

### Feature engineering at scale with Dagger







## **Ravi Suhag**

Founder and Principal Architect, Open DataOps Foundation

www.ravisuhag.com





### **Feature Engineering**

The process of transforming raw data into high quality input signal for models that better represent the underlying problem to the models.



https://github.com/odpf

Picture Credits: GCP





### **Need for Feature Engineering**

### Better features means flexibility

Better features means simpler models.

Better features means better results.

Good data structure allows for better flexibility in choosing models

Easier to pick right parameters and models

Quality of model results is dependent on quality of features



### **Feature Engineering Challenges**

- Inconsistency between training and serving
- Managing data pipelines
- Scaling data infrastructure
- Lack of standardization
- Real-time features required skilled data engineers



### **Feature Engineering Platform Goals**

- Unified processing for streaming and batch
- Self-service platform
- Elastic infrastructure
- Standard and reusable way for feature engineering
- No added skill needed for real-time features.





## Stream processing made easy

Open source stream processing framework for transforming, aggregating and enriching data with ease of operation and reliability.





Configuration over code, cloud-native framework built on top of Apache Flink for stateful processing of data.



📚 Others

Prometheus



## Feature store for machine learning

Feature store the fastest path to operationalizing analytic data for model training and online inference.



### **Dagger Architecture**







### **Dagger Key Features**

#### **SQL First**

Query writing made easy through formatting, suggestions, auto-completes and template queries.

#### Flexibility

Add custom business logic in form of plugins with UDFs, Transformers, Preprocessors and Post Processors.

#### **Stream Enrichment**

Enrich Kafka messages from API endpoints or database sources to bring offline & reference data context to real-time processing.

Learn more: https://odpf.github.io/dagger/

# Self service platform

Data Scientists can create and manage their feature engineering jobs through a complete self-service platform.

0	Dagger Edit		p-goid-indonesia 🗸 🌲 🕜 🌍
Q	Route Type Tracking 📌 Stops in : 2 days	X Close 🔒 Savepoint	ල Upgrade 🕑 Reset 🔘 Stop
	General Data Sink Settings	Query Editor	🎦 Templates 🛛 Emplates 🔚 Shortcuts 🚦 Full Screen
((0))	Firehose allows smooth and effortless consumption of messages from data streams. This data can then be used into different downstream applications like an HTTP service.	1 INSERT INTO easy_drinks 2 VALUES (Oh My Gosh', 'soda', 1.5, 'pimeapple juice', 1, 'stir with ice, strain i	nto shot glass');
~~	Team Name		
ø	Team name v	8	
4	Privacy		
	Select privacy details		
₽: <sup>1</sup>	Description (optional)		
	Tell people about what is this		
Q			
=		Logs Output Results Monitoring Exceptions	🚦 Full Screen
-		g-gojek-id-vigilante-service-device-login-firehose-5f6758fvdp7x → [pool-1-thread-1] INFO com.gojek.esb.sink.ExponentialBackOffProvider - ba	cking off for 60000 milliseconds
٩Ξ		[pool-1-thread-1] INFO com.gojek.esb.sink.SinkWithRetry - Retrying messag [pool-1-thread-1] INFO com.gojek.esb.sink.htp.HttpSink - pushing 1 messa [pool_1-thread-1] INFO com.gojek.esb.sink.htp.HttpSink-10etraSinkFilore	es attempt count: 9946 ges - Response Status, 500
(		<pre>[pool-1-thread-1] INFO com.gojek.esb.sink.http.client.BasidwitpSinkOlient [pool-1-thread-1] INFO com.gojek.esb.sink.ExponentialBackOffProvider - ba</pre>	- Response Status: 500 cking off for 60000 milliseconds
Ē		[pool-1-thread-1] INFO com.gojek.esb.sink.SinkWithRetry - Retrying messag [pool-1-thread-1] INFO com.gojek.esb.sink.http.HttpSink - pushing 1 messas [pool-1_thread-1] INFO com.gojek.esb.sink.http.liont Basi/HttpSinkClient	es attempt count: 9948 ges - Response Statue: 500
		[poil-ithread-1] INO com.gojek.esb.sink.ExponentialBackOffProvider - ba [pool-i-thread-1] INO com.gojek.esb.sink.SinkWithRetry - Retrying messag [pool-i-thread-1] INO com.gojek.esb.sink.http.HttpSink - pushing I messa [pool-ithread-1] INO com.gojek.esb.sink.http.IttpSink - pushing I messa [pool-ithread-1] INO com.gojek.esb.sink.http.IttpSinkClinatureS	cking off for 60000 milliseconds es attempt count: 9949 ges - Resonnse Status: 500
		[pool-1-thread-1] INFO com.gojek.esb.sink.ExponentialBackOffProvider - ba [pool-1-thread-1] INFO com.gojek.esb.sink.SinkWithRetry - Retrying messag	cking off for 60000 milliseconds es attempt count: 9942
>	[pool-1-thread-1] INFO com.gdjek.esb.sink.http.HttpSink - pushing 1 messages Save Cancel >		ges

Use below link for more details on case study: www.p4suhag.com



# Managing with GitOps

Data Scientists can specify YAML specifications to create Dagger. This allows them to version control, use GitOps for managing feature engineering pipelines.

