



Using Apache Spark with Amazon SageMaker

Julien Simon
Principal Technical Evangelist, AI and Machine Learning
[@julsimon](#)

May 2018

Agenda

- Apache Spark on AWS
- Amazon SageMaker
- Combining Spark and SageMaker
- Demos with the SageMaker SDK for Spark
- Getting started

Services covered: Amazon EMR, Amazon SageMaker

Apache Spark on AWS

Apache Spark

<https://spark.apache.org/>



- Open-source, **distributed processing** system.
- In-memory caching and optimized execution for **fast performance** (typically 100x faster than Hadoop).
- Batch processing, streaming analytics, machine learning, graph databases and ad hoc queries.
- API for Java, Scala, Python, R, and SQL.

Apache Spark – *DataFrame*



- Distributed collection of data organized into **named columns**.
- Conceptually equivalent to a table in a relational database.
- Wide array of **sources**: structured files, databases.
- Wide array of **formats**: text, CSV, JSON, Avro, ORC, Parquet.

```
{"name": "Michael"}  
{"name": "Andy", "age": 30}  
{"name": "Justin", "age": 19}
```

```
df = spark.read.json("people.json")  
df.show()  
+----+-----+  
| age| name |  
+----+-----+  
|null|Michael|  
| 30 | Andy |  
| 19 | Justin|  
+----+-----+
```

MLlib – Machine learning library

<https://spark.apache.org/docs/latest/ml-guide.html>



- ML **algorithms**: classification, regression, clustering, collaborative filtering.
- Featurization: feature extraction, transformation, dimensionality reduction.
- Tools for constructing, evaluating and tuning ML **pipelines**
- *Transformer* – a **transform function** that maps a *DataFrame* into a new one
 - Adding a column, changing the rows of a specific column, etc.
 - Predicting the label based on the feature vector.
- *Estimator* – an **algorithm** that trains on data
 - Consists of a *fit()* function that maps a *DataFrame* into a *Model*.

Apache Spark on Amazon EMR

<https://aws.amazon.com/emr/>



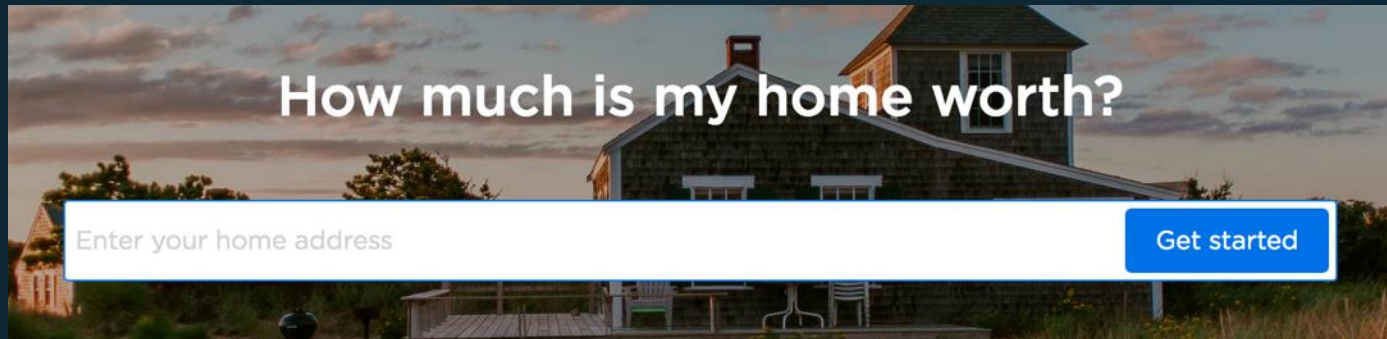
- Spark is natively supported in **Amazon EMR**.
- Amazon S3 connectivity using the **EMR File System (EMRFS)**.
- Amazon **Kinesis**, **Redshift** and **DynamoDB** as data sources.
- Integration with the AWS **Glue** Data Catalog.
- Auto Scaling to **add or remove instances** from your cluster.
- Integration with the Amazon EC2 **Spot** market.

