# Feature Store: the missing data layer in ML pipelines?[1]

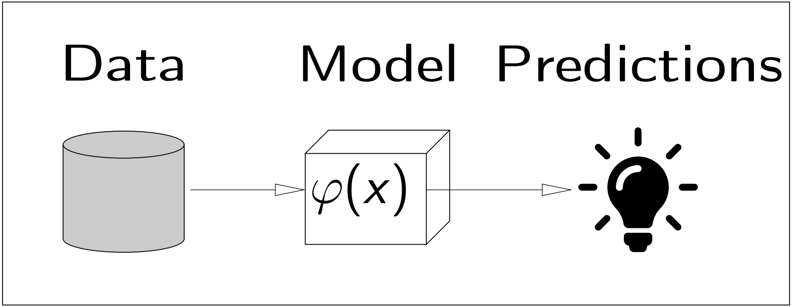
*HopsML Stockholm*

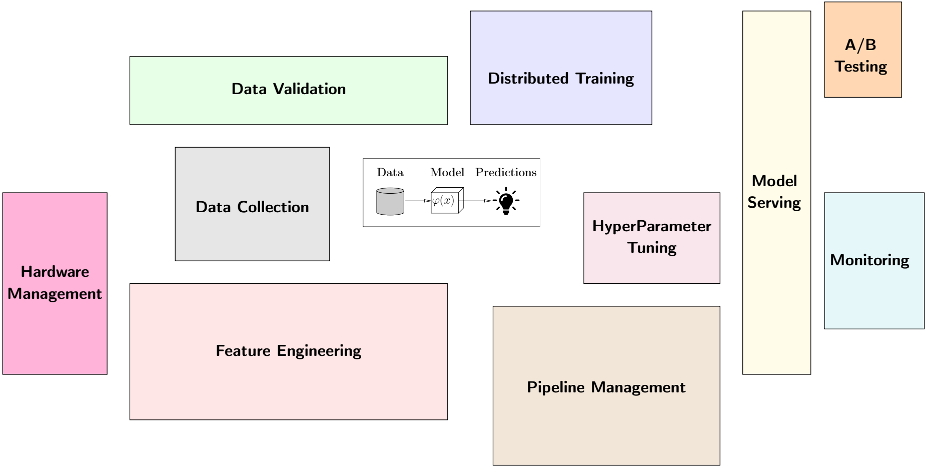
Kim Hammar

*kim@logicalclocks.com*

January 29, 2019

[1] Kim Hammar and Jim Dowling. *Feature Store: the missing data layer in ML pipelines?* https://www.logicalclocks.com/feature-store/. 2018.


Data
Model
Predictions
$\varphi(x)$



Data Validation

Distributed Training

A/B Testing

Data Collection

HyperParameter Tuning

Model Serving

Hardware Management

Monitoring

Feature Engineering

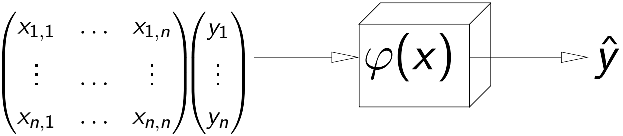
Pipeline Management

[2]

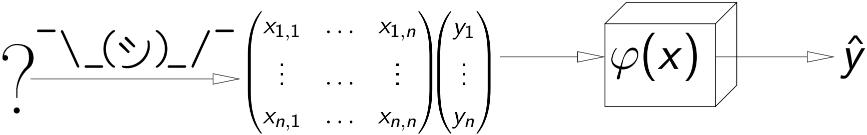
[2] Image inspired from Sculley et al. (Google) *Hidden Technical Debt in Machine Learning Systems*

# Outline

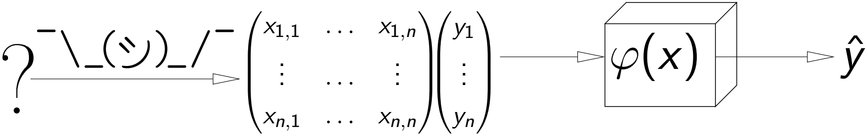
1. **What** is a Feature Store
2. **Why** You Need a Feature Store
3. **How** to Build a Feature Store (Hopsworks Feature Store)
4. Demo



$$\begin{pmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \dots & \vdots \\ x_{n,1} & \dots & x_{n,n} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \longrightarrow \boxed{\varphi(x)} \longrightarrow \hat{y}$$


?
¯_(ツ)_/¯
$\begin{pmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \dots & \vdots \\ x_{n,1} & \dots & x_{n,n} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$
$\varphi(x)$
$\hat{y}$

[3] Jeremy Hermann and Mike Del Balso. *Scaling Machine Learning at Uber with Michelangelo*. https://eng.uber.com/scaling-michelangelo/. 2018.

