[Feature Store Summit]
Jukebox

Aman Khan, Product Manager
Daniel Kristjansson, Staff Engineer
Overview

- ML Platform at Spotify
- What is Jukebox
- How it works
- Why it works like this
- Where we're going
  - Jukebox API
  - Feature Workflows
  - Feature Marketplace
Active Users: 365 Million
Tracks: 70 Million
Podcasts: 2.9 Million
Playlists: 4 Billion
Available in: 178 Markets
Models Trained: 30 Thousand
Prediction Requests: 300 Thousand/Second
Teams on ML Platform: 50
Refresher

Raw Data

```json
0: {
  house_info: {
    num_rooms: 6
    num_bedrooms: 3
    street_name: "Shorebird Way"
    num_basement_rooms: -1
    ...
  }
}
```

Feature Vector

```
[6.0,
  1.0,
  0.0,
  0.0,
  0.0,
  9.321,
  -2.20,
  1.01,
  0.0,
  ...
]
```

Feature Engineering

Process of creating features from raw data is **feature engineering**.

Raw data doesn't come to us as feature vectors.
Refresher

Raw Data

```json
0: {
  house_info: {
    num_rooms: 6,
    num_bedrooms: 3,
    street_name: "Shorebird Way",
    num_basement_rooms: -1
  }
}
```

Feature Engineered

```json
[ 6.0, 1.0, 0.0, 0.0, 0.0, 9.321, -2.20, 1.01, 0.0, ...
]
```

Feature Vector

Raw data doesn't come to us as feature vectors.

Jukebox helps manage this for model use case.

Process of creating features from raw data is feature engineering.
Features at Spotify

Technical

- Dynamic features
- Near "real time" features
- Complexity between trying to bridge offline and online

Organization

- Highly autonomous culture
- 50+ squads, across different use cases, recreating and serving the same features

Features are a huge problem
A brief history of our platform
Quick background on the ML Platform

5 products that serve various stages of the ML Lifecycle
Refresher

Real-Time Data

Raw Data

Batch Data

Feature Store

Transform

Store

Serve

Online and Offline Predictions

Feature Data

Model Training
Where we were

- Python and JVM components that help manage data

**Feature Registry And Gallery**
- Register features with metadata for ease of consumption and reproducibility

**Feature Preparation**
- **Converter**
  - Convert Avro or BigQuery datasets into TFRecords
- **Collector**
  - Select and join features from upstream data endpoints

**Feature Serving**
- **Loader**
  - Load TFRecords (or Protobuf) into BigTable or BigQuery
- **Online Reader**
  - Fetch registered features by name for online serving

Decentralized management through libraries
Jukebox 2.0 Overview

- Functionality exposed as a service API
  - Make features available for Jukebox ecosystem (load)
  - Prepare training dataset using point in time joins
  - Load feature set for online inference (Bigtable)
Jukebox 2.0 Overview

- Service will automatically:
  - Convert between formats
  - Register Bigtable + GCS locations in feature registry
  - Join datasets
  - Compute statistics

- Benefits
  - Centralized Management
  - User Experience
  - Development

User focused, centralized workflow
What's new?

Jukebox 1.0

- Reference sources between training and serving workflows (error prone)
- Library dependencies and version maintenance for latest and greatest
- No trust in reusing other team's features
- No connection to rest of ML Platform

Jukebox 2.0

- Register your features and reference them by name (simplifies code and reduces chance of incorrect definition)
- No more managing pesky library versions or dependencies to get the latest Jukebox updates
- <soon> Profiling information in the Feature Marketplace UI, and automatic feature fetching in Salem
- *Bonus* Point in time joins!
Feature Workflows

Generate  Train  Load  Serve
Feature Workflows
Home Use Case

Goal: Listener finds a session quickly that will sustain their satisfaction with Spotify

- Many models, multiple teams
- Daily, near-real-time and contextual features
- Five feature stores*
Feature Sources

Logs - Impression, Interaction, EndSong, etc

- Scio pipelines
- BigQuery

Embeddings - Music, Users, Episodes, etc

- Bag-of-words
- BERT

Context

- Local time, device, etc.
Feature Engineering

- Hypothesis
- Data Engineering
- Modeling/Embedding
- Offline testing
- Online testing
- Ship
Feature Engineering -- trouble

Home x Salem

#home-ml-ops

Rule #29: The best way to make sure that you train like you serve is to save the set of features used at serving time, and then pipe those features to a log to use them at training time.
Feature Engineering