Customer use case: Zillow

<https://aws.amazon.com/solutions/case-studies/zillow-zestimate/>



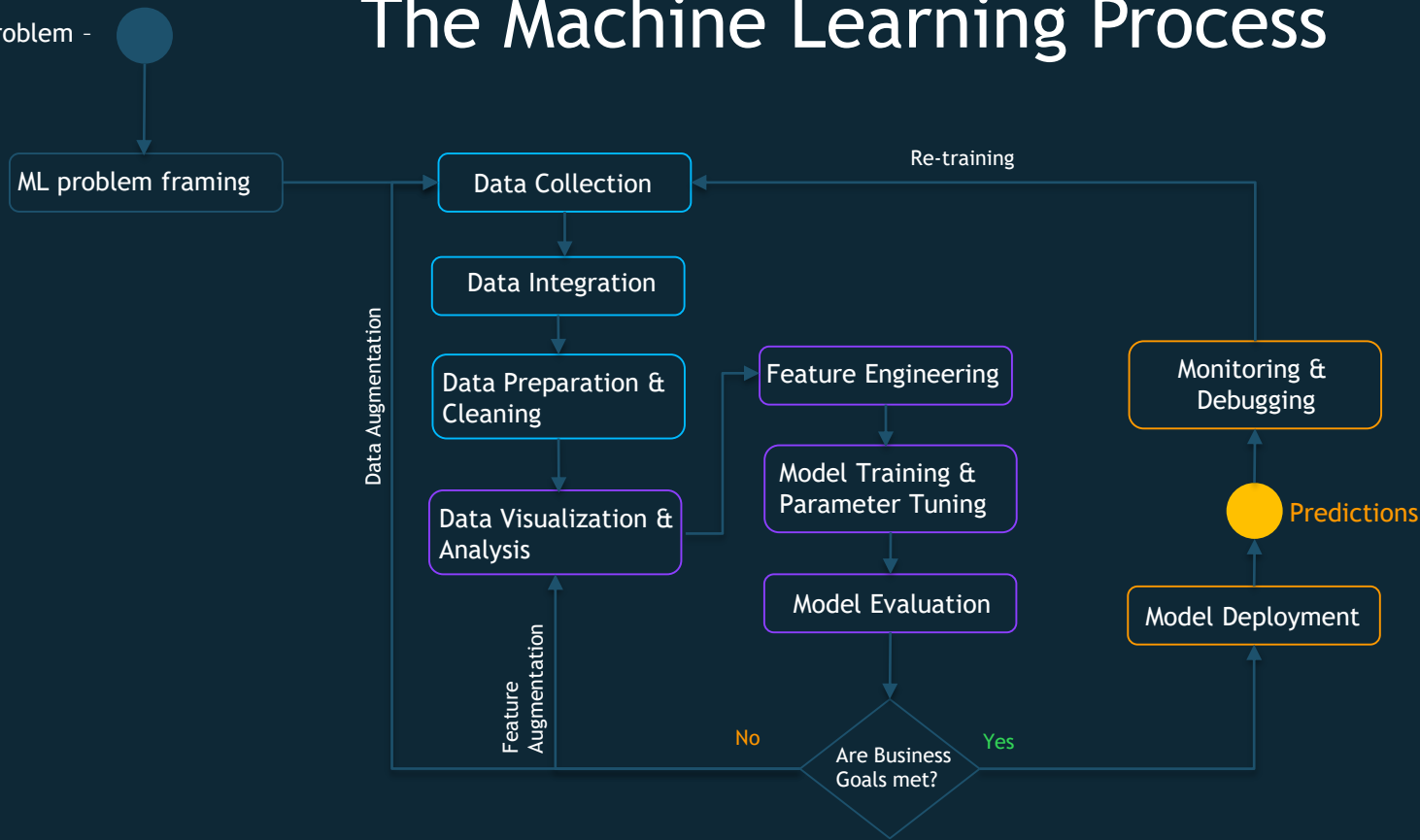
- Most popular **online real-estate website** in the US.
- Zestimate: a **home valuation** tool for over **100 million homes**.
- Computed from data ingested into **Kinesis** and pushed into **Spark** on **EMR**.
- Machine learning **models** give users **near real-time** Zestimates.



Amazon SageMaker

The Machine Learning Process

Business Problem -



Amazon SageMaker

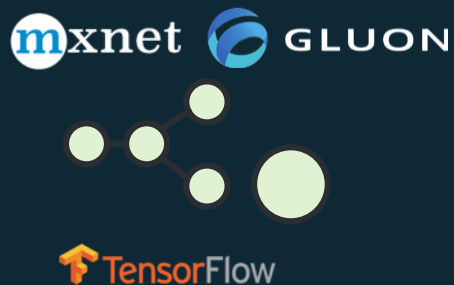
Build, train, and deploy models at scale



