# Scaling Real-time Feature Generation

BEAM SUMMIT NYC 2025

- Rakesh
- Sr. Staff Eng @ Lyft

# Agenda

- Context
- Data Consistency & Accuracy
- Feature Serving
- Evolution of Pipeline
- Performance Optimization
- Data Lineage & Traceability
- Strategies for Pipeline Design

# Context

Lyft is a leading rideshare company that facilitates millions of rides across hundreds of cities. The company leverages sophisticated data processing and machine learning infrastructure to optimize operations and ensure scalable efficiency across all current cities.

# Context Continued..

Use Cases of data processing and ML.

- Dynamic Pricing & Demand Forecasting
- Matching & Routing
- Fraud Detection & Prevention
- Safety
- Etc.

# Context Continued..

- DS/ML workflow
  - Develop model based on offline features
  - Offline features based on offline table queries
  - Productionisation of model require online feature
  - Convert offline queries into online feature pipelines
  - Add Validation in place to ensure data quality & consistency
  - Run online experimentation which could last more than a month
  - After successful experiment, rollout the model in production

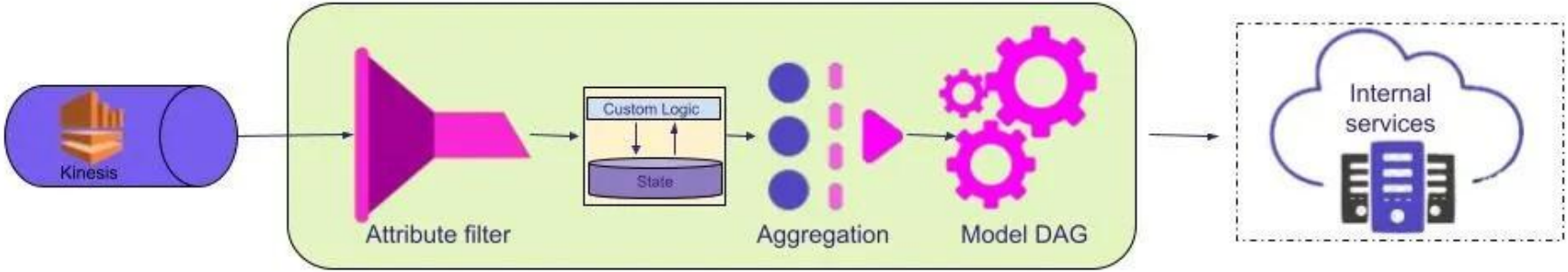
# Data Consistency & Accuracy

- Data consistency & Accuracy is important for model performance
- Proprietary framework to compare online feature with ground truth
- Most of the MAPE is < 0.05%

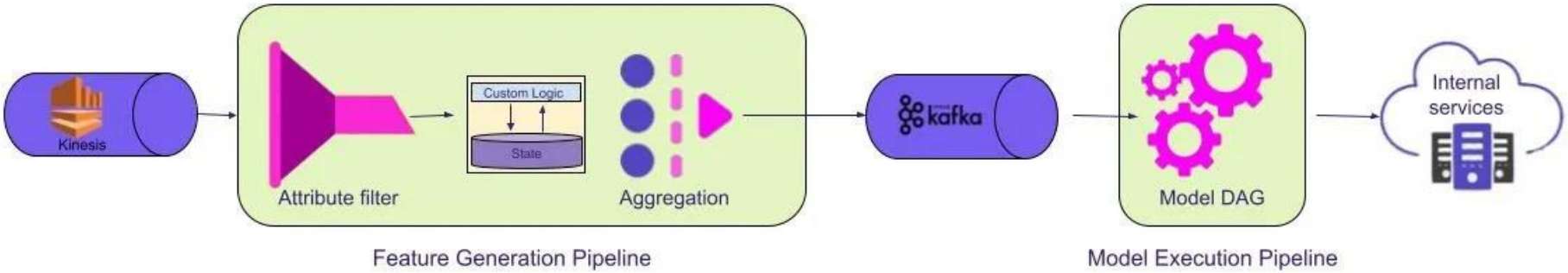
## Feature Serving

- Sync (Service based) Vs Async Mode (Pub-sub based)
- Point feature Vs Geospatial aggregated features

