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

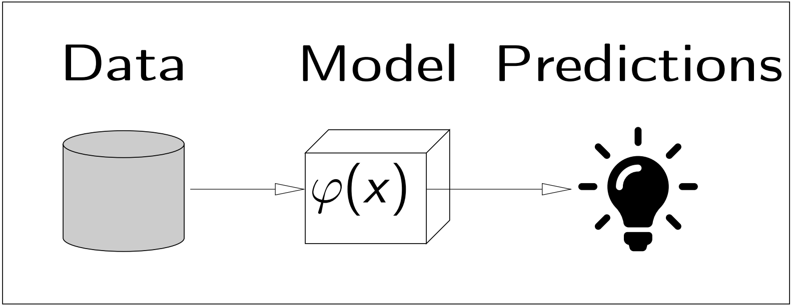
*Spotify ML Guild Fika*

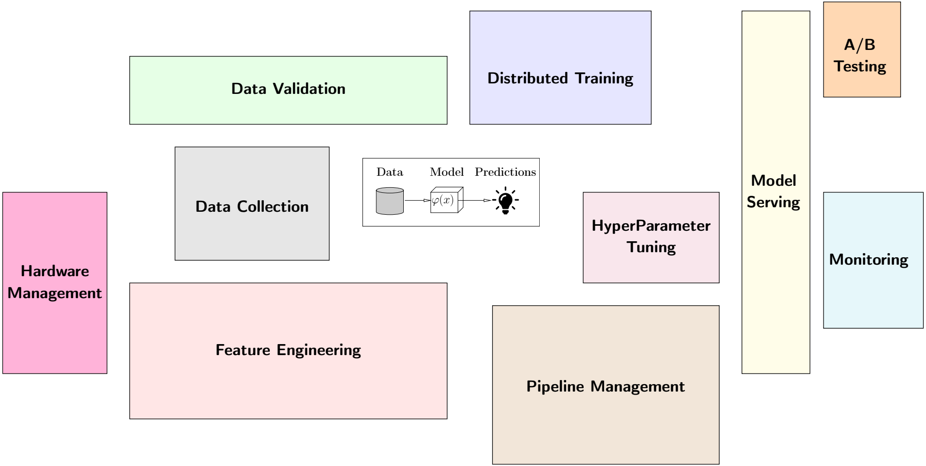
Kim Hammar

*kim@logicalclocks.com*

February 26, 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
$\varphi(x)$



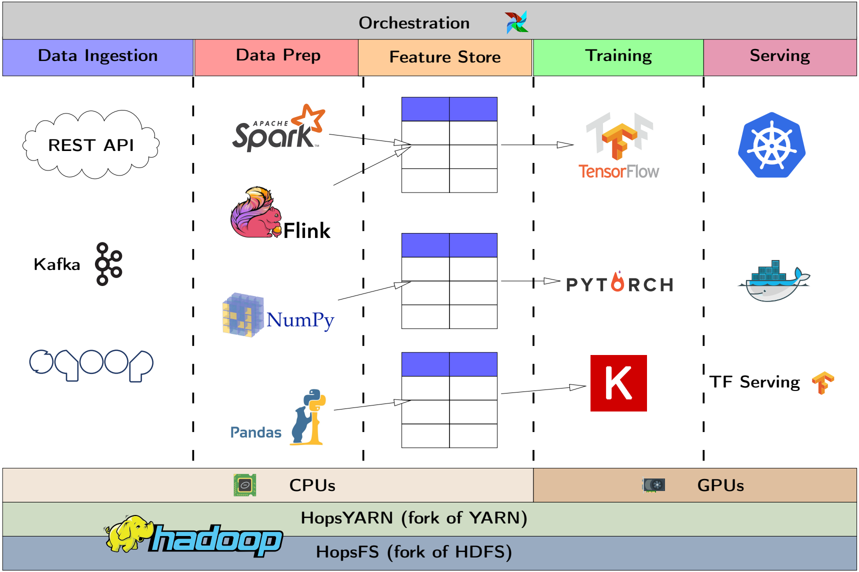

2

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

# Outline

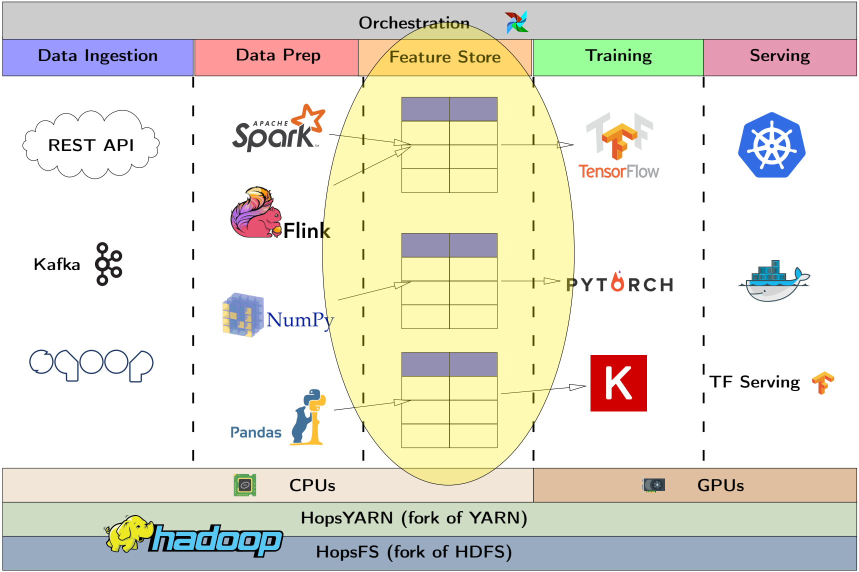
1. **Hopsworks**: Quick background of the platform
2. **What** is a Feature Store
3. **Why** You Need a Feature Store, Things to Consider:
    - How to encourage feature reusage?
    - How to store large-scale datasets for deep learning?
    - How to serve features for inference?
4. **How** to Build a Feature Store (Hopsworks Feature Store Case Study)
5. **Demo**