#### • • •

kind: daggerJob description: demand (unique customers) for s2id level 11 entity: gojek flink name: p-godata-id-ds-marketplace privacy: public sink type: kafka configuration: FLINK PARALLELISM: 1 FLINK SQL QUERY: |-SELECT \* FROM `data streams 0` FLINK WATERMARK DELAY MS: '4000' FLINK WATERMARK INTERVAL MS: '60000' PROCESSOR POSTPROCESSOR ENABLE: false SINK KAFKA TOPIC: booking-S2L11-EW1m-agg SINK TYPE: kafka

#### **CODE EDITOR**



## **SQL first**

Dagger is built keeping SQL first philosophy in mind.

SQL is not "Turing complete" in a key way: it always terminates. Which makes it **less likely** to monopolize all the compute power in a data center.

#### •••

#### SELECT

api\_name AS api\_name, api\_uri AS api\_uri, api\_method AS api\_method, Cast(http\_status\_code AS BIGINT) as http\_status\_code, count(1) as request\_count, Tumble\_end(rowtime, INTERVAL '60' second) AS event\_timestamp FROM `api\_logs` GROUP BY api\_name, api\_uri, http\_status\_code, api\_method,

Tumble (rowtime, INTERVAL '60' second)



### UDFs

Allows custom business logic with User Defined Functions written in Python or Java for advanced use cases.

#### Example:

GeoHash(Double latitude, Double
longitude, int level)

Returns a geohash for a given level and lat-long for the given WGS84 point.

#### • • •

#### SELECT

data1\_location.longitude AS long,

data1\_location.latitude AS lat,

#### $\odot$

data1\_location.longitude,

data1\_location.latitude,

6

GeoHash(

AS geohashPickup,

TUMBLE\_END(rowtime, INTERVAL '60' SECOND) AS window\_timestamp

#### FROM

data stream

GROUP BY

TUMBLE (rowtime, INTERVAL '60' SECOND),

data1 location.longitude,

data1\_location.latitude

#### **CODE EDITOR**



### **Data masking**

Enables encryption on a set of fields as configured.

Used in data forwarding to clone production data to integration and local environments for model training with encryption on sensitive data fields.

• • •	CODE EDITOR	
SELECT		
<pre>event_timestamp,</pre>		
test_data		
FROM		
data_stream		
************** Post processor config: ************************************		
{		
"internal_source": [{		
"output_field": "*",		
"value": "*",		
"type": "sql"}],		
"transformers": [{		
"Transformation_class": "io.odpf.dagger.functions.	transformers.HashTransfor	
"transformation_arguments": {		
"maskColumns": ["test_data.data1"]		
}}1		



### Hybrid data

### source

- Consume data from multiple sources.
- Auto switch from batch source to stream source.





# Backfill with hybrid source

Unified processing across batch and stream data sources.

Backfill historic data for model training.

Join data across multiple streams for complex use cases.

#### • • •

"INPUT SCHEMA TABLE": "booking log stream",

"SOURCE\_KAFKA\_TOPIC\_NAMES": "go-food-booking-log|go-ride-booking-log",

"INPUT\_SCHEMA\_PROTO\_CLASS": "com.esb.proto.BookingLogMessage",

"INPUT\_SCHEMA\_EVENT\_TIMESTAMP\_FIELD\_INDEX": "5",

"SOURCE\_KAFKA\_CONSUMER\_CONFIG\_BOOTSTRAP\_SERVERS": "localhost:9092, localhost:7336, localhost:6262",

"SOURCE\_KAFKA\_CONSUMER\_CONFIG\_GROUP\_ID": "dummy-consumer-group", "SOURCE\_PARQUET\_UNREADABLE\_FILE\_BEHAVIOUR": "FAIL\_WITH\_EXCEPTION", "SOURCE\_PARQUET\_FILE\_PATHS": [

"gs://p-godata-id-mainstream-bedrock/GO\_FOOD-booking-log/dt=2022-01-23/",
"gs://p-godata-id-mainstream-bedrock/GO\_RIDE-booking-log/dt=2021-01-23/"

"SOURCE\_PARQUET\_READ\_ORDER\_STRATEGY": "EARLIEST\_TIME\_URL\_FIRST",

"SOURCE\_PARQUET\_SCHEMA\_MATCH\_STRATEGY": "BACKWARD\_COMPATIBLE\_SCHEMA",

"SOURCE\_DETAILS": [

 $\bigcirc$ 

{"SOURCE\_TYPE": "BOUNDED", "SOURCE\_NAME": "PARQUET\_SOURCE"},

{"SOURCE\_TYPE": "UNBOUNDED","SOURCE\_NAME": "KAFKA\_SOURCE"]

CODE EDITOR



## Stream enrichment

Use external data sources to fetch additional information about each event in processing.