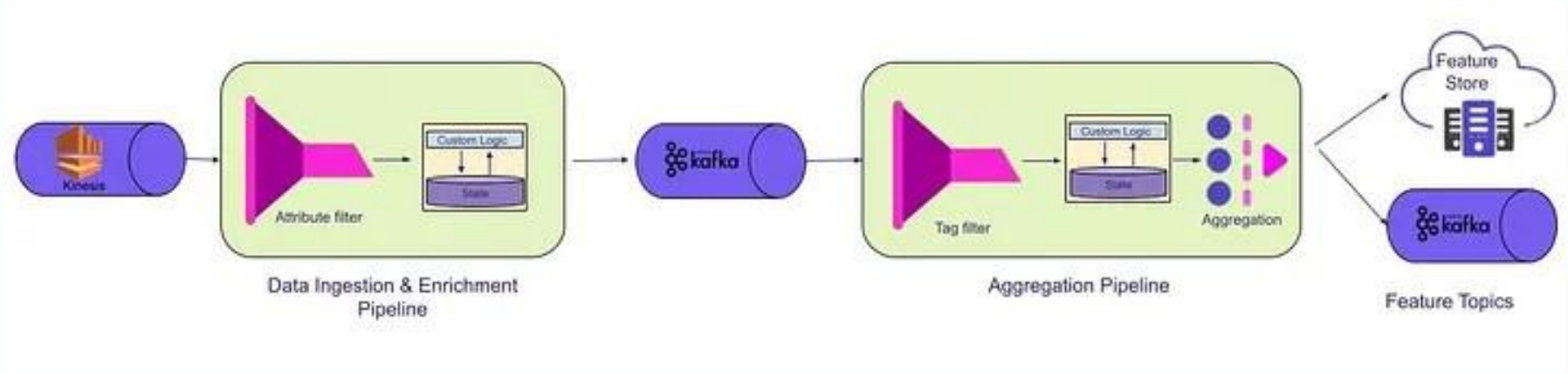
# Evolution of Pipeline: Single Pipeline



# Evolution of Pipeline: SoC



# Evolution of Pipeline: Pipeline Sharing



# Evolution of Pipeline:  Yaml Based

```yaml
- feature_name: sessions
  sources:
    - type: kafka
      proto_type: UserState
      timestamp_field: last_updated_at_ms
      kafka_topic_name: realtimefeatures
  aggregation:
    - aggregation_name: 1min_agg
      window_type: sliding
      window_size_sec: 1800
      region_group_fn: featureaggregation.session.grouping_function
      fn: featureaggregation.session.agg_sessions_proto
  sinks:
    - sink_name: featureservice
      fn: services.featureservice.update_new_sessions
    - sink_name: offlinestore
      fn: services.offline_store.write_to_offlinestore
```

# Performance Optimization

- Understand your data
- Filter early to minimize data size
- Distribute data processing across operators
  - Vertical distribution
  - Horizontal distribution
- Avoid mega pipelines
- Partition data strategically
  - May require multiple partition
- State management
  - Purge state as early as possible
- Partition sources smartly

# Data Lineage & Traceability

- Data Visibility & Traceability is important for an efficient operation
- Need 360° View of Data
- Should be able to identify producer and consumers easily.

# Strategy of Feature  Pipeline Design

- Avoid God Pipeline
- Understand your data
- Identify data skewness early on
  - Avoid huge performance impact
- App based observability is important
  - Should be able to monitor each of the operator performance
- Build for customers
  - Understand customers requirements
  - Make it easier for them to build a pipeline (e.g Yaml based pipelines)
- Make ownership as first class citizen

Rakesh Kumar

# QUESTIONS?

linkedin.rakeshkumar.info
rakeshkumar.info

BEAM SUMMIT NYC 2025