



AWS
re:Invent

AIM361-R

Optimizing your machine learning models on Amazon SageMaker

Julien Simon

AI/ML Evangelist
Amazon Web Services

Dr. Steve Turner

Head of Emerging Technologies, UKIR
Amazon Web Services

Agenda

1. Welcome and housekeeping
2. An introduction to automatic model tuning (AMT) and AutoML
3. Labs
4. Wrap-up and cleanup

What you'll learn today

- How to use AMT to find optimal model hyperparameters
- How to use AMT to explore deep-learning architectures
- How to use Amazon SageMaker Autopilot to find the optimal algorithm, data preprocessing steps, and hyperparameters

Our team today

Housekeeping

- Please be a good neighbor 😊
- Turn off network backups and any network-hogging apps
- Switch your phones to silent mode
- Help the people around you if you can
- Don't stay blocked—ask questions!

Automatic model tuning with Amazon SageMaker

Hyperparameters

XGBoost

Tree depth
Max leaf nodes
Gamma
Eta
Lambda
Alpha
Etc.

Which ones
are the most
influential?

Which values
should I pick?

How many
combinations
should I try?

Neural networks

Number of layers
Hidden layer width
Learning rate
Embedding dimensions
Dropout
Etc.

Tactics to find the optimal set of hyperparameters

- **Manual search:** "I know what I'm doing"
- **Grid search:** "X marks the spot"
Typically training hundreds of models
Slow and expensive
- **Random search:** "Spray and pray"
"Random Search for Hyper-Parameter Optimization," Bergstra & Bengio,
Journal of Machine Learning Research, 2012
Works better and faster than grid search
But ... but ... but ... it's random!
- **Hyperparameter optimization:** Use ML to predict hyperparameters
Training fewer models
Gaussian process regression and Bayesian optimization
https://docs.aws.amazon.com/en_pv/sagemaker/latest/dg/automatic-model-tuning-how-it-works.html

Setting hyperparameters in Amazon SageMaker

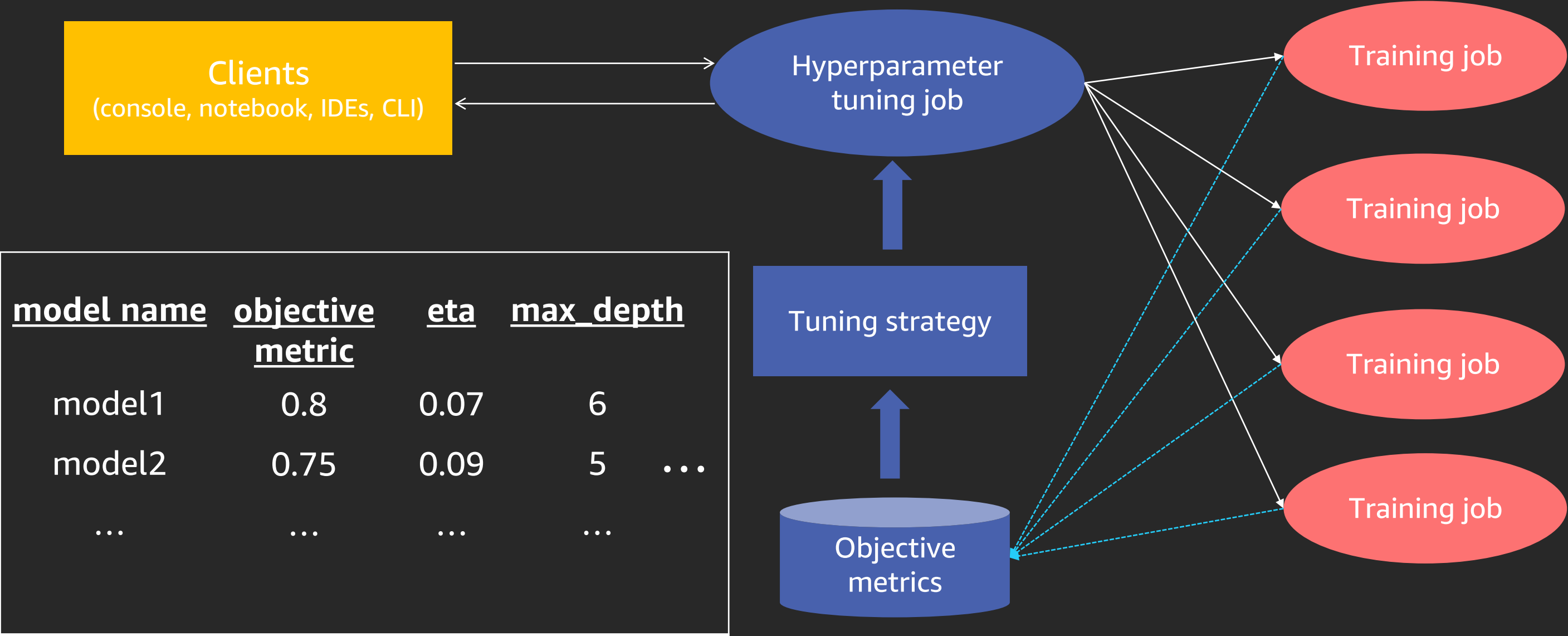
- Built-in algorithms
 - Python **parameters** for the relevant estimator (KMeans, LinearLearner, etc.)
- Built-in frameworks
 - *hyperparameters* parameter for the relevant estimator (TensorFlow, MXNet, etc.)
 - This must be a Python **dictionary**

```
tf_estimator = TensorFlow(..., hyperparameters={'epochs': 1, 'lr': '0.01'})
```
 - Your code must be able to accept them as command-line arguments (**script mode**)
- Bring your own container
 - *hyperparameters* parameter for *Estimator*
 - This must be Python dictionary
 - It's copied inside the container: `/opt/ml/input/config/hyperparameters.json`