Salem - new capabilities contribute to longer feature iteration cycles

- Feature logging for everyone :D
- Longer Feature Engineering workflow for everyone :(  
- Longer running A/B tests :( 
Feature Engineering

Killer features for fast iteration

- Point-in-time joins
- Automated backfills
- Streaming ingestion
Feature Engineering

- Zipline
- Airbnb
- Vertex Feature Store
- Hopsworks
- Jukebox 2.0
- FEAST
- Kaskada
- Tecton
Adding a new feature

First steps

- Register Feature
- Offline Load
- Prepare Training Set
# Artist preference example - load all offline endpoints
# Use parts of the first as a template for the rest of the jukebox.load-

```yaml
- &jukebox-apm-defaults
  id: jukebox.load-offline.apm.artist_entities
  schedule: daily
  service_account: jukebox-examples@sp-ml-infra.iam.gserviceaccount.com
  docker_image: gcr.io/sp-ml-infra/jukebox-luigi:latest
  docker_args:
    'load-offline',
    'features', '{"/entity/artist/component/jukebox-apm-sample/name/genres":"genres","/entity/artist/component/jukebox-apm-sample/name/popularity_normal"
    'keys', '[["primitive_entity": "PRIMITIVE_ENTITY_ARTIST", "identifier": "IDENTIFIER_BASE62", "column_name": "artist_gid"],
    'data-endpoint', 'jukebox.examples.apm.artist_entities',
    'storage-format', 'STORAGE_FORMAT_RF_EXAMPLE',
    'project', 'sp-ml-infra',
    'uri-prefix', 'gs://sp-ml-infra-temp-eu/apm/',
    'date', '{}'
```
Create a Training Set

Feature Producer: Register Feature & Offline Load
Training

1. Create label dataset in training pipeline

2. **Prepare Training Dataset** is called providing label dataset (logs) and features

3. Workflow job is run that will collect feature values at the right point in time
Online Data Loader

1. Features are specified to be loaded in an online store for serving, or for being refreshed.
2. **Online Load** is called specifying features to loaded.
3. Workflow is run, feature values are collected and loaded into BT.
4. Online Source is created to reference newly loaded data in BT is created.

**Diagram:**
- **Serving Pipeline:** User Pipeline -> OnlineLoad -> Prepare Training Set
- **Jukebox 2.0:** Offline Load
- **Feature Registry:**
  - Offline Sources
  - Online Sources
  - Artists Features
  - User Features
  - User Artist Features

**Feature Available online:** BigTable
Online Load Example

```python
# Artist preference example - load into Bigtables by entity
- <<: *jukebox-apm-defaults
  id: jukebox.load-online.apm.artist
  docker_args: [
    'load-online',
    '--features', '{"/entity/artist/component/jukebox-apm-sample/name/genres":"genres","/entity/artist/component/jukebox-apm-sample/name/popularity_normal'...
    '--project', 'sp-ml-infra',
    '--bigtable-instance', 'jukebox-apm',
    '--bigtable-table', 'artist',
    '--date', '{}'
  ]
```
Online Serving

Consuming a feature
Online Serving

Idealized feature consumer view
Home Use Case

- Many models, multiple teams
- Daily, near-real-time and contextual features
- Five feature stores in all
Home Use Case

Many models, multiple teams

- Adoption can be piecemeal
- Feature registry useful for internal feature sharing
- API makes adding each feature easier
Feature Gallery
# Gallery Homepage

**Spotify Machine Learning β**

**Feature Gallery 🌟**

All Machine Learning features available through our Feature Marketplace. For more details on how to register a feature, check out the Feature Documentation.

