Machine Learning on AWS with Amazon SageMaker

Constantin Gonzalez
Principal Solutions Architect, Amazon Web Services
glez@amazon.de

December 2017
Computer Programming (1936 – Today)

1.

2.
Machine Learning (1959 – Today)

1. 

2. 

aws
Welcome to Amazon.com Books!

One million titles, consistently low prices.

If you explore just one thing, make it our personal notification service. We think it's very cool!

Spotlight! -- August 16th
These are the books we love, offered at Amazon.com low prices. The spotlight moves EVERY day so please come often.

One Million Titles
Search Amazon.com's million title catalog by author, subject, title, keyword, and more... Or take a look at the books we recommend in over 20 categories... Check out our customer reviews and the award winners from the Hugo and Nebula to the Pulitzer and Nobel... and bestsellers are 30% off the publishers list.

Eyes & Editors, a personal notification service
Like to know when that book you want comes out in paperback or when your favorite author releases a new title? Eyes, our tireless, automated search agent, will send you mail. Meanwhile, our human editors are busy previewing galleys and reading advance reviews. They can let you know when especially wonderful works are published in particular genres or subject areas. Come in, meet Eyes, and have it all explained.

Your Account
Check the status of your orders or change the email address and password you have on file with us. Please note that you do not need an account to use the store. The first time you place an order, you will be given the opportunity to create an account.
Deep Learning (1986 – Today)

1. 

2. 

aws
Artificial Intelligence At Amazon

Thousands Of Employees Across The Company Focused on AI

- Discovery & Search
- Fulfilment & Logistics
- Enhance Existing Products
- Define New Product Categories
- Bring Machine Learning To All

Artificial Intelligence At Amazon
AWS Customers using AI

Netflix Recommendation Engine

Pinterest Lens

- fresh strawberry
- strawberries
- food
- berries
- grow strawberries

TV Shows

Sort by: SUGGESTIONS FOR YOU
ML Process

Business Problem –

ML problem framing

Data Collection

Data Integration

Data Preparation & Cleaning

Data Visualization & Analysis

Feature Engineering

Model Training & Parameter Tuning

Model Evaluation

Are Business Goals met?

No

Yes

Model Deployment

Monitoring & Debugging

– Predictions

Re-training
ML Process: Discovery

- Ask the right questions
- Domain Knowledge
ML Process: Integration – Data Architecture

- **Business Problem**
  - ML problem framing

- **Data Collection**
  - Data Integration
  - Data Preparation & Cleaning
  - Data Augmentation
  - Feature Augmentation

- **Feature Engineering**
  - Model Training & Parameter Tuning
  - Model Evaluation

- **Model Deployment**
  - Monitoring & Debugging
  - Predictions

**Build the Data Platform**
- Amazon S3
- AWS Glue
- Amazon Athena
- Amazon EMR
- Amazon Redshift
- Amazon Kinesis
AWS Big Data Services

Data Sources
- Amazon Elasticache
- Amazon Aurora
- Amazon Athena
- Amazon Redshift
- Amazon EMR
- Amazon Aurora
- Amazon Machine Learning
- Any Open Source Tool of Choice on EC2

Data Science Sandbox
- SciPy
- mxnet
- Anaconda
- Caffe
- RStudio
- torch
- TensorFlow
- AWS QuickSight
- QlikView
- Tableau
- Kibana
- IBM Cognos
- MicroStrategy
- TIBCO
- Jaspersoft

Visualization / Reporting

Data Lake
- Amazon S3
- Amazon Kinesis
- Amazon Glue
- Amazon Athena
- Clusterless SQL Query
- Clusterless ETL
- GPU Instances
- Hadoop / Spark
- AWS Lambda
- Apache Storm on EMR
- Apache Flink on EMR

Any Open Source Tool of Choice on EC2

Real-time Analytics
- Amazon Kinesis
- Amazon Stream Analytics
- Amazon Athena
- Amazon Glue
- Amazon EMR
- Amazon Machine Learning
- Predictive Analytics
- GPU Instances
- Hadoop / Spark
- AWS Lambda
- Apache Storm on EMR
- Apache Flink on EMR

Batch Analytics
- Amazon S3
- Amazon Kinesis
- Amazon Glue
- Amazon Athena
- Clusterless SQL Query
- Clusterless ETL
- GPU Instances
- Hadoop / Spark
- AWS Lambda
- Apache Storm on EMR
- Apache Flink on EMR

Any Open Source Tool of Choice on EC2

Transaction Data

Visualization / Reporting
ML Process: Model Training

- Setup and manage notebook environments
- Setup and manage training clusters
- Write data connectors
- Scale ML algorithms to large datasets
- Distribute ML training algorithms to multiple machines
- Secure and manage model artifacts
ML Process: Model Deployment