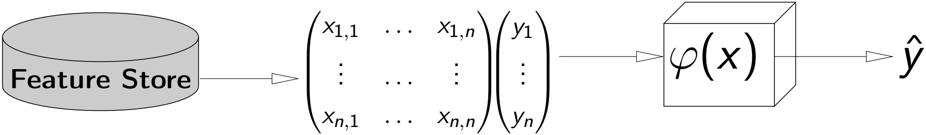

?
¯_(ツ)_/¯
$\begin{pmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \dots & \vdots \\ x_{n,1} & \dots & x_{n,n} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$
$\varphi(x)$
$\hat{y}$

*"Data is the hardest part of ML and the most important piece to get right.*

*Modelers spend most of their time selecting and transforming features at training time and then building the pipelines to deliver those features to production models."*

*- Uber*[3]

[3] Jeremy Hermann and Mike Del Balso. *Scaling Machine Learning at Uber with Michelangelo.* https://eng.uber.com/scaling-michelangelo/. 2018.



Feature Store

$$\begin{pmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \dots & \vdots \\ x_{n,1} & \dots & x_{n,n} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \longrightarrow \varphi(x) \longrightarrow \hat{y}$$

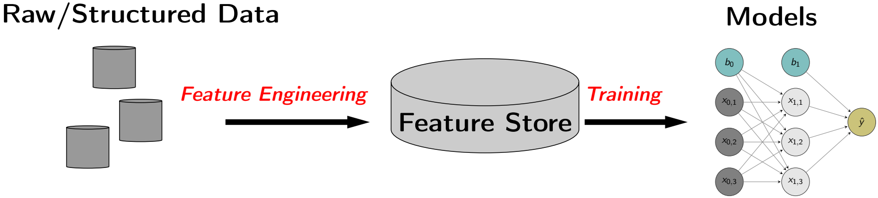
*"Data is the hardest part of ML and the most important piece to get right.*

*Modelers spend most of their time selecting and transforming features at training time and then building the pipelines to deliver those features to production models."*

*- Uber*[4]

[4] Jeremy Hermann and Mike Del Balso. *Scaling Machine Learning at Uber with Michelangelo*. https://eng.uber.com/scaling-michelangelo/. 2018.

# Solution: Disentangle ML Pipelines with a Feature Store



Raw/Structured Data

*Feature Engineering*

Feature Store

*Training*

Models

- A feature store is a central vault for storing documented, curated, and access-controlled features.
- The feature store is the interface between data engineering and data model development

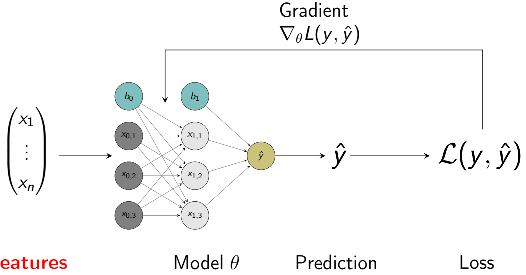
# What is a Feature?

*A feature is a measurable property of some data-sample*

A feature could be..

- An aggregate value (min, max, mean, sum)
- A raw value (a pixel, a word from a piece of text)
- A value from a database table (the age of a customer)
- A derived representation: e.g an embedding or a cluster

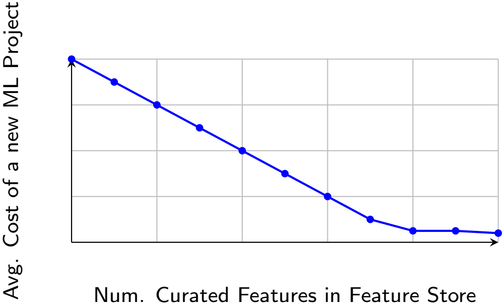
**Features are the fuel for AI systems:**



# Motivation for the Feature Store

**The feature store enables:**

- Reusability of features between models and teams
- Automatic backfilling of features
- Automatic feature documentation and analysis
- Feature versioning
- Standardized access of features between training and serving
- Feature discovery



