Build Models Faster, and Serve Predictions at Scale, using Amazon SageMaker Feature Store

Mark Roy
Principal ML Architect
AWS
Who’s really using feature stores?
Feature store usage cuts across industries and use cases

- Financial services
- Consumer credit
- Insurance
- Online shopping
- Travel
- Dating
Agenda

• Feature Store overview
• Deep dive
• Demo
• Resources
• Q&A
Why bother using a feature store?
60% of time spent on data preparation

Source: Forbes survey of 80 data scientists, March 2016
… and teams too often start from scratch

Standalone feature engineering for each new model

Data sources

Vehicle damage images
Claims, customers, ...
Text interactions
Telemetry

Vehicle total loss
Team 1
… and teams too often start from scratch

Standalone feature engineering for each new model

Data sources

Vehicle damage images
Claims, customers, ...
Text interactions
Telemetry

Vehicle total loss
Team 1
… and teams too often start from scratch

Standalone feature engineering for each new model

Data sources

- Vehicle damage images
- Claims, customers, ...
- Text interactions
- Telemetry

?  
Customer churn  
Team 2
… and teams too often start from scratch

Standalone feature engineering for each new model

Data sources

- Vehicle damage images
- Claims, customers, ...
- Text interactions
- Telemetry

Customer churn

Team 2
… and teams too often start from scratch

Standalone feature engineering for each new model

Data sources
- Vehicle damage images
- Claims, customers, ...
- Text interactions
- Telemetry

Next best action

Team N
… and teams too often start from scratch

Standalone feature engineering for each new model

Challenges
• Slow time to market
• Feature duplication
• Inaccurate predictions
With SageMaker Feature Store…

Build features once, reuse them across teams and models

Data sources

- Vehicle damage images
- Claims, customers, ...
- Text interactions
- Telemetry

Feature pipelines

SageMaker Feature Store

Models, Endpoints

Benefits

- Feature reuse
- Reproducible features
- Accurate training datasets
- Low latency inference
- Consistent features for training and inference
- Managed service

- Vehicle total loss
- Customer churn
- Next best action
Amazon SageMaker Feature Store

Securely store, discover, and share features for both real-time inference and training

Batch and streaming ingestion
High throughput writes for ingesting features

Online and offline features
Online features for real-time prediction, and offline features for historical data for model training and batch prediction

Feature metadata, automatic data catalog entries
Store metadata for features, and automatically create a data catalog to easily query and extract feature data

Feature discovery and reuse
Search for features to reuse, before starting new development

Security and access control
Access control for feature data and feature metadata, and support for encryption at rest, Amazon VPC, and AWS PrivateLink

Fully managed
Online features cached in low-latency store; maintain consistency between online and offline store to avoid train-infer skew
Feature store in context

Data sources
- Amazon Kinesis
- Amazon MSK

SageMaker Feature Store
- Online store
- Offline store

Batch feature pipelines
- Streaming feature pipelines

Model endpoints
- Model training
- Batch scoring

© 2021 Amazon Web Services, Inc. or its affiliates. All rights reserved
Using Amazon SageMaker Feature Store with other services

- Feature authoring:
  - SageMaker Data Wrangler
  - Glue
  - DataBrew

- Feature discovery:
  - SageMaker Studio

- Feature processing:
  - EMR
  - Glue
  - SageMaker Processing

- Streaming:
  - Kinesis
  - Kafka

- Online inference:
  - SageMaker Hosting
  - Lambda

- Training, batch scoring:
  - SageMaker Training
  - Batch Transform
  - Athena

- Feature pipelines:
  - Step Functions
  - SageMaker Pipelines
  - Apache Airflow
  - Amazon EventBridge
Working with Amazon SageMaker Feature Store

- Feature ingestion
- Offline store queries
- Online feature retrieval
Feature ingestion APIs

PutRecord API

```python
record = [{'FeatureName': 'feature_1', 'ValueAsString': 'val_1'},
          ...
          {'FeatureName': 'feature_N', 'ValueAsString': 'val_N'}]
sm_fs.put_record(FeatureGroupName='my-fg-name', Record=record)
```

Python SDK

```python
fg = FeatureGroup(name='my-fg-name',
                  sagemaker_session=fs_session)

fg.ingest(df, max_processes=20, max_workers=4)
```
# PutRecord behavior