- Setup and manage model inference clusters
- Manage and scale model inference APIs
- Monitor and debug model predictions
- Model versioning and performance tracking
- Automate new model version promotion to production (A/B testing)
Amazon SageMaker

A fully managed service that enables data scientists and developers to quickly and easily build machine-learning based models into production smart applications.
Amazon SageMaker

Pre-built notebook instances

Highly-optimized machine learning algorithms

Build
Amazon SageMaker

Pre-built notebook instances

Easier training with hyperparameter optimization

Highly-optimized machine learning algorithms

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

Build

Train

Build

Pre-built notebook instances

Easier training with hyperparameter optimization

Highly-optimized machine learning algorithms

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

**Build**
- One-click training for ML, DL, and custom algorithms
- Highly-optimized machine learning algorithms
- Easier training with hyperparameter optimization

**Deploy**
- Fully-managed hosting at scale
- Deployment without engineering effort

**Train**
- Pre-built notebook instances

**Tools**
- TensorFlow
- PyTorch
- MXNet
- Gluon
- scikit-learn
Amazon SageMaker

Build, train, and deploy machine learning models at scale

End-to-End Machine Learning Platform

Zero setup

Flexible Model Training

Pay by the second
Behind the scenes

Client application

Amazon SageMaker

Amazon ECR

Inference code

Training code

Model Training (on EC2)
Behind the scenes

Amazon ECR

Training data

Model Training (on EC2)

Training code

Helper code

Client application

Amazon SageMaker

Amazon ECR

Inference code

Training code
Behind the scenes

Amazon ECR

Model Training (on EC2)

Amazon SageMaker

Client application

Training data

Model artifacts

Inference code

Training code

Helper code

Client application

Amazon ECR

Training data

Model artifacts

Inference code

Training code

Helper code

Client application

Amazon ECR

Training data

Model artifacts

Inference code

Training code

Helper code
Behind the scenes

Model Training (on EC2)

Model Hosting (on EC2)

Amazon SageMaker

Amazon ECR
Behind the scenes

Model Training (on EC2)
- Training data
- Model artifacts
- Training code
- Helper code

Model Hosting (on EC2)
- Inference request
- Inference response
- Inference code
- Helper code

Amazon SageMaker
- Inference Endpoint

Amazon ECR
- Inference code
- Training code
Behind the scenes

Model Training (on EC2)

- Training code
- Helper code

Model Hosting (on EC2)

- Inference code
- Helper code

Amazon ECR

- Inference code
- Training code

Amazon SageMaker

- Inference request
- Inference response

Client application

- Ground Truth
- Model artifacts

Training data

Helper code
Customer Example: Intuit
Near real-time fraud detection in AWS using Amazon SageMaker
Zero Setup For Exploratory Data Analysis

- EBS
- Access to S3 Data Lake
- ETL Access to AWS Database services
- Authoring Notebooks

“Just add data”

- Recommendations/Personalization
- Fraud Detection
- Forecasting
- Image Classification
- Churn Prediction
- Marketing Email/Campaign Targeting
- Log processing and anomaly detection
- Speech to Text
- More...
Algorithms

Amazon SageMaker: 10x better algorithms

- Streaming datasets, for cheaper training
- Train faster, in a single pass
- Greater reliability on extremely large datasets
- Choice of several ML algorithms
Linear Learner

- Find linear equation that best approximates the data
- Supervised
- Supports:
  - Binary classification
  - Multiclass classification
  - Linear regression
  - Floating-point or test
  - CSV or recordIO-protobuf
- Parallel training of multiple models with automatic hyperparameter optimization
Linear Learner

Regression (mean squared error)

<table>
<thead>
<tr>
<th></th>
<th>SageMaker</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.02</td>
<td>1.06</td>
<td></td>
</tr>
<tr>
<td>1.09</td>
<td>1.02</td>
<td></td>
</tr>
<tr>
<td>0.332</td>
<td>0.183</td>
<td></td>
</tr>
<tr>
<td>0.086</td>
<td>0.129</td>
<td></td>
</tr>
<tr>
<td>83.3</td>
<td>84.5</td>
<td></td>
</tr>
</tbody>
</table>

Classification (F1 Score)