# Reusing Features
# **Without a Feature Store** is Complex

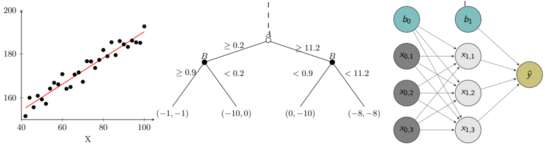
**Data Sources** Dataset 1 Dataset 2 · · · Dataset $n$

**Siloed Feature Sets**
Without a feature store it is typical to have feature sets stored in isolation from each other.

$$\begin{pmatrix} w_{1,1} & \dots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{n,1} & \dots & w_{n,n} \end{pmatrix} \quad \begin{pmatrix} w_{1,1} & \dots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{n,1} & \dots & w_{n,n} \end{pmatrix} \quad \begin{pmatrix} w_{1,1} & \dots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{n,1} & \dots & w_{n,n} \end{pmatrix}$$

Features Features Features

**Models**
Models are trained using sets of features. Without a feature store each model typically defines its own feature definitions, without feature sharing across models.

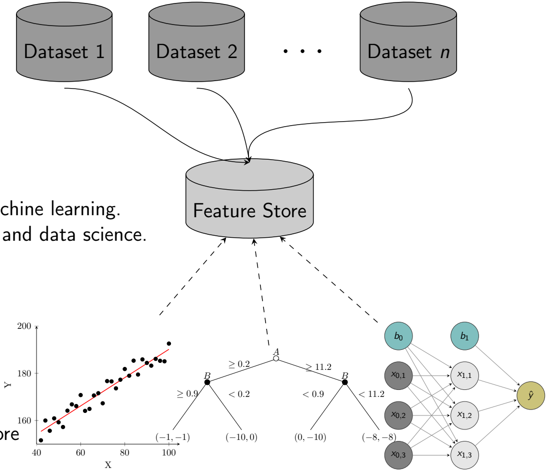


# Reusing Features
# **With a Feature Store** is Simple

**Data Sources**

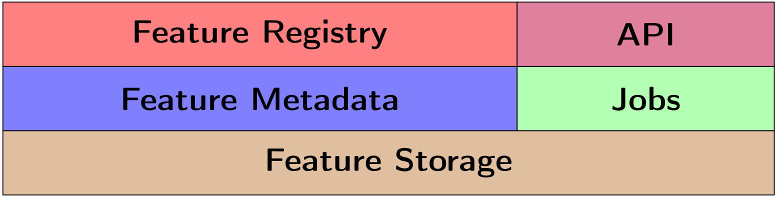
**Feature Store**
A data management platform for machine learning.
The interface between data engineering and data science.

**Models**
Models are trained using sets of features.
The features are fetched from the feature store
and can overlap between models.



# The Components of a Feature Store

- **The Storage Layer:** For storing feature data in the feature store
- **The Metadata Layer:** For storing feature metadata (versioning, feature analysis, documentation, jobs)
- **The Feature Engineering Jobs:** For computing features
- **The Feature Registry:** A user interface to share and discover features
- **The Feature Store API:** For writing/reading to/from the feature store



