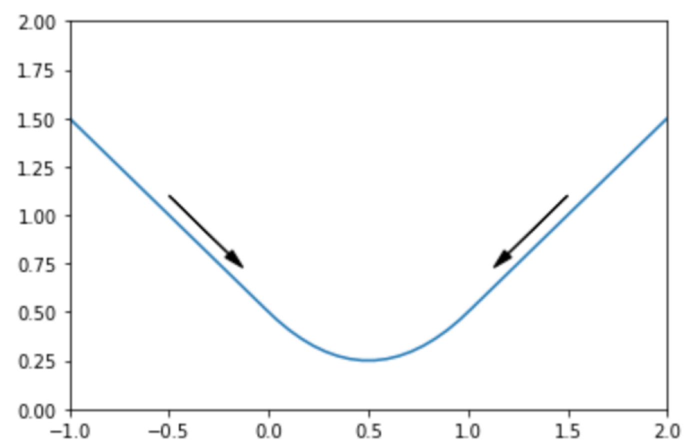
- After the seeing half the items, each one of them might still be the median.
- The algorithm must remember all of them.
- It cannot have a fixed memory footprint.

Gradient Descent



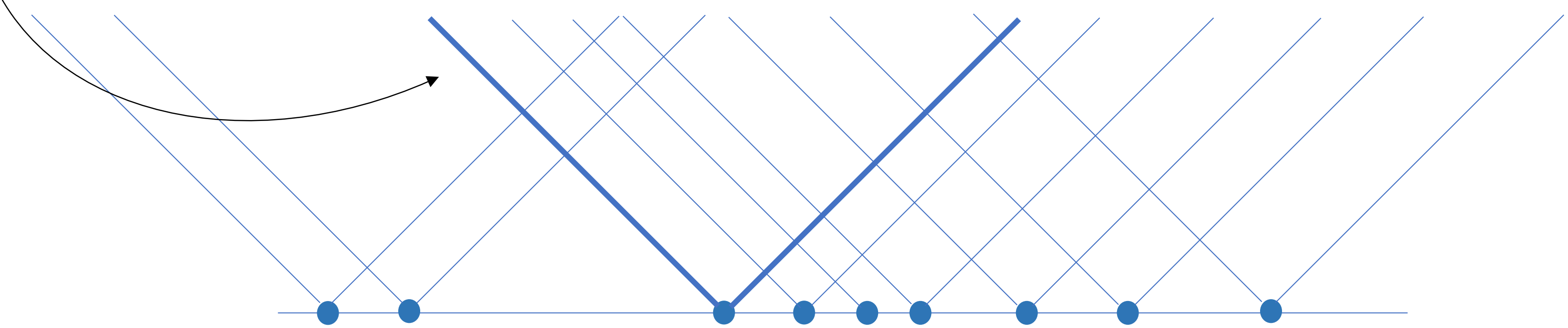
$$f(x) = \frac{1}{n}\sum_{i=1}^{n} |x - x_i|$$

$$m = \arg\min_x f(x)$$



Stochastic Gradient Descent

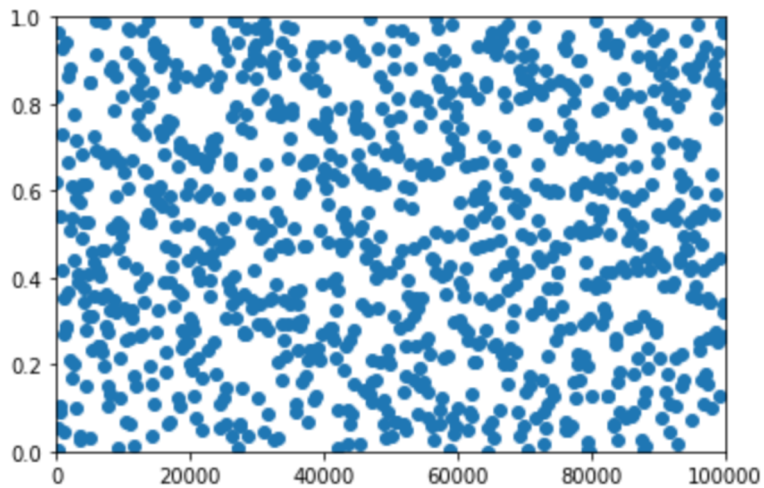
$$f_i = |x_i - x| \longrightarrow \mathbb{E}_i[f_i] = f \longrightarrow \mathbb{E}_i[f_i'] = f'$$

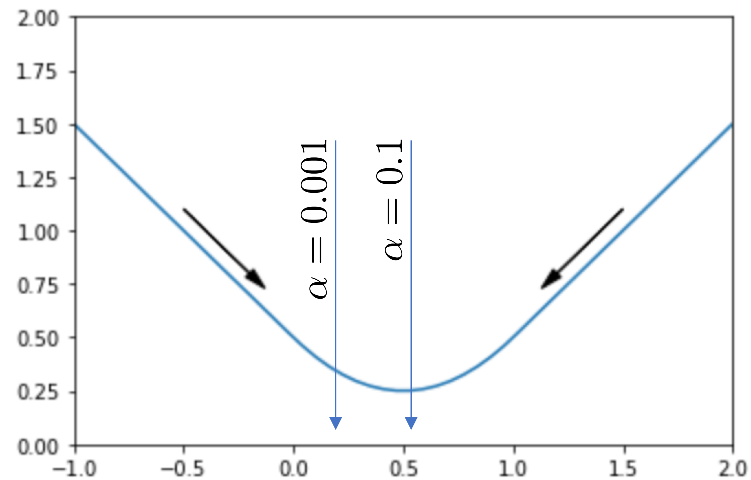


$$m_t = \begin{cases} m_{t-1} + \alpha/\sqrt{t} & \text{if } m_{t-1} < x_i \\ m_{t-1} - \alpha/\sqrt{t} & \text{if } m_{t-1} > x_i \end{cases}$$

Frugal Streaming for Estimating Quantiles: One (or two) memory suffices: Qiang Ma, S. Muthukrishnan, Mark Sandler