<table>
<thead>
<tr>
<th></th>
<th>SageMaker</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.980</td>
<td>0.981</td>
<td></td>
</tr>
<tr>
<td>0.870</td>
<td>0.930</td>
<td></td>
</tr>
<tr>
<td>0.997</td>
<td>0.997</td>
<td></td>
</tr>
<tr>
<td>0.978</td>
<td>0.964</td>
<td></td>
</tr>
<tr>
<td>0.914</td>
<td>0.859</td>
<td></td>
</tr>
<tr>
<td>0.470</td>
<td>0.472</td>
<td></td>
</tr>
<tr>
<td>0.903</td>
<td>0.908</td>
<td></td>
</tr>
<tr>
<td>0.508</td>
<td>0.508</td>
<td></td>
</tr>
</tbody>
</table>

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

Cost in Dollars vs. Billable time in Minutes

- SageMaker-url
- SageMaker-spam
- Other-url
- Other-spam
Factorization Machines

• Extends linear model to pair-wise interactions between features

\[ \tilde{y} = w_0 + \langle w_1, x \rangle + \sum_{i,j>i} x_i x_j \cdot \langle v_i, v_j \rangle \]

• Good for high dimensional sparse datasets, e.g.:
  • Click prediction
  • Item recommendation systems

• Supports binary classification or linear regression

• recordIO-protobuf data format
Factorization Machines

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

<table>
<thead>
<tr>
<th></th>
<th>Log_loss</th>
<th>F1 Score</th>
<th>Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>SageMaker</td>
<td>0.494</td>
<td>0.277</td>
<td>820</td>
</tr>
<tr>
<td>Other (10 Iter)</td>
<td>0.516</td>
<td>0.190</td>
<td>650</td>
</tr>
<tr>
<td>Other (20 Iter)</td>
<td>0.507</td>
<td>0.254</td>
<td>1300</td>
</tr>
<tr>
<td>Other (50 Iter)</td>
<td>0.481</td>
<td>0.313</td>
<td>3250</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>Billable Time in Hours</th>
<th>Cost in Dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$-</td>
</tr>
<tr>
<td>2</td>
<td>$20,00</td>
</tr>
<tr>
<td>3</td>
<td>$40,00</td>
</tr>
<tr>
<td>4</td>
<td>$60,00</td>
</tr>
<tr>
<td>5</td>
<td>$80,00</td>
</tr>
<tr>
<td>6</td>
<td>$100,00</td>
</tr>
<tr>
<td>7</td>
<td>$120,00</td>
</tr>
<tr>
<td>8</td>
<td>$140,00</td>
</tr>
<tr>
<td>9</td>
<td>$160,00</td>
</tr>
<tr>
<td>10</td>
<td>$180,00</td>
</tr>
<tr>
<td>11</td>
<td>$200,00</td>
</tr>
</tbody>
</table>
K-Means Clustering

• Cluster data into k groups
• Unsupervised
• Algorithm:
  1. Start with K= k*x cluster centers
  2. Iterate over cluster centers in mini-batches and adjust them based on training data
  3. Reduce K cluster centers to k final clusters by using the same algorithm.
K-Means Clustering

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

Running Time vs. Number of Clusters

~10x Faster!
Principal Component Analysis (PCA)

- Reduce dimensionality while retaining as much information as possible
- Finds new components, sorted by information value.
- Unsupervised
- Regular (for sparse) and Randomized (for large, dense datasets) modes
Principal Component Analysis (PCA)

Cost vs. Time

More than 10x faster at a fraction the cost!

Throughput and Scalability

Cost in Dollars

Billable time in Minutes

Mb/Sec/Machine

Number of Machines

Cost vs. Time

Throughput and Scalability
Neural Topic Modeling

Encoder: feedforward net

Input term counts vector

Decoder: Softmax

Sampled Document Representation

Document Posterior

Output term counts vector

Perplexity vs. Number of Topic
(~200K documents, ~100K vocabulary)
Spectral Latent Dirichlet Allocation (LDA)

High-Tech Industry, Long Shy of Politics, Is Now Belle of Ball

By LIZETTE ALVAREZ (14C, 16, 1999)

Correction Appended

At a time when Congress is bitterly divided and unable to reach consensus on issues like gun control and health care, Democrats and Republicans are baspically reaching across party lines to pass legislation backed by high-tech companies.

The high-tech industry, at the same moment, is devolving new attention on Washington and changing its once aloof posture toward the federal government.

Republicans and Democrats are both eager to win the loyalty of high-tech companies and executives, knowing that they represent untold jobs, wealth and ultimately future contributions.

For its part, the industry has realized that the federal government can do its members as much harm as good. Microsoft, and its battle with the Justice Department, along with a spate of other threatened legal problems, drilled this point home.