Feature Registry

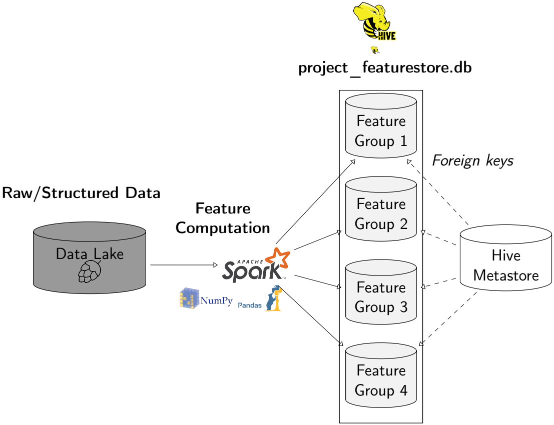
API

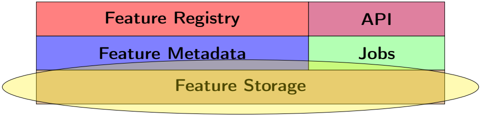
Feature Metadata

Jobs

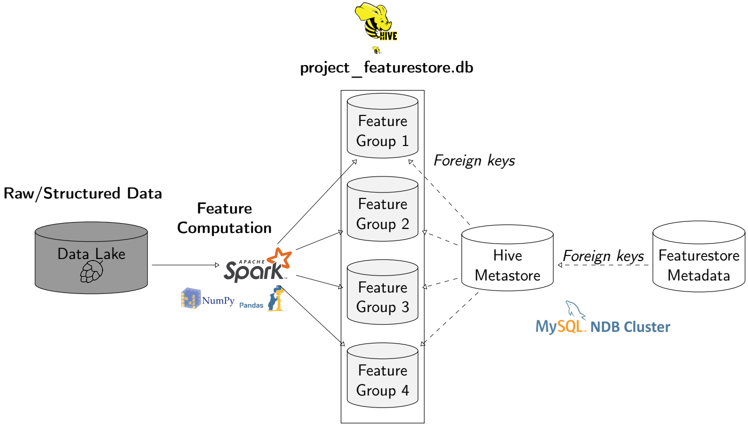
Feature Storage

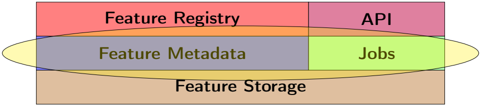
# Feature Storage



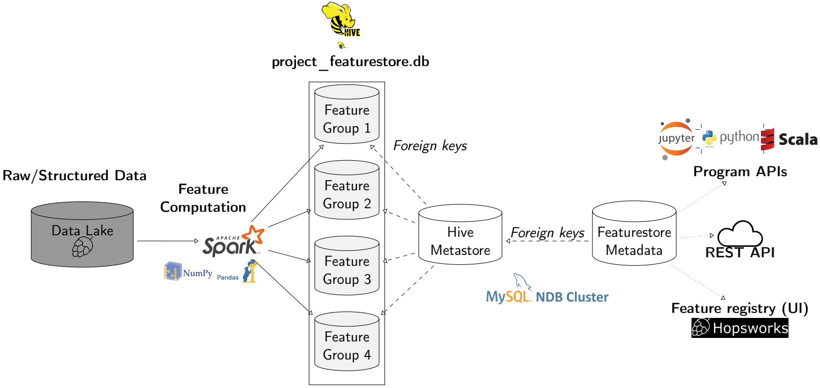


# Feature Metadata





# Feature Registry and API



project_featurestore.db

Raw/Structured Data

Data Lake

Feature Computation

Feature Group 1

Feature Group 2

Feature Group 3

Feature Group 4

*Foreign keys*

Hive Metastore

*Foreign keys*

Featurestore Metadata

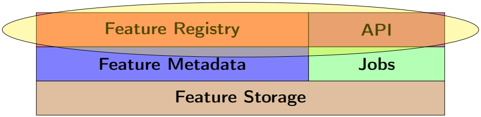
MySQL NDB Cluster

Program APIs

REST API

Feature registry (UI)

Hopsworks



Feature Registry

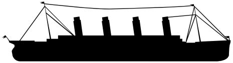
API

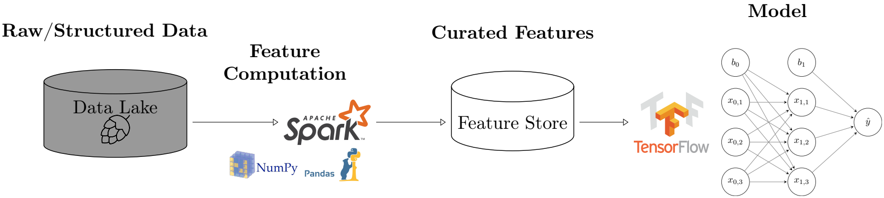
Feature Metadata

Jobs

Feature Storage

# Demo-Setting





Raw/Structured Data

Data Lake

Feature Computation

Curated Features

Feature Store

Model

# Summary

- Machine learning comes with a high technical cost
- Machine learning pipelines needs proper data management
- A **feature store** is a place to store curated and documented features
- The feature store serves as an interface between feature engineering and model development, it can help disentangle complex ML pipelines
- *Hopsworks*[5] provides the world's first open-source feature store

@hopshadoop
www.hops.io

@logicalclocks
www.logicalclocks.com

We are open source:
https://github.com/logicalclocks/hopsworks
https://github.com/hopshadoop/hops

6

[5] Jim Dowling. *Introducing Hopsworks*. https://www.logicalclocks.com/introducing-hopsworks/. 2018.