# HopsML



Orchestration
Data Ingestion
Data Prep
Feature Store
Training
Serving
REST API
Kafka
Spark
Flink
NumPy
Pandas
TensorFlow
PYTORCH
TF Serving
CPUs
GPUs
HopsYARN (fork of YARN)
HopsFS (fork of HDFS)
hadoop

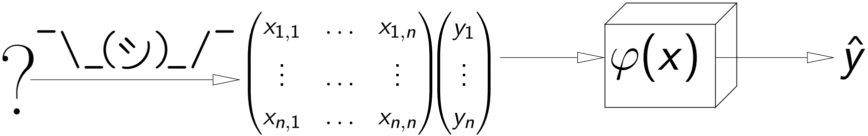
# HopsML


Orchestration
Data Ingestion
Data Prep
Feature Store
Training
Serving
REST API
Kafka
APACHE Spark
Flink
NumPy
Pandas
TensorFlow
PYTORCH
TF Serving
CPUs
GPUs
HopsYARN (fork of YARN)
hadoop
HopsFS (fork of HDFS)

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

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

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

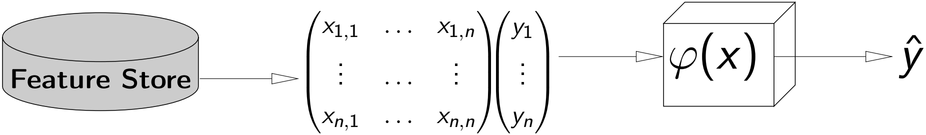



*"Data is the hardest part of ML and the most important piece to get right.*

*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.