End-to-End
Machine Learning
Platform



Zero setup



Flexible Model
Training



Pay by the second

Amazon SageMaker

Deploy

Fully-managed hosting at scale

Deployment without engineering effort

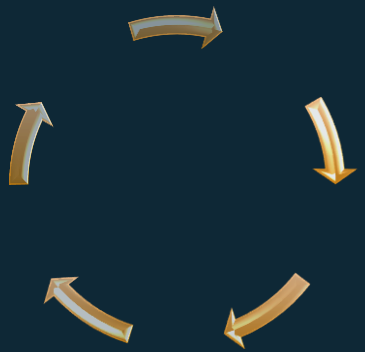
Pre-built notebook instances

Build

Highly-optimized machine learning algorithms

One-click training for ML, DL, and custom algorithms

Easier training with hyperparameter optimization



Customer use case: Digital Globe

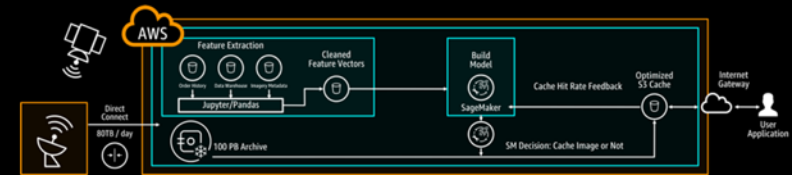
<https://aws.amazon.com/solutions/case-studies/digitalglobe-machine-learning/>



- Operating **Earth imaging** satellites and providing **image analysis** services.
- Over **100 PB** of imagery.
- Extensive use of Machine Learning on **SageMaker** to extract information from images.
- Working with the **AWS ML Lab**, built a predictive model **reducing cloud storage costs by 50%**.



USING AMAZON SAGEMAKER TO CUT CLOUD STORAGE COSTS IN HALF



Combining Spark and SageMaker

Decouple ETL and Machine Learning

- Different workloads require **different instance types**.
 - Say, M4 for ETL, P3 for ML training and C5 for ML prediction?
 - If you need GPUs for training, running your EMR cluster on GPU instances wouldn't be cost-efficient.
- Scale them **independently**.
 - Avoid oversizing your Spark cluster because one part of the process requires more capacity.
 - Avoid time-consuming resizing operations on EMR.
 - Run ETL once, train many models in parallel.
- SageMaker terminates training instances **automatically**.

Run any ML algorithm in any language

- Spark MLlib is great, but you may need something else.
- SageMaker algorithms, able to train on **huge data sets** without the need for huge clusters.
- Deep Learning libraries, like **TensorFlow** or **Apache MXNet**.
- Your own **custom code** in any language.

Deploy ML models in production

- Perform ML predictions **without** using Spark.
 - Save the overhead of the Spark framework.
 - Save loading your data in a *DataFrame*.
- Improve **latency** for small-batch predictions.
 - It can be difficult to achieve low-latency predictions with Spark ML models.
 - You can get real-time predictions with models hosted in SageMaker.
 - You can use very powerful instances for prediction endpoints.

Sample use cases

- Data preparation and feature engineering before training.
- Data transformation before batch prediction (model reuse).
- Data enrichment with predictions.
 - Predict missing values instead of using median.
 - Add new predicted features.
- Train on extremely large datasets with built-in algos.

SageMaker SDK for Spark

<https://github.com/aws/sagemaker-spark>

- Python and Scala SDK, for Apache Spark 2.1.1 and 2.2.
- Pre-installed on EMR 5.11 and later.
- Train, import, deploy and predict with SageMaker models **directly** from your Spark application.
 - Standalone,
 - Integration in Spark MLlib pipelines.
- *DataFrames* in, *DataFrames* out:
automatic conversion to and from protobuf

SageMaker SDK for Spark – built-in algorithms

<https://docs.aws.amazon.com/sagemaker/latest/dg/algos.html>

- High-level API for:

- Linear Learner
- Factorization Machines
- K-Means
- PCA
- LDA
- XGBoost

Infinitely scalable algorithms:
no limit to the amount of data that they can process

- The *SageMakerEstimator* object lets you use **any containerized code** stored in Amazon ECR (just like the regular SageMaker SDK).

Demos

<https://github.com/juliensimon/dlnotebooks>

- 1 – Classifying MNIST in Python with XGBoost (SageMaker)
- 2 – Clustering MNIST in Scala with K-Means (SageMaker)
- 3 – Clustering MNIST in Scala with a Pipeline: PCA (MLlib) + K-Means (SageMaker)

Getting started

<https://aws.amazon.com/machine-learning>

<https://aws.amazon.com/sagemaker>

<https://aws.amazon.com/emr>

[An overview of Amazon SageMaker](#)

[AWS re:Invent 2016: Best Practices for Apache Spark on Amazon EMR](#)

<https://github.com/aws/sagemaker-python-sdk>

<https://github.com/aws/sagemaker-spark>

<https://medium.com/@julsimon>

Thank you!

Julien Simon
Principal Technical Evangelist, AI and Machine Learning
@julsimon