SGD – Parameter Tuning

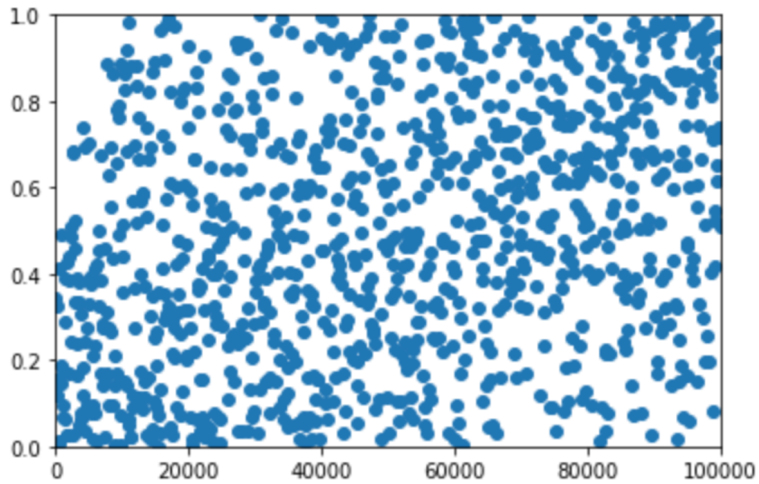


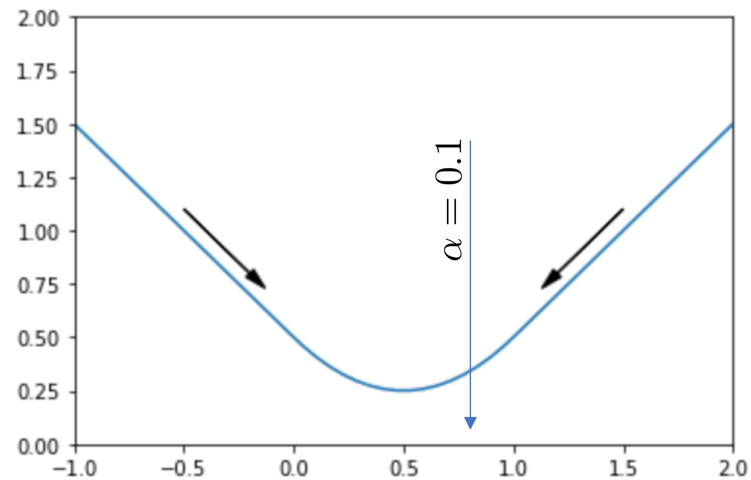


$$m_t = \begin{cases} m_{t-1} + \alpha/\sqrt{t} & \text{if } m_{t-1} < x_i \\ m_{t-1} - \alpha/\sqrt{t} & \text{if } m_{t-1} > x_i \end{cases}$$

Frugal Streaming for Estimating Quantiles: One (or two) memory suffices: Qiang Ma, S. Muthukrishnan, Mark Sandler

SGD – Distribution Drift

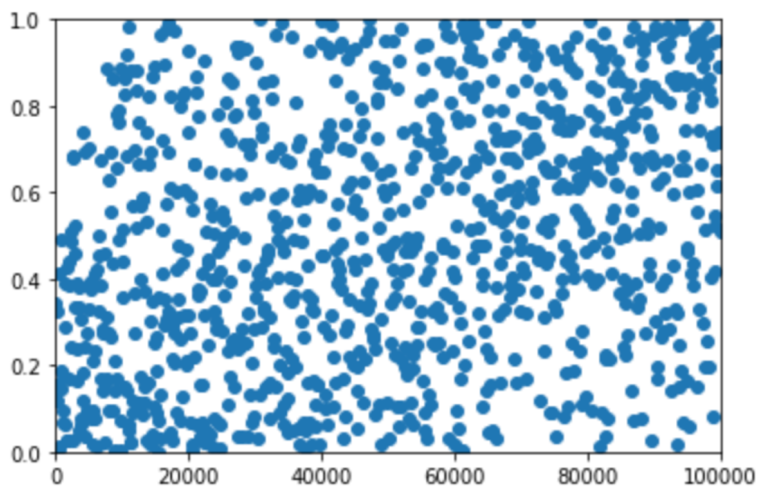


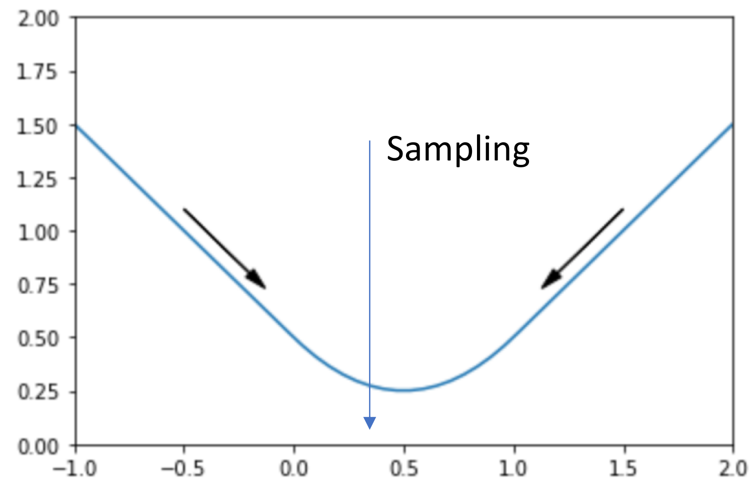


$$m_t = \begin{cases} m_{t-1} + \alpha/\sqrt{t} & \text{if } m_{t-1} < x_i \\ m_{t-1} - \alpha/\sqrt{t} & \text{if } m_{t-1} > x_i \end{cases}$$

Frugal Streaming for Estimating Quantiles: One (or two) memory suffices: Qiang Ma, S. Muthukrishnan, Mark Sandler

Median - Sampling Algorithm

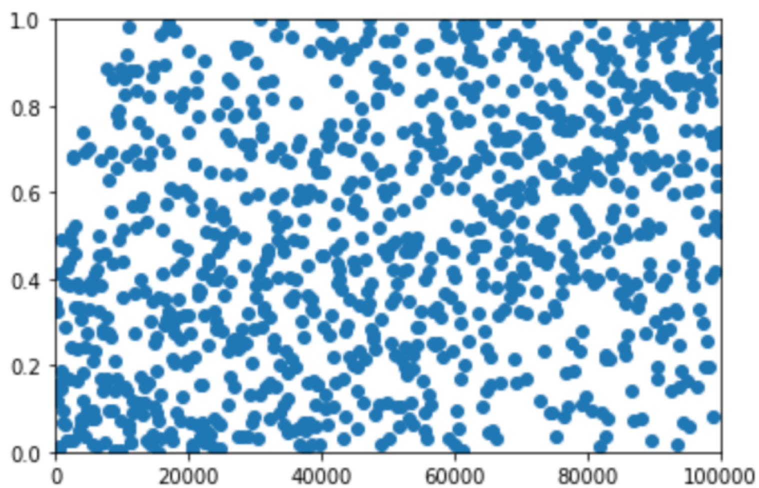


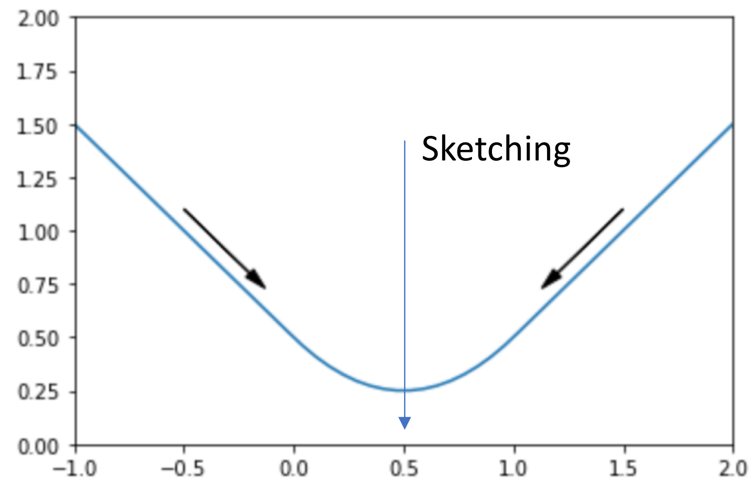


Sampling Algorithm:

1) Reservoir Sample k points from the data
2) Return the median of the sample

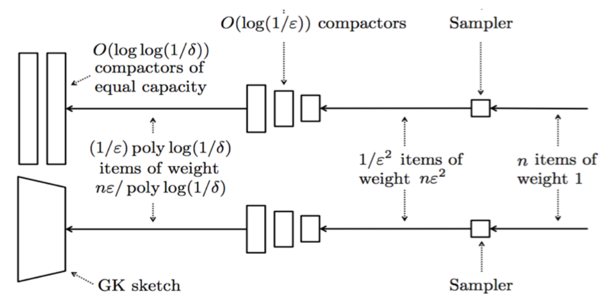
Median – Sketching Algorithm





Sketching Algorithm

1) Too complex to explain here…
2) Optimal Quantile Approximation in Streams; Zohar Karnin, Kevin Lang, Edo Liberty



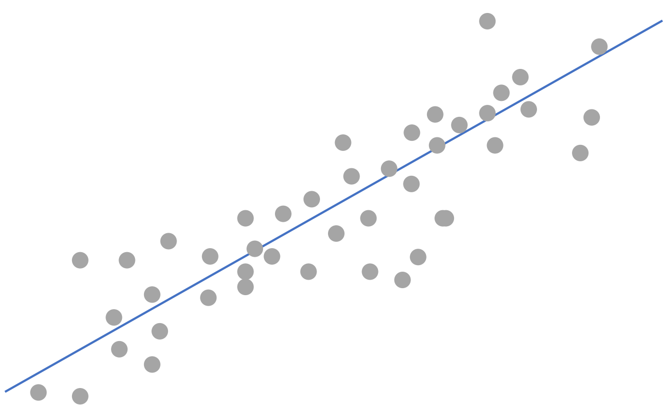
SageMaker Algorithms – Accurate, Fast, Scalable, and Easy to Use.

Algorithms- Example Usage

Algorithm	Function	Example Usage
Linear Learner	Classification and regression, these are the most popular ML algorithms used today.	• Estimating click probability for online advisements (for a customer) • Directing a customer's inbound phone call to relevant agents • Deciding whether a login event is legitimate.
Boosted Decision Trees (XGBoost)		
Factorization Machines		
K-means	Clustering	• Grouping similar events/document/images together
PCA	Principal Component Analysis	• Reduce Dimensionality of data • Explore main factors/trends in data • Visualization
Neural Topic Modelling	Topic Modeling	• Maps documents into distribution over topics • Discover dominant topics in your text corpus
Spectral LDA		
Blazing Text	Word Embedding	• Feature Engineering for text
DeepAR	Time-series Forecasting	• Predict the number of page views you'll get in an hour (and the number of servers you'll need to host them!)
Image Classification	Classification of Images	• Detect quality assurance issues in manufactured goods using images.
Sequence to Sequence	Learn mapping between pairs of sequences	• Translating text between different languages.

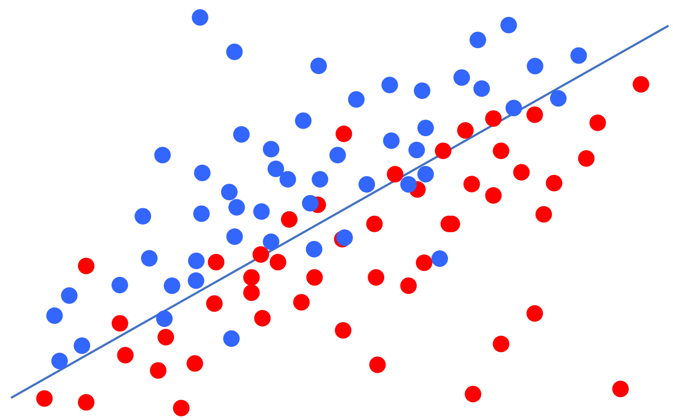
Linear Learner

Regression:
Estimate a real valued function



$$\hat{y} = \langle x, w \rangle + t$$

Binary Classification:
Predict a 0/1 class



$$\hat{y} = \begin{cases} 1 & \text{if } x\langle x, w \rangle \geq t \\ 0 & \text{otherwise} \end{cases}$$

Linear Learner

>8x speedup over naïve parallel training!

Train

↓

Fit thresholds and select

$$w_1 = \min_w \sum_i L_1(w^T x_i, y_i) + \alpha_1 \|w\|_1 + \beta_1 \|w\|_2$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$w_k = \min_w \sum_i L_k(w^T x_i, y_i) + \alpha_k \|w\|_1 + \beta_k \|w\|_2$$

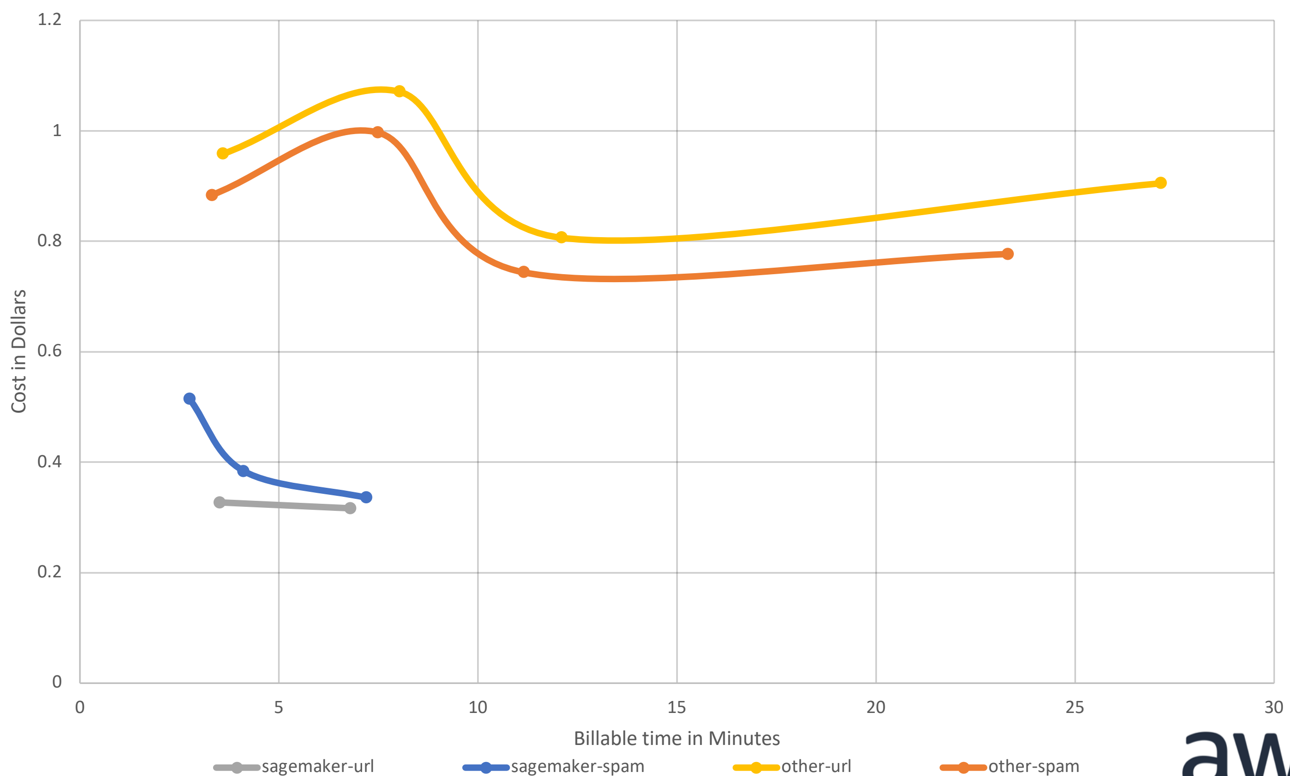
Select model with best validation performance