"Microsoft was a poster child for our industry," said Connie Carroll, director of communications for the Information Technology Industry Council, a trade organization that represents America Online, Dell and I.B.M., among others.
Boosted Decision Trees

- XGBoost gradient boosted trees algorithm
- Combines multiple weak decision tree models
- Supervised
- Binary and multiclass classification
- Libsvm or CSV data
XGBoost is one of the most commonly used implementations of boosted decision trees in the world.

It is now available in Amazon SageMaker!
Sequence to Sequence

- Based on Sockeye Seq2Seq implementation
- Encoder, Attention, Decoder architecture

Sequence to Sequence

Based on Sockeye and Apache incubated MxNet, Multi-GPU, and can be used for Neural Machine Translation.

Supports both RNN/CNN as encoder/decoder

English-German Translation

Best known result!
Image Classification

• Build your own “Rekognition” service!
• Currently based on ResNet
• Configurable number of layers
• Full training and transfer learning
• Data formats:
  • Apache MXNet RecordIO
  • .jpg or .png
Image Classification

Implementation in MxNet of ResNet. Other networks such as DenseNet and Inception will be added in the future.

Transfer learning: begin with a model already trained on ImageNet!
Your Own Algorithms

• Bring your own algorithm!
• Just wrap it into a Docker container
  • One container for training
  • One container for inference
• Combine SageMaker containers with your own
• Documented example on GitHub
Deep Neural Networks

- Train your own Deep Neural Networks!
- TensorFlow and Apache MXNet supported
- You provide training/inference scripts with your DNN
- SageMaker does the rest
Using SageMaker with Apache Spark

• Apache Spark library for Amazon SageMaker provided
• Both Python and Scala
• Makes org.apache.spark.sql.DataFrame objects available to SageMaker
• Training and Inference supported
Algorithms

2. Amazon SageMaker: 10x better algorithms

- Matrix Factorization
- Regression
- Principal Component Analysis
- K-Means Clustering
- Gradient Boosted Trees
- And More!

Amazon provided Algorithms

Bring Your Own Script (SageMaker builds the Container)

SageMakerEstimators in Apache Spark

Bring Your Own Algorithm (You build the Container)
ML Training Service

- Matrix Factorization
- Regression
- Principal Component Analysis
- K-Means Clustering
- Gradient Boosted Trees
- And More!

Amazon provided Algorithms

Bring Your Own Script
(SageMaker builds the Container)

SageMakerEstimators in Apache Spark

Bring Your Own Algorithm (You build the Container)

CPU

GPU

HPO
Versions of the same inference code saved in inference containers. **Prod** is the primary one, 50% of the traffic must be served there!
Model Deployment

Versions of the same inference code saved in inference containers. Prod is the primary one, 50% of the traffic must be served there!

Create a Model

Model Name: prod

Amazon ECR

Amazon SageMaker
Version of the same inference code saved in inference containers. Prod is the primary one, 50% of the traffic must be served there!
Model Deployment

Versions of the same inference code saved in inference containers. Prod is the primary one, 50% of the traffic must be served there!

Create weighted ProductionVariants

InstanceType: c3.4xlarge
InitialInstanceCount: 3
ModelName: prod
VariantName: primary
InitialVariantWeight: 50
Model Deployment

4

Versions of the same inference code saved in inference containers. **Prod** is the primary one, 50% of the traffic must be served there!

Inference Image

Model Artifacts

Amazon ECR

Inference Endpoint

Model versions

EndpointConfiguration

Amazon SageMaker

InstanceType: c3.4xlarge
InitialInstanceCount: 3
ModelName: prod
VariantName: primary
InitialVariantWeight: 50

ProductionVariant

Create an Endpoint from one EndpointConfiguration
Model Deployment

Version 1

Model Artifacts

Inference Image

Amazon ECR

Inference Endpoint

EndpointConfiguration

Model versions

ProductionVariant

One-Click!

Amazon Provided Algorithms

Product

Model Name: prod

VariantName: primary

InitialVariantWeight: 50

InstanceType: c3.4xlarge

InitialInstanceCount: 3

Versions of the same inference code saved in inference containers. Prod is the primary one, 50% of the traffic must be served there!
Model Deployment

- Auto-Scaling Inference APIs
- A/B Testing (more to come)
- Low Latency & High Throughput
- Bring Your Own Model
- Python SDK
Amazon SageMaker—Your Turn

• Getting started with Amazon SageMaker: https://aws.amazon.com/sagemaker/
• Use the Amazon SageMaker SDK:
  • For Python: https://github.com/aws/sagemaker-python-sdk
  • For Spark: https://github.com/aws/sagemaker-spark
• SageMaker Examples: https://github.com/awslabs/amazon-sagemaker-examples
• Let us know what you build!
Thank you!

Constantin Gonzalez
glez@amazon.de
@zalez