 $\bigcirc$ 

Allows to use any external Data sources, Service endpoints, Object stores, Cache for enriching your stream.

Learn more: https://odpf.github.io/dagger/docs/usecase/stream\_enrich ment

### **CODE EDITOR** [{"internal source": [{ "output field": "booking log", "type": "sql", "value": "\*" }]}, {"external source":{ "es":[{ "host":"127.0.0.1", "port":"9200", "endpoint pattern":"/customers/customer/%s", "endpoint variables":"customer id", "stream timeout":"5000", "connect timeout":"5000", "capacity":"30", "output mapping":{ "customer profile":{"path":"\$. source"} }}]





### **Stream Inference**

Stream inference allows real time predictions where multiple users might want to consume predictions of a single model.

Prediction logs can be stored for quality and monitoring purposes





### **Stream** Inference

Filtering and custom field mapping during stream inference.

#### Model Endpoint \* Model Type \* http://sklearn-3.sample.models.id.d.gods.golabs.io/v1/predict $\otimes$ $\checkmark$ Arbitrary Request Payload Mapping \* SOURCE FIELD receipt\_url AS <alias> SOURCE FIELD service\_type AS <alias> "receipt\_url": "\${receipt\_url}", "service\_type": "\${service\_type}", SOURCE FIELD status AS <alias> "status": "\${status}", "order\_number": "\${order\_number}", SOURCE FIELD order\_number AS <alias> "service\_area\_id": "\${service\_area\_id}", "driver\_id": "\${driver\_id}" SOURCE FIELD service\_area\_id AS <alias> SOURCE FIELD driver\_id AS <alias> SOURCE FIELD <field> Specify the filter for queries from Dagger, if applicable Resulting Type Mapping \*

SOURCE FIELD service\_type AS SINK FIELD service\_type SOURCE FIELD order\_number AS SINK FIELD order\_number SOURCE FIELD receipt\_url AS SINK FIELD receipt\_url FUNCTION CURRENT\_TIMESTAMP AS SINK FIELD event\_timestamp SOURCE FIELD driver\_id AS SINK FIELD driver\_id SOURCE FIELD service\_area\_id AS SINK FIELD service\_area\_id SOURCE FIELD status AS SINK FIELD status JSON PATH \$.prediction\_score AS SINK FIELD prediction\_score

✓ Filter WHERE

JSON PATH <\$.predictions[0]> AS SINK FIELD <field>



### **Dagger adoption at Gojek**

300+

Dagger jobs for feature engineering.

50+

Data scientists creating Dagger jobs.



Data processed each day.





### Dagger is part of Open Data Ops Foundation

ODPF is a modern DataOps platform that empowers organizations to discover, transform, analyse and secure data faster and efficiently.



### An experience-first approach



#### Discover

Search through large amount of data across the organization.

#### Understand

\*\*\*

Get contextual knowledge with lineage, quality and other aspects of data.

#### Operate

\*\*\*

Process, wrangle, transform or analyse data as per your needs.



#### Apply

Drive business value with data, models and insights.



# A fully-integrated suite of data products

A fully-integrated suite of open-source products that are required to build an end-to-end data platform for all your needs from ingestion to insights. It also provides products for data management plane ranging across infrastructure orchestration, observability, security, access control and data catalog.



#### DATA LIFECYCLE



Raccoon Data ingestion



Dagger Stream processsing



Firehose Data loading



 $\bigcirc$ 

Guardian Data access control

Metadata collection

DATA MANAGEMENT

Compass

Meteor

Data catalog



Stencil Schema registry

https://github.com/odpf



Optimus Data transformation



**Enigma** Operational analytics



Shield Identity management



Predator Data observability

#### **DATA OPERATIONS**



<u>\*</u>



Siren Site reliability



### **Powering data platform for** large-scale data teams

### **⊘gojek ı|ı** mıdtrans

mapan 📚

MOKO



### **Active community**

### 200+

Contributors.

With 80% growth year over year

2000+

Commits last year.

year

With 39% growth year over

Community members.

Across Github and Slack

1000 +









2









### **Get involved**

### ()

Explore and contribute to ODPF data platform on Github.

https://github.com/odpf



Join the community on Slack and talk to maintainers.

https://bit.ly/2RzPbtn







### Thank you

Email: <u>suhag.ravi@gmail.com</u>

Twitter: ravi\_suhag

Github: ravisuhag

Website: www.ravisuhag.com



0

 $\diamond$ 

0

0

0

0