Linear Learner

Regression (mean squared error)	
SageMaker	Other
1.02	1.06
1.09	1.02
0.332	0.183
0.086	0.129
83.3	84.5

Classification (F1 Score)	
SageMaker	Other
0.980	0.981
0.870	0.930
0.997	0.997
0.978	0.964
0.914	0.859
0.470	0.472
0.903	0.908
0.508	0.508

30 GB datasets for web-spam and web-url classification

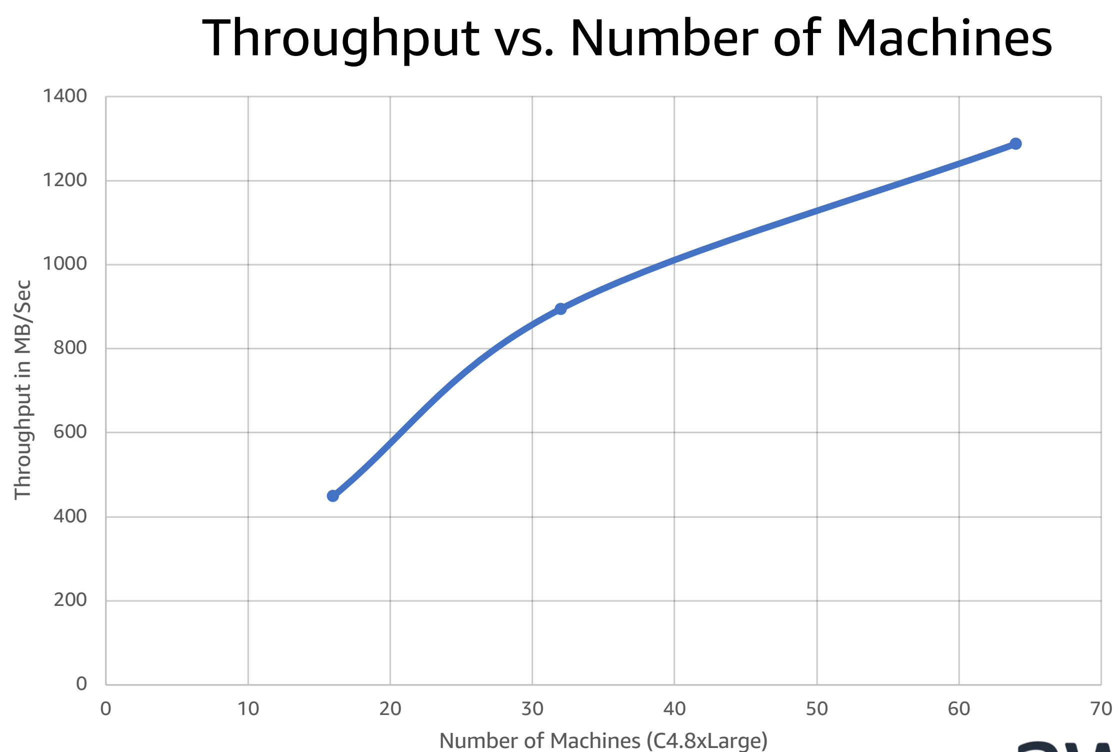


Boosted Decision Trees

XGBoost is one of the most commonly used implementations of boosted decision trees in the world.

It is now available in Amazon SageMaker!

dmlc
XGBoost

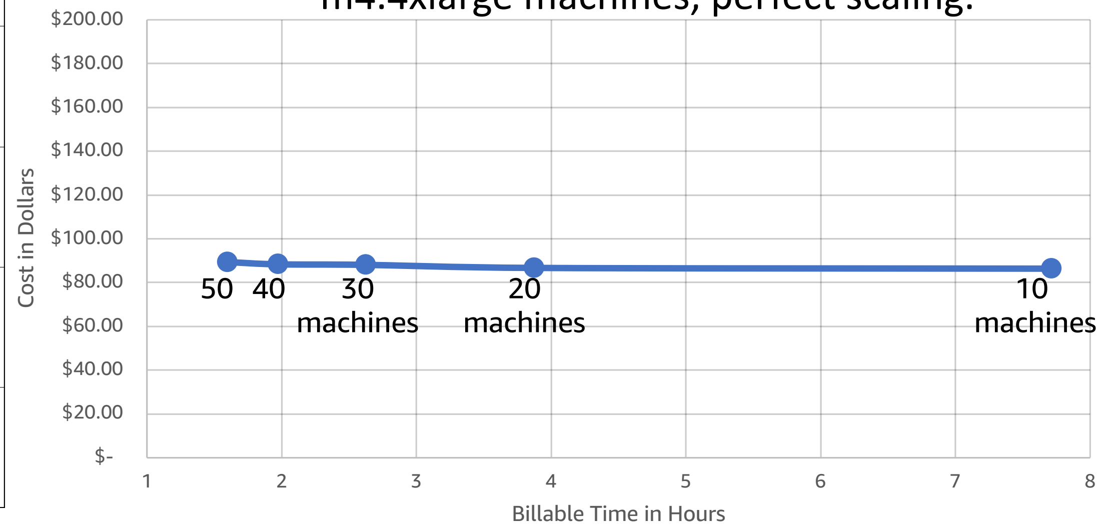


Factorization Machines

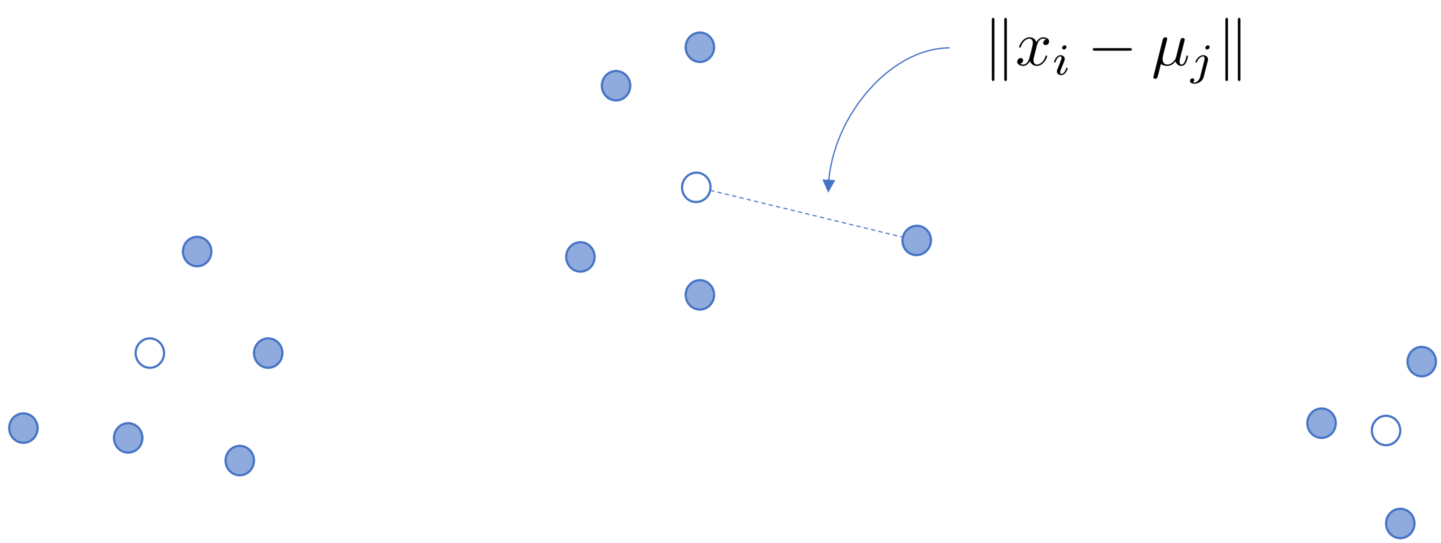
$$\hat{y} = w_0 + \langle w_1, x \rangle + \sum_{i,j>i} x_i x_j \langle v_i, v_j \rangle$$