[6] Thanks to Logical Clocks Team: Jim Dowling, Seif Haridi, Theo Kakantousis, Fabio Buso, Gautier Berthou, Ermias Gebremeskel, Mahmoud Ismail, Salman Niazi, Antonios Kouzoupis, Robin Andersson, and Alex Ormenisan

## References

LOGICAL CLOCKS

- Hopsworks' feature store[7] (**the only open-source one!**)
- Uber's feature store[8]
- Airbnb's feature store[9]
- Comcast's feature store[10]
- GO-JEK's feature store[11]
- HopsML[12]
- Hopsworks[13]

[7] Kim Hammar and Jim Dowling. *Feature Store: the missing data layer in ML pipelines?* https://www.logicalclocks.com/feature-store/. 2018.

[8] Li Erran Li et al. "Scaling Machine Learning as a Service". In: *Proceedings of The 3rd International Conference on Predictive Applications and APIs*. Ed. by Claire Hardgrove et al. Vol. 67. Proceedings of Machine Learning Research. Microsoft NERD, Boston, USA: PMLR, 2017, pp. 14–29. URL: http://proceedings.mlr.press/v67/li17a.html.

[9] Nikhil Simha and Varant Zanoyan. *Zipline: Airbnb's Machine Learning Data Management Platform*. https://databricks.com/session/zipline-airbnbs-machine-learning-data-management-platform. 2018.

[10] Nabeel Sarwar. *Operationalizing Machine Learning—Managing Provenance from Raw Data to Predictions*. https://databricks.com/session/operationalizing-machine-learning-managing-provenance-from-raw-data-to-predictions. 2018.

[11] Willem Pienaar. *Building a Feature Platform to Scale Machine Learning | DataEngConf BCN '18*. https://www.youtube.com/watch?v=0iCXY6VnpCc. 2018.

[12] Logical Clocks AB. *HopsML: Python-First ML Pipelines*. https://hops.readthedocs.io/en/latest/hopsml/hopsML.html. 2018.

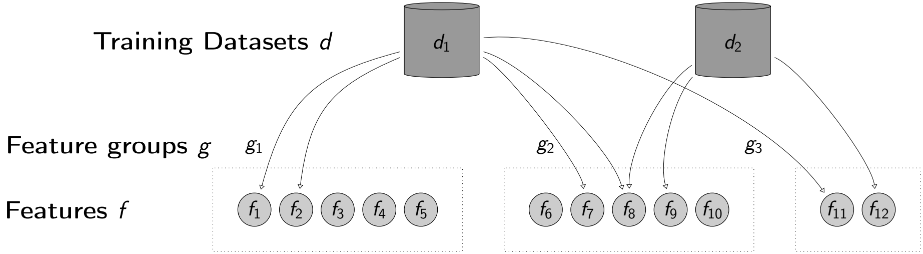
[13] Jim Dowling. *Introducing Hopsworks*. https://www.logicalclocks.com/introducing-hopsworks/. 2018.

# Backup Slides

# Modeling Data in the Feature Store

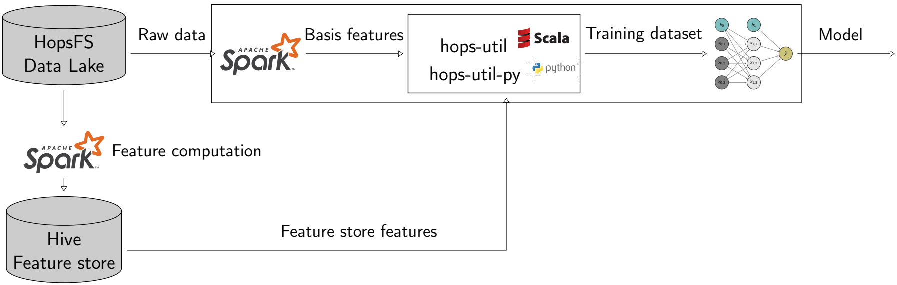
- A **feature group** is a logical grouping of **features**
  - Typically from the same input dataset and computed with the same job

- A **training dataset** is a set of features suitable for a prediction task
  - Features in a training dataset are often from several feature groups
  - E.g features on customers, features on user activities, etc.



# Training Pipeline in HopsML

1. Create job/notebook to **compute features** and publish to the feature store
2. Create job/notebook to read features/labels and **save to a training dataset**
3. **Read the training dataset into your model** for training



# Hopsworks Feature Store API

## Reading from the Feature Store:

```
from hops import featurestore
features_df = featurestore.get_features([
                                         "average_attendance",
                                         "average_player_age"
                                        ])
```

## Writing to the Feature Store:

```
from hops import featurestore
raw_data = spark.read.parquet(filename)
pol_features = raw_data.map(lambda x: x^2)
featurestore.insert_into_featuregroup(pol_features, "pol_featuregroup")
```