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

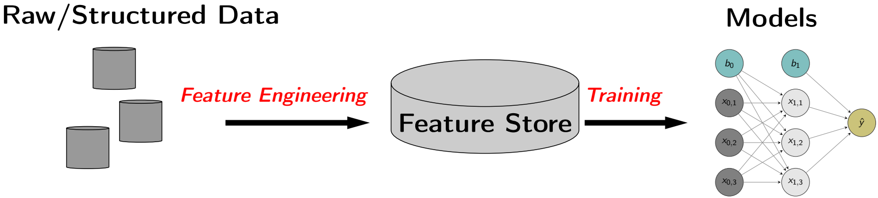
*Modelers spend most of their time selecting and transforming features at training time and then building the pipelines to deliver those features to production models."*

*- Uber*[4]

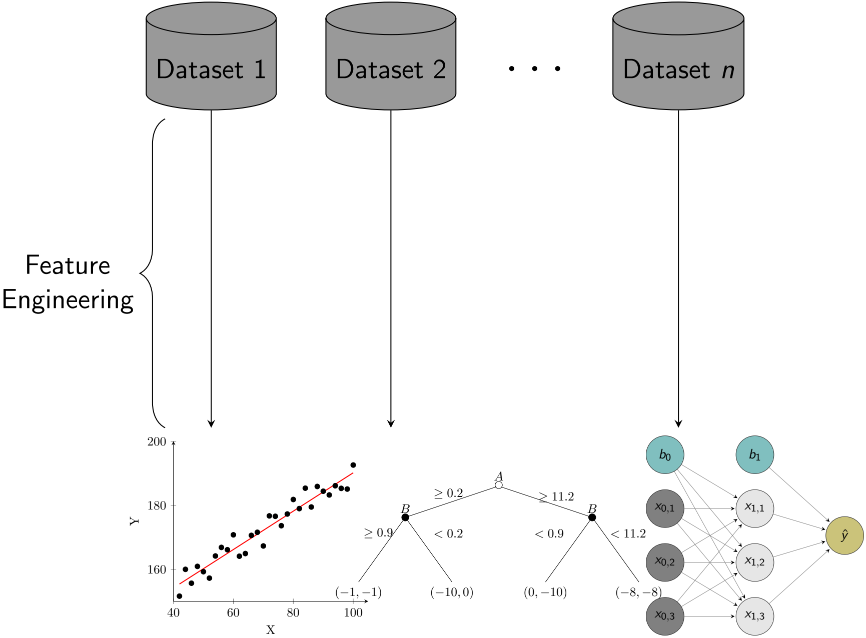
---

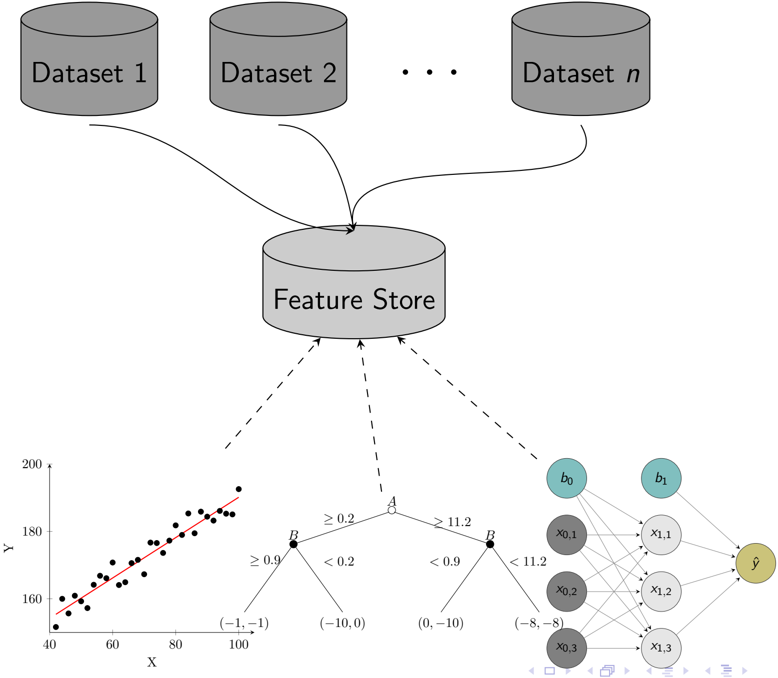
[4] Jeremy Hermann and Mike Del Balso. *Scaling Machine Learning at Uber with Michelangelo*. https://eng.uber.com/scaling-michelangelo/. 2018.