	Log_loss	F1 Score	**Seconds**
SageMaker	0.494	0.277	820
Other (10 Iter)	0.516	0.190	650
Other (20 Iter)	0.507	0.254	1300
Other (50 Iter)	0.481	0.313	3250

Click Prediction 1 TB advertising dataset, m4.4xlarge machines, perfect scaling.



K-Means Clustering



$$\frac{1}{n}\sum_i \min_j \|x_i - \mu_j\|^2$$

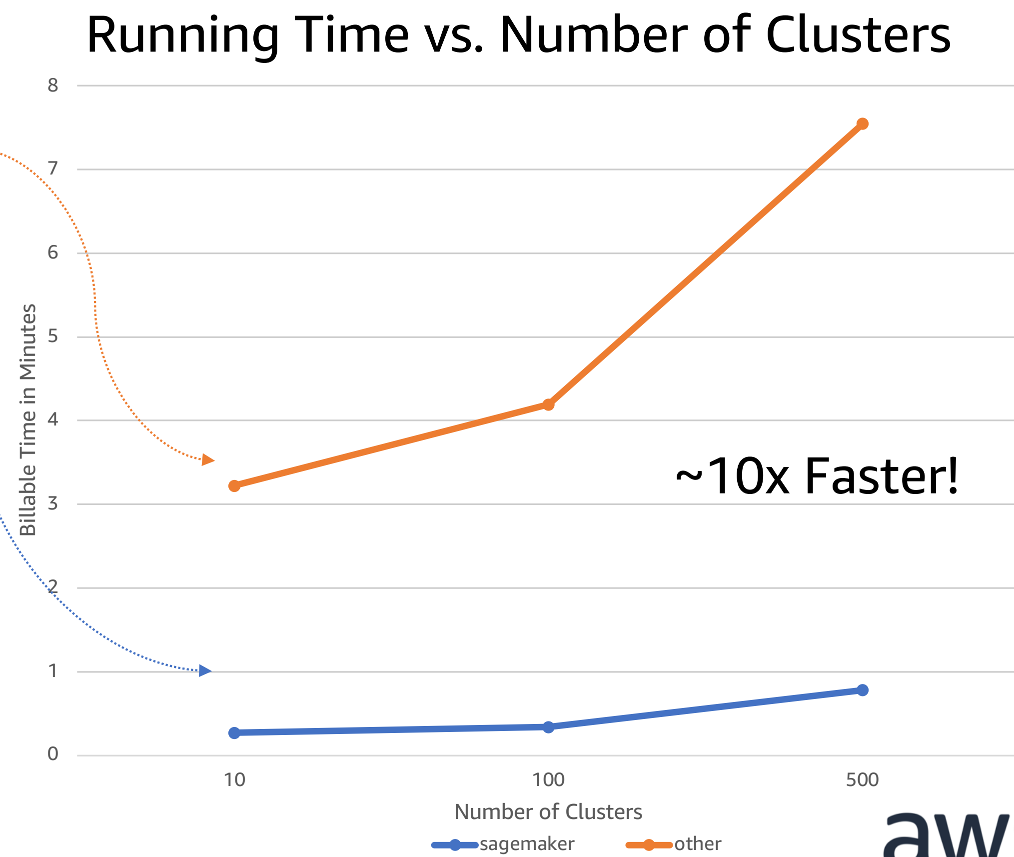
K-Means Clustering

Method	Accurate?	Passes	Efficient Tuning	Comments
Lloyds [1]	Yes*	5-10	No	
K-Means ++ [2]	Yes	k+5 to k+10	No	scikit-learn
K-Means|| [3]	Yes	7-12	No	spark.ml
Online [4]	No	1	No	
Streaming [5,6]	No	1	No	Impractical
Webscale [7]	No	1	No	spark streaming
Coresets [8]	No	1	Yes	Impractical
SageMaker	**Yes**	**1**	**Yes**	

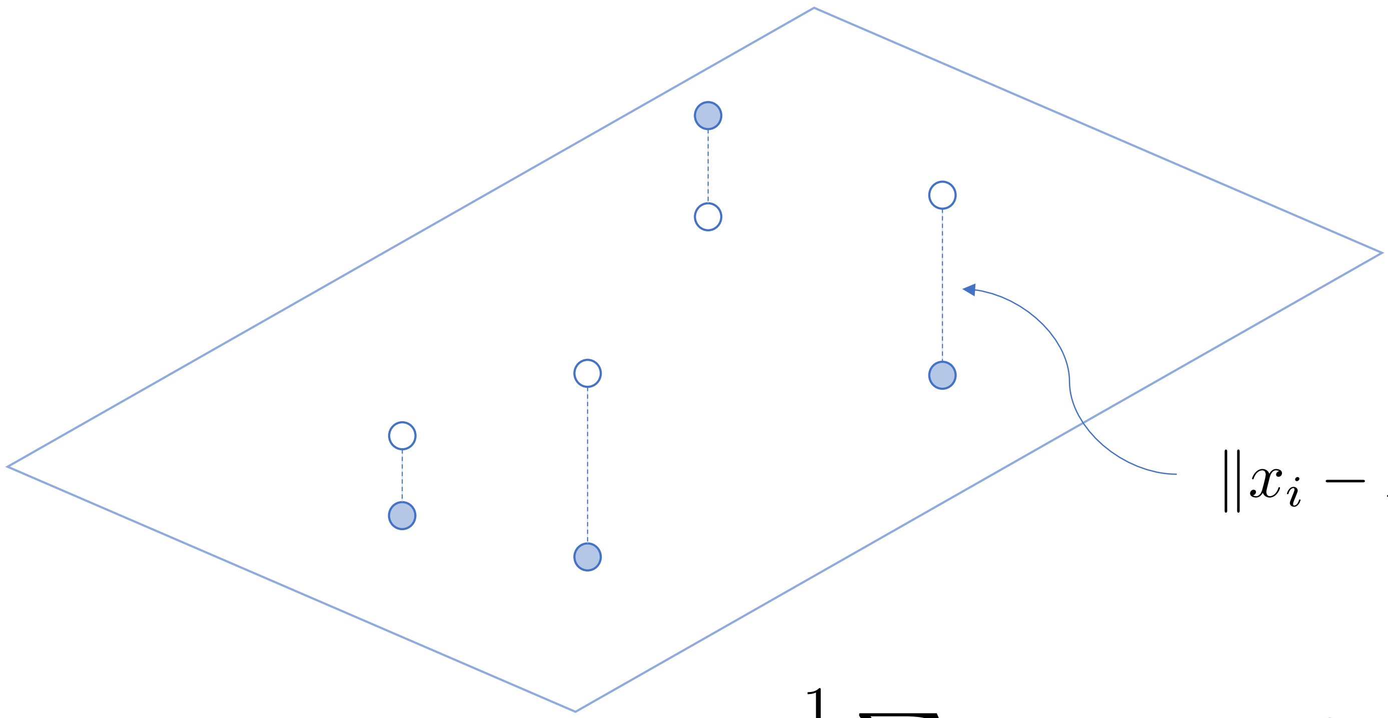
[1] Lloyd, IEEE TIT, 1982
[2] Arthur et. al. ACM-SIAM, 2007
[3] Bahmani et. al., VLDB, 2012
[4] Liberty et. al., 2015
[5] Shindler et. al, NIPS, 2011
[6] Guha et. al, IEEE Trans. Knowl. Data Eng. 2003
[7] Sculley, WWW, 2010
[8] Feldman et. al.