<table>
<thead>
<tr>
<th>Feature Family</th>
<th>Entity</th>
<th>Feature Count</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>/show/showNormalizedMusicVector.batch</td>
<td>show</td>
<td>1</td>
<td>/show/PromotionsTargeting_JoinedShowNewFeatures.gcs/showNormalizedMusicVector batch</td>
</tr>
<tr>
<td>/entity/user_artist/component/user-artist-interactions-pipeline/days_28_summary.total_skips</td>
<td>user_artist</td>
<td>1</td>
<td>/entity/user_artist/component/user-artist-interactions-pipeline/name/days_28_summary.total_skips</td>
</tr>
<tr>
<td>/user_artist/days_7_summary.top_track_counts.by_count</td>
<td>user_artist</td>
<td>1</td>
<td>/user_artist/user-artist-interactions-pipeline/days_7_summary.top_track_counts.by_count</td>
</tr>
<tr>
<td>/user_artist/splitStreamsPerTrack_7dWindowMaxTracks.byPlatform</td>
<td>user_artist</td>
<td>1</td>
<td>/user_artist/ListenerType/splitStreamsPerTrack_7dWindowMaxTracks.byPlatform</td>
</tr>
<tr>
<td>/user_artist/splitStreamsPerTrack_maxStreamsPerTrack28DayWindow.byTopType</td>
<td>user_artist</td>
<td>1</td>
<td>/user_artist/cccd-streams/splitStreamsPerTrack_maxStreamsPerTrack28DayWindow.byTopType</td>
</tr>
<tr>
<td>/user_adcreative/realtimegenre</td>
<td>user_adcreative</td>
<td>1</td>
<td>/user_adcreative/paradox-as/realtimegenre</td>
</tr>
<tr>
<td>/user/avg_danceability</td>
<td>user</td>
<td>2</td>
<td>/user/jukebox-apm-sample/avg_danceability, /user/ml-golden-path/avg_danceability</td>
</tr>
</tbody>
</table>
# Feature Overview

Feature Family is a group of features that share the same entity (i.e. user), and may express a similar idea or measurement (i.e. country). They can span across different sources.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Entity</th>
<th>Owner</th>
<th>Online Source</th>
<th>Offline Source</th>
<th>Lineage</th>
</tr>
</thead>
<tbody>
<tr>
<td>/track/catapult-metadata-entities/artists</td>
<td>track</td>
<td>catapult</td>
<td>N/A</td>
<td>MetadataEntities.Track.gcs</td>
<td>0 Models</td>
</tr>
</tbody>
</table>

1 of 1
## Feature Details

### Feature Details

<table>
<thead>
<tr>
<th>Feature Family</th>
<th>/track/artists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>/track/capapult-metadata-entities/artists</td>
</tr>
<tr>
<td>Entity</td>
<td>track</td>
</tr>
<tr>
<td>Owner</td>
<td>catapult</td>
</tr>
<tr>
<td>Online Source</td>
<td>N/A</td>
</tr>
<tr>
<td>Offline Source</td>
<td>MetadataEntities.Track.gcs</td>
</tr>
<tr>
<td>Description</td>
<td>Lists the artists who contributed to a track specifying their role and order. The same artist can have different roles. The same role can have multiple artists that can be ordered using the artist_ordinal_number field.</td>
</tr>
<tr>
<td>Scope</td>
<td>SCOPE_UNDEFINED</td>
</tr>
<tr>
<td>Life Cycle</td>
<td>LIFECYCLE_UNDEFINED</td>
</tr>
<tr>
<td>Data Type</td>
<td>DATA_TYPE_ANY</td>
</tr>
<tr>
<td>Lineage</td>
<td>0 Models</td>
</tr>
</tbody>
</table>

### Offline Source

Data Endpoint: MetadataEntities.Track.gcs

### Scope

Scope from Data Endpoint.

### Delivery

#### Daily Delivery

Status Legend

- Status Legend
- Monitor this endpoint in Data Monitoring tool

<table>
<thead>
<tr>
<th>DELIVERY TIME - P90</th>
<th>SLO COMPLIANCE</th>
<th>SLO BREACHES</th>
<th>BREACH DURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4h 49m</td>
<td>100.0%</td>
<td>0</td>
<td>0 min</td>
</tr>
</tbody>
</table>

#### SLO Configured

- SLO Configured
- 10 hrs

#### Status

- 2021-09-27 is OK
- TRY SLO CALCULATOR
In summary

**Usability**
Abstracts away complexity of feature workflow, and makes it easy to implement features

**Trust**
Jukebox enables a rich metadata layer containing information about ML model association, feature quality, and more

**Built for reuse**
Build different datasets, serve different models – you won’t need to build another data pipeline that joins features together or another backend service to query it when you need to fetch actual feature values for serving your ML model
Where we're going - Feature Marketplace

Saving Spotifiers time, money and toil through feature reuse and enhanced feature management.

- Builds on top of our API strategy and infrastructure
- Platformize the interface between producers and consumers
- Enhance trust with information: statistics, lineage, ownership
Future Improvements

Storage tradeoff for cost and performance

Automation

Model interpretability

Feature Store Meetup
May 2021
Spotify
Lessons Learned

- Listen to your customers (and if you can, rope them in to work with you)
- Feature stores are an evolving space
- Think about scale
Thank you!

Do you have any questions?

Aman Khan
amank@spotify.com

Daniel Kristjansson
danielk@spotify.com


@_amankhan