Automatic model tuning in Amazon SageMaker

1. Define an *Estimator* the normal way
2. Define the **metric** to tune on
 - Predefined metrics for built-in algorithms and frameworks
 - Or anything present in the training log, provided that you pass a regular expression for it
3. Define **parameter ranges** to explore
 - Type: categorical (avoid if possible), integer, continuous (aka floating point)
 - Range
 - Scaling: linear (default), logarithmic, reverse logarithmic
4. Create a *HyperparameterTuner*
 - *Estimator*, metric, parameters, total number of jobs, number of jobs in parallel
 - Strategy: Bayesian (default) or random search
5. Launch the tuning job with *fit()*

Workflow



Automatic model tuning in Amazon SageMaker

- You can **view** ongoing tuning jobs in the AWS console
 - List of training jobs
 - Best training job
- You can also **query** their status with the Amazon SageMaker SDK
- Calling ***deploy()*** on the *HyperparameterTuner* deploys the best job
 - The best job so far if the tuning job has not yet completed

Tips

- Use the Bayesian strategy for better, faster, cheaper results
 - Most customers use random search as a baseline, to check that Bayesian performs better
- Don't run too many jobs in parallel
 - This gives the Bayesian strategy fewer opportunities to predict
 - Instance limits!
- Don't run too many jobs
 - Bayesian typically requires 10x fewer jobs than random
 - Cost!

Resources on automatic model tuning

Documentation

<https://docs.aws.amazon.com/sagemaker/latest/dg/automatic-model-tuning.html>

<https://sagemaker.readthedocs.io/en/stable/tuner.html>

Notebooks

https://github.com/awslabs/amazon-sagemaker-examples/tree/master/hyperparameter_tuning

Blog posts

<https://aws.amazon.com/blogs/aws/sagemaker-automatic-model-tuning/>

<https://aws.amazon.com/blogs/machine-learning/amazon-sagemaker-automatic-model-tuning-produces-better-models-faster/>

<https://aws.amazon.com/blogs/machine-learning/amazon-sagemaker-automatic-model-tuning-now-supports-early-stopping-of-training-jobs/>

<https://aws.amazon.com/blogs/machine-learning/amazon-sagemaker-automatic-model-tuning-becomes-more-efficient-with-warm-start-of-hyperparameter-tuning-jobs/>

<https://aws.amazon.com/blogs/machine-learning/amazon-sagemaker-automatic-model-tuning-now-supports-random-search-and-hyperparameter-scaling/>

AutoML with Amazon SageMaker Autopilot

AutoML

- AutoML aims at automating the process of building a model
 - **Problem identification**: Looking at the dataset, what class of problem are we trying to solve?
 - **Algorithm selection**: Which algorithm is best suited to solve the problem?
 - **Data preprocessing**: How should data be prepared for best results?
 - **Hyperparameter tuning**: What is the optimal set of training parameters?
- Black box vs. white box
 - Black box: The **best model** only
 - Hard to understand the model, impossible to reproduce it manually
 - White box: The **best model**, other **candidates**, full **source code** for preprocessing and training
 - See how the model was built and keep tweaking for extra performance

AutoML with Amazon SageMaker Autopilot

- Amazon SageMaker Autopilot covers all steps
 - **Problem identification**: Looking at the dataset, what class of problem are we trying to solve?
 - **Algorithm selection**: Which algorithm is best suited to solve the problem?
 - **Data preprocessing**: How should data be prepared for best results?
 - **Hyperparameter tuning**: What is the optimal set of training parameters?
- Amazon SageMaker Autopilot is **white box** AutoML
 - You can understand how the model was built, and you can keep tweaking
- Supported **algorithms** at launch:
Linear Learner, Factorization Machines, KNN, XGBoost

AutoML with Amazon SageMaker Autopilot

1. Upload the **unprocessed dataset** to Amazon S3
2. Configure the AutoML job
 - Location of dataset
 - Completion criteria
3. Launch the job
4. View the list of **candidate** and the **auto-generated notebook**
5. Deploy the **best candidate** to a real-time endpoint, or use batch transform

Labs

Labs

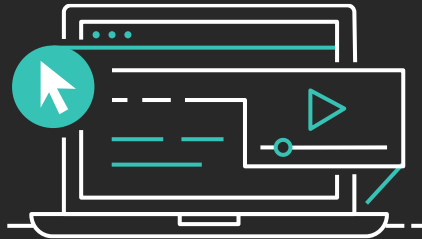
1. Use AMT to find optimal model hyperparameters for XGBoost
2. Use Amazon SageMaker Autopilot to find the optimal algo, preprocessing steps, and hyperparameters
3. Use AMT to explore deep-learning architectures on Keras

Learn ML with AWS Training and Certification

The same training that our own developers use, now available on demand



Role-based ML learning paths for developers, data scientists, data platform engineers, and business decision makers



70+ free digital ML courses from AWS experts let you learn from real-world challenges tackled at AWS



Validate expertise with the
AWS Certified Machine Learning - Specialty exam

Visit <https://aws.training/machinelearning>

Thank you!



Please complete the session
survey in the mobile app.