K-Means Clustering

	k	SageMaker	Other
Text 1.2GB	10	1.18E3	1.18E3
	100	1.00E3	9.77E2
	500	9.18.E2	9.03E2
Images 9GB	10	3.29E2	3.28E2
	100	2.72E2	2.71E2
	500	2.17E2	Failed
Videos 27GB	10	2.19E2	2.18E2
	100	2.03E2	2.02E2
	500	1.86E2	1.85E2
Advertising 127GB	10	1.72E7	Failed
	100	1.30E7	Failed
	500	1.03E7	Failed
Synthetic 1100GB	10	3.81E7	Failed
	100	3.51E7	Failed
	500	2.81E7	Failed



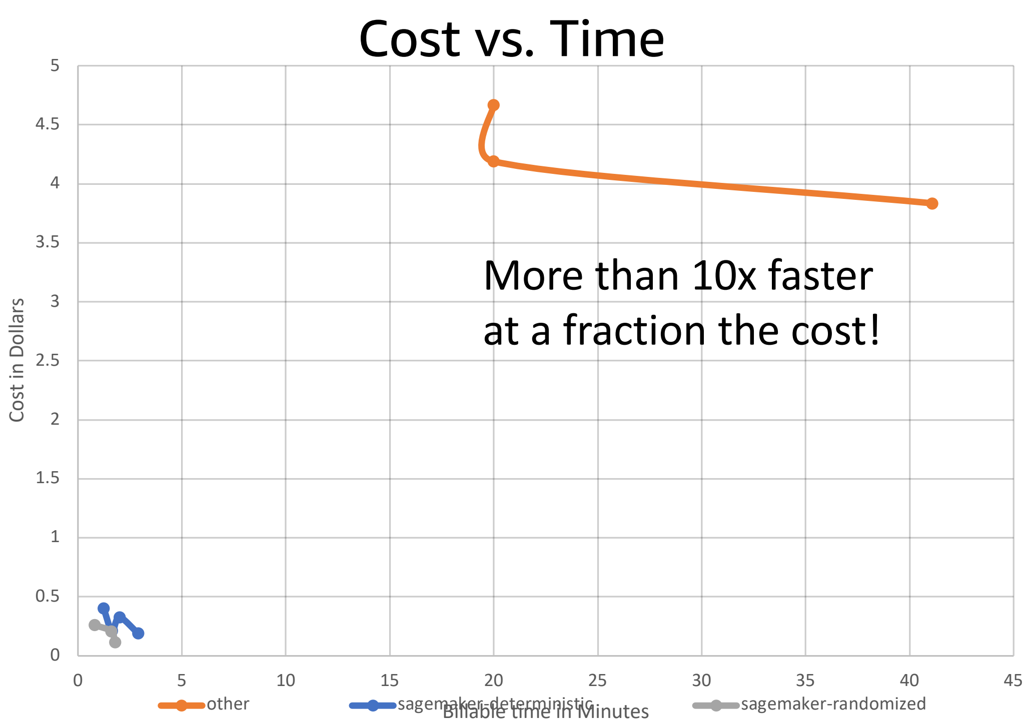
Principal Component Analysis (PCA)

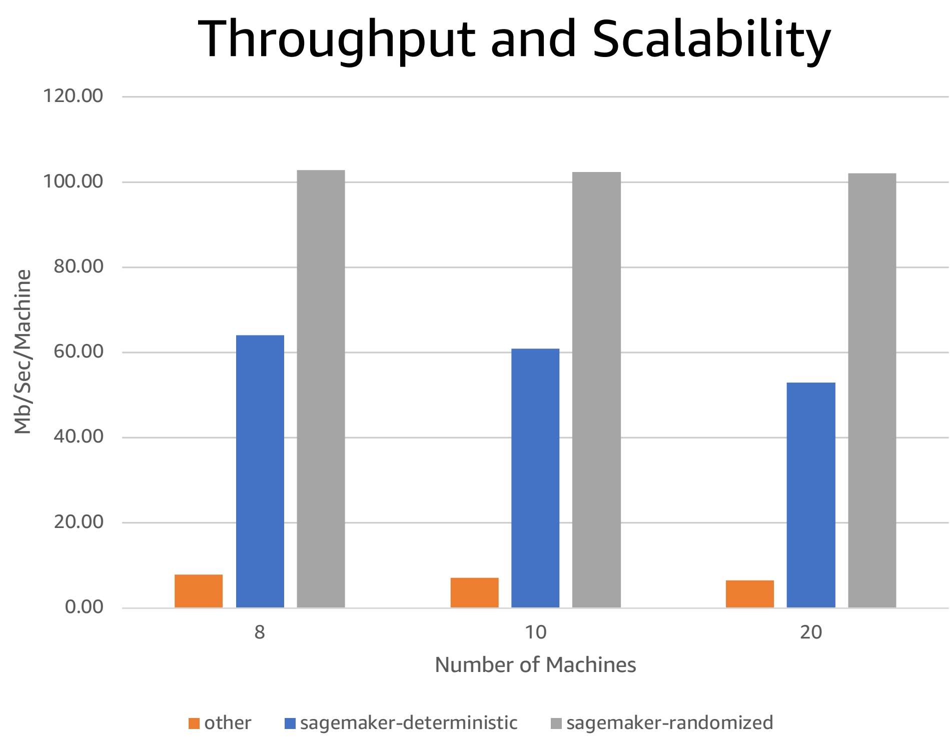


$$\|x_i - P(x_i)\|$$

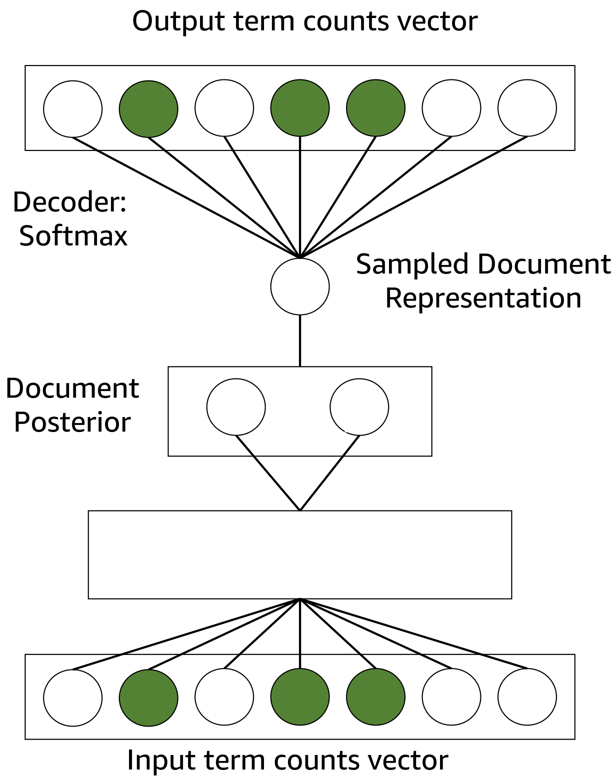
$$\frac{1}{n}\sum_i \|x_i - P(x_i)\|^2$$