- Online store keeps **latest feature values** for each record identifier
- Online feature values are available **immediately**, for use by any model
- Offline store appends each new record, keeping a **history of all feature values**

![Feature Group after all PutRecord calls](image)

# 3 uniquely identified records

```plaintext
put_record('id-1', t1, 0.1)
put_record('id-2', t2, 0.2)
put_record('id-3', t3, 0.3)
```

# new records for existing id

```plaintext
put_record('id-1', t4, 0.4)
put_record('id-1', t5, 0.5)
```
Scalable bulk ingestion with Python SDK

Ingestion processing job – Python SDK

Shard files by S3 key

Instance 1

N processes, M workers

Instance 2

...

Instance N

Parallel calls to PutRecord

Feature Group

Online store

Offline store

s3://my-bucket/east/001.csv ... 050.csv
west/001.csv ... 020.csv
central/001.csv ... 030.csv

Each instance builds a Pandas dataframe, and ingests it using `fg.ingest(df, N, M)`
Scale up and out to speed up feature pipelines

Drive higher throughput with larger instances or more instances

- 1 m5.4xl
- 2 m5.4xl
- 4 m5.4xl
- 8 m5.4xl
- 2 c5.9xl
- 3 c5.9xl

© 2021 Amazon Web Services, Inc. or its affiliates. All rights reserved
Use SQL to create datasets from offline stores

**Training**

```sql
SELECT loyalty_status, last_purch_date, last_purch_amt, churn_label
FROM customers
```

**Batch scoring**

```sql
SELECT id, loyalty_status, last_purch_date, last_purch_amt
FROM customers
```

---

**Offline feature store**

- customers
- orders
- claims
- visits

**Amazon Athena**

**AWS Glue Data Catalog**

**SageMaker Training**

- customer_train.csv

**SageMaker Batch Transform**

- customer_score.csv
Query features interactively, or with Python SDK

```
1 select purchase_amount, is_married, p.*
2 from "customers-1628610865" c,
3   "orders-1628616113" o,
4   "products-1628610886" p
5 where c.customer_id = o.customer_id and
6   o.product_id = p.product_id

s = f"SELECT COUNT(*) FROM "{fg.athena_query().table_name}" ' + "WHERE f7_date = '2020-03-31'"
q = feature_group.athena_query()
q.run(s, output_location=output_location)
q.wait()
df = q.as_dataframe()
```
Online feature retrieval for inference

**GetRecord API**

```python
sm_fs.get_record(FeatureGroupName='customer-fg',
                 RecordIdentifierValueAsString='CUST-001',
                 FeatureNames=['spend-last7d', 'tenure-days'])
```

**BatchGetRecord API**

```python
sm_fs.batch_get_record(Identifiers=[
    {'FeatureGroupName': 'customer-fg',
     'RecordIdentifierValueAsString': ['CUST-001', 'CUST-002', 'CUST-003'],
     'FeatureNames': ['spend-last7d', 'tenure-days']},
    {'FeatureGroupName': 'product-fg',
     'RecordIdentifierValueAsString': ['P-100', 'P-200'],
     'FeatureNames': ['orders-last7d', 'daily-revenue']}
])
```
Demo
Resources

- Workshop
- Blog posts
- Documentation
SageMaker Feature Store workshop

New workshop repo gives you end-to-end hands-on introduction to SageMaker Feature Store - [link](https://github.com/aws-samples/amazon-sagemaker-feature-store-end-to-end-workshop)
SageMaker Feature Store blog posts

Understanding key capabilities - [link](#)

Using streaming ingestion to make ML-backed decisions in near-real time - [link](#)

Automating feature pipelines - [link](#)

Directly ingesting historical feature data to S3 - [link](#)

Building accurate training datasets using point-in-time queries on Apache Spark - [link](#)

Enabling feature reuse across accounts - [link](#)

Scaling batch ingestion - [link](#)

Using feature store in a Java environment - [link](#)
SageMaker Feature Store documentation

Developer guide link

SageMaker Python SDK link

Boto3 control plane API link

Boto3 runtime API link

© 2021 Amazon Web Services, Inc. or its affiliates. All rights reserved
Thank you!

Do you have any questions?

Mark Roy
Principal ML Architect, AWS

https://www.linkedin.com/in/markproy/