Principal Component Analysis (PCA)

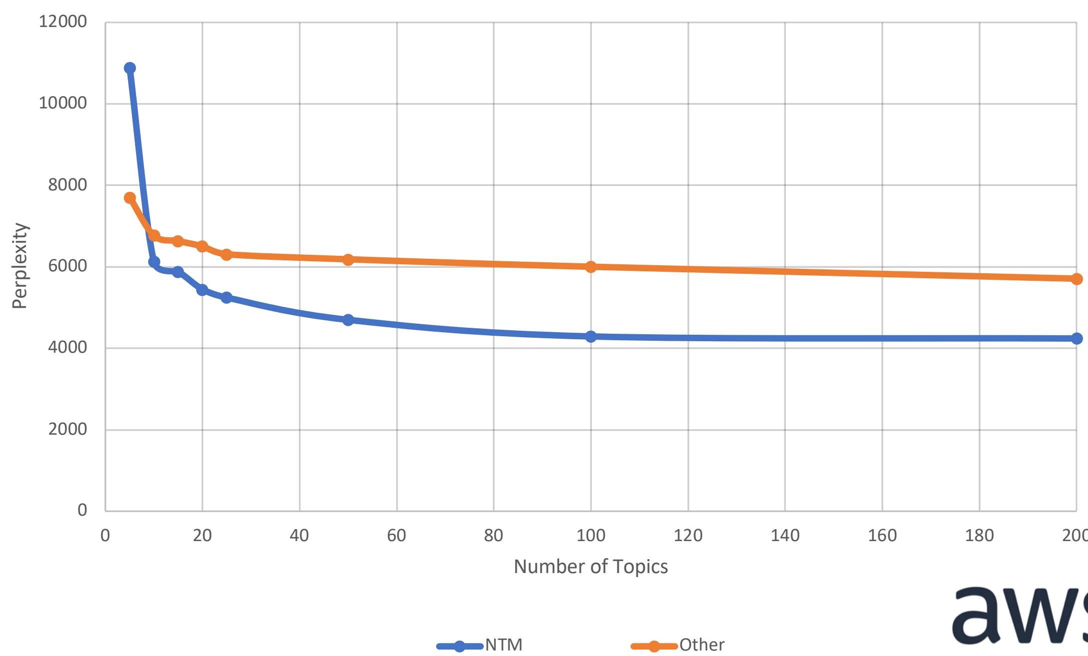




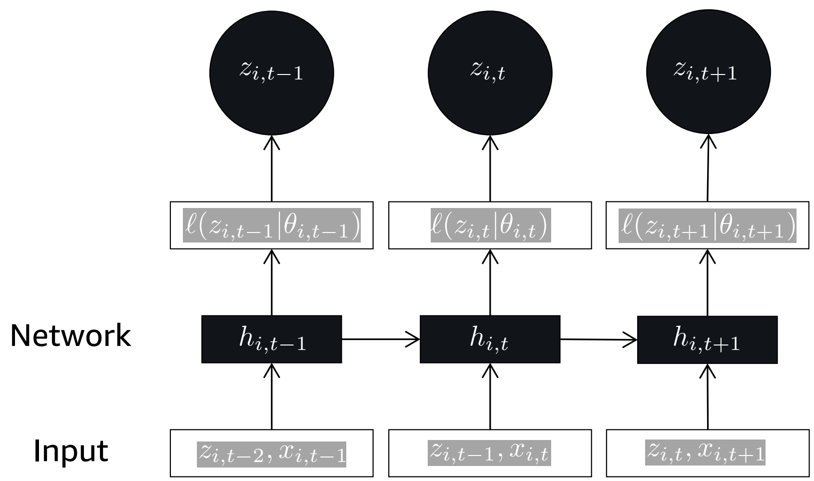
Neural Topic Modeling



- Perplexity vs. Number of Topic
- (~200K documents, ~100K vocabulary)



Time Series Forecasting



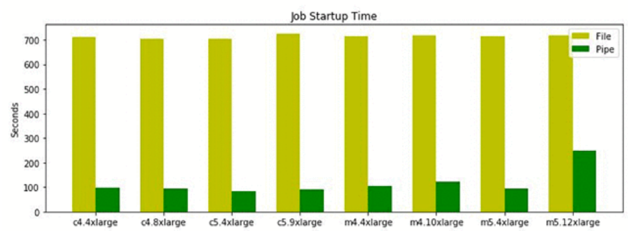
		Mean absolute percentage error		P90 Loss	
		DeepAR	R	DeepAR	R
traffic Hourly occupancy rate of 963 bay area freeways		**0.14**	0.27	**0.13**	0.24
electricity Electricity use of 370 homes over time		**0.07**	0.11	**0.08**	0.09
pageviews Page view hits of websites	10k	**0.32**	**0.32**	0.44	**0.31**
	180k	**0.32**	0.34	**0.29**	NA

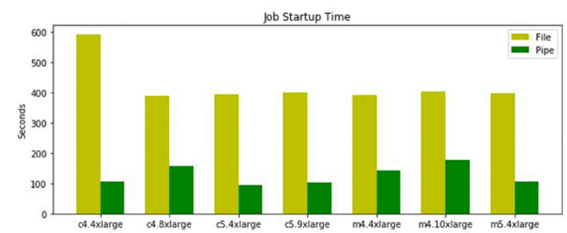
One hour on p2.xlarge, $1

Pipe Mode (launched May 23rd)

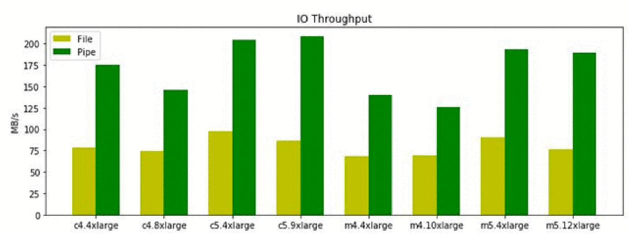
Job Execution Time

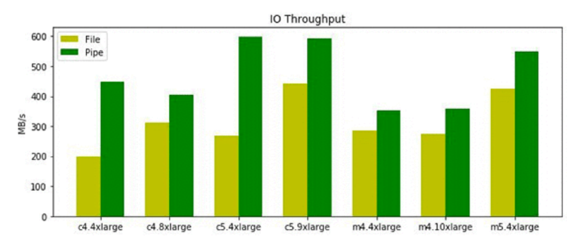
Job Startup Time





Throughput





PCA

K-Means

Using Amazon SageMaker Algorithms on AWS

From Amazon SageMaker Notebooks

Hardware

Parameters

Start Training

```
import boto3
import sagemaker

sess = sagemaker.Session()

pca = sagemaker.estimator.Estimator(containers[boto3.Session().region_name],
                                    role,
                                    train_instance_count=1,
                                    train_instance_type='ml.c4.xlarge',
                                    output_path=output_location,
                                    sagemaker_session=sess)
pca.set_hyperparamters(feature_dim=50000,
                       num_components=10,
                       subtract_mean=True,
                       algorithm_mode='randomized',
                       mini_batch_size=200)

pca.fit({'train': s3_train_data})
```

Host model

```
pca_predictor = pca.deploy(initial_instance_count=1,
                           instance_type='ml.c4.xlarge')
```

From Command Line

Algorithm

Input Data

Hardware

```
profile=<your_profile>

arn_role=<your_arn_role>

training_image=382416733822.dkr.ecr.us-east-1.amazonaws.com/kmeans:1

training_job_name=clutering_text_documents_`date '+%Y_%m_%d_%H_%M_%S'`

aws --profile $profile \

    --region us-east-1 \

    sagemaker create-training-job \

    --training-job-name $training_job_name \

    --algorithm-specification TrainingImage=$training_image,TrainingInputMode=File \

    --hyper-parameters k=10,feature_dim=1024,mini_batch_size=1000 \

    --role-arn $arn_role \

    --input-data-config '{"ChannelName": "train", "DataSource": {"S3DataSource":{"S3DataType":         "S3Prefix", "S3Uri":
"s3://kmeans_demo/train", "S3DataDistributionType": "ShardedByS3Key"}},   "CompressionType": "None", "RecordWrapperType": "None"}' \

    --output-data-config S3OutputPath=s3://training_output/$training_job_name

    --resource-config InstanceCount=2,InstanceType=ml.c4.8xlarge,VolumeSizeInGB=50 \

    --stopping-condition MaxRuntimeInSeconds=3600
```

SageMaker + Spark =

```python
# Python/PySpark Example
from sagemaker_pyspark import SageMakerEstimator

features = spark.read.parquet('s3://<bucket>/<dataset>’)

algorithm = SageMakerEstimator(
    trainingImage=ntm_container,
    modelImage=ntm_container,
    trainingInstanceType='ml.p3.8xlarge',
    trainingInstanceCount=16,
    endpointInstanceType='ml.c5.2xlarge',
    endpointInitialInstanceCount=4,
    hyperParameters={
        "num_topics": "100",
        "feature_dim": 250000",
        "mini_batch_size": "10000",
    },
    sagemakerRole=IAMRole(role_arn)
)

model = algorithm.fit(features)
```

SageMaker + Spark =

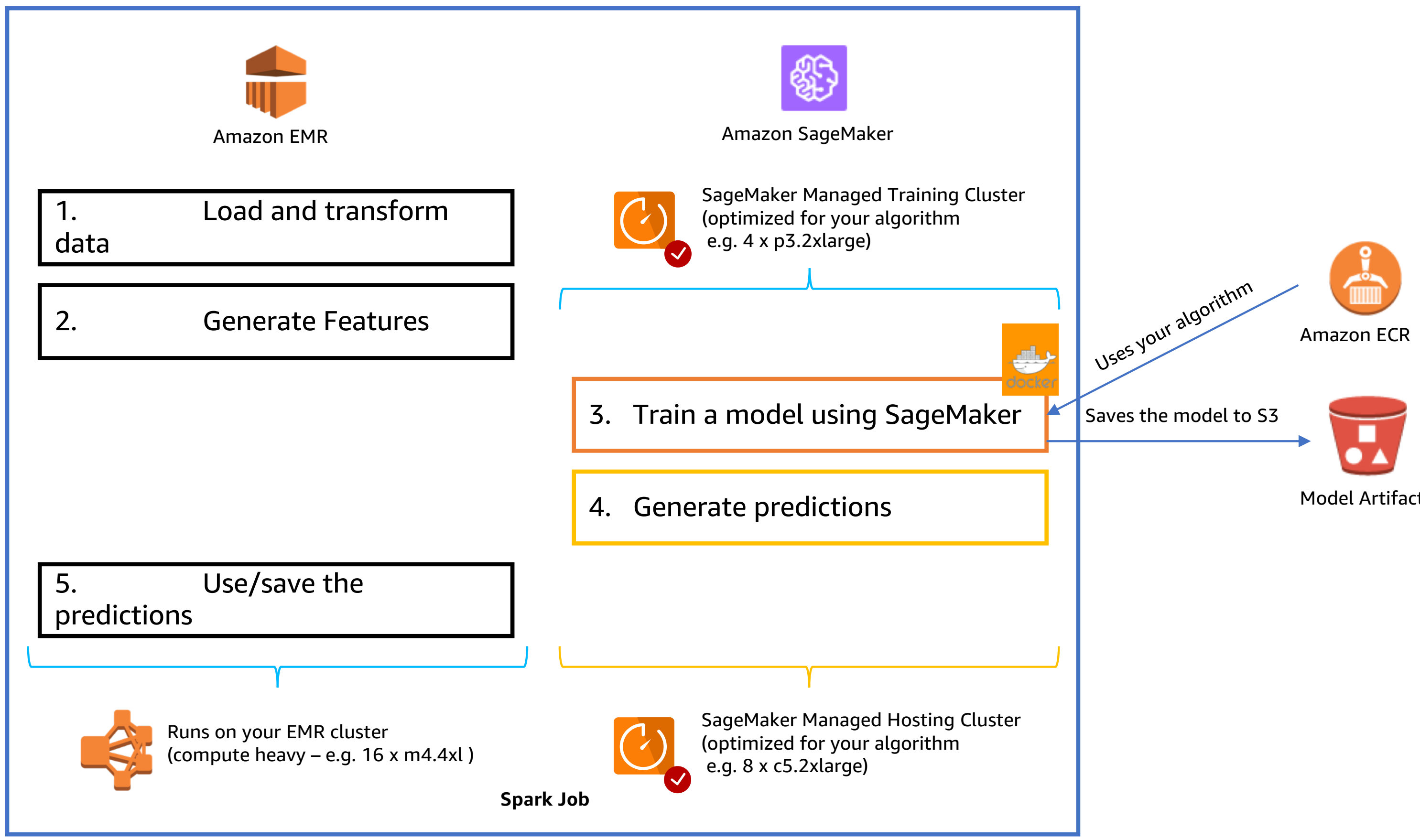
```scala
// Scala Example
import com.amazonaws.services.sagemaker.sparksdk.{IAMRole, SageMakerEstimator}

val features = spark.read.parquet("s3://<bucket>/<dataset>")

val algorithm = new SageMakerEstimator(
  trainingImage = ntm_container,
  modelImage = ntm_container,
  trainingInstanceType = "ml.p3.8xlarge",
  trainingInstanceCount = 16,
  endpointInstanceType = "ml.c5.2xlarge",
  endpointInitialInstanceCount = 4,
  hyperParameters = Map(
    "num_topics"  -> "100",
    "feature_dim" -> "250000",
    "mini_batch_size" -> "10000"
  ),
  sagemakerRole = IAMRole(roleArn)
)

val model = estimator.fit(features)
```

SageMaker + Spark =



Amazon SageMaker - Try It Out