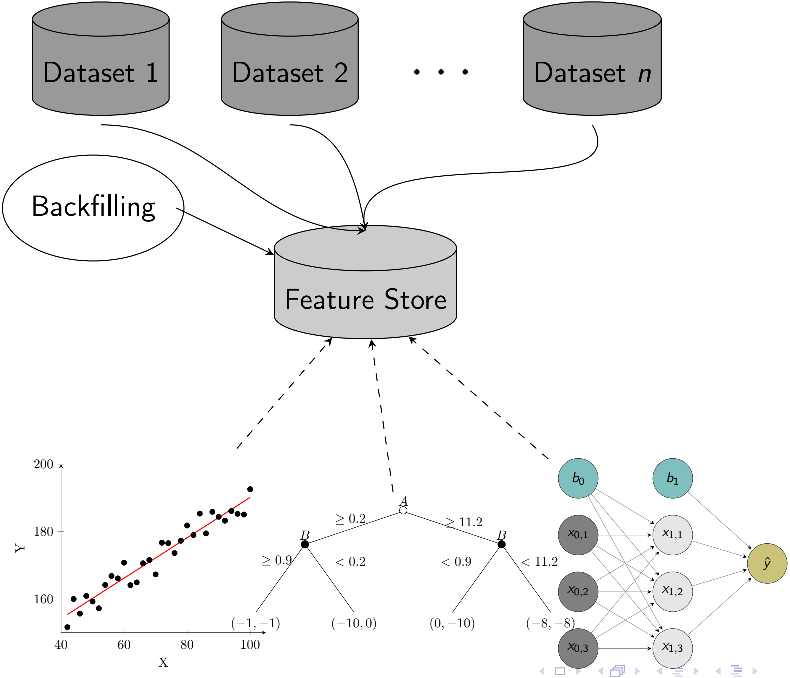
# Disentangle ML Pipelines with a Feature Store



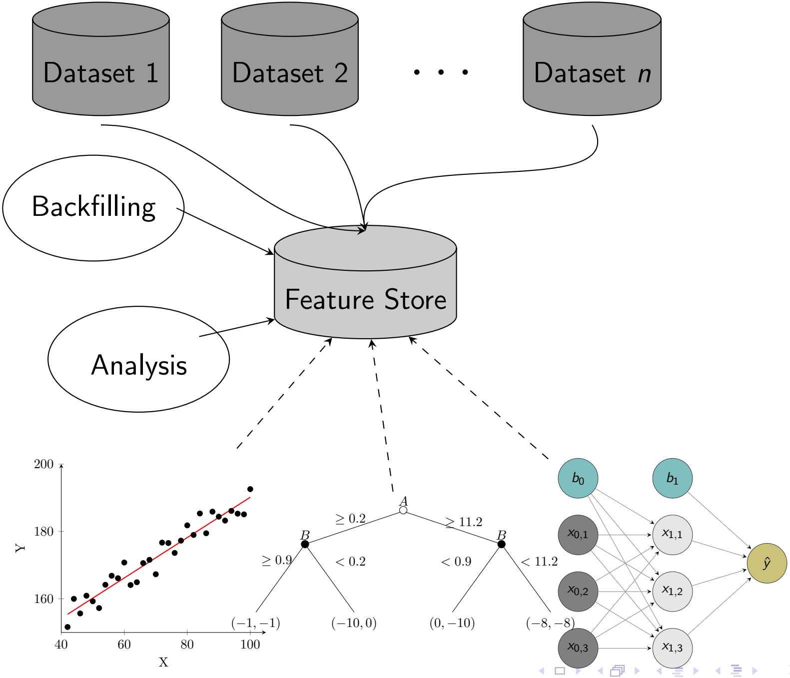

- A feature store is a central vault for storing documented, curated, and access-controlled features.
- The feature store is the interface between data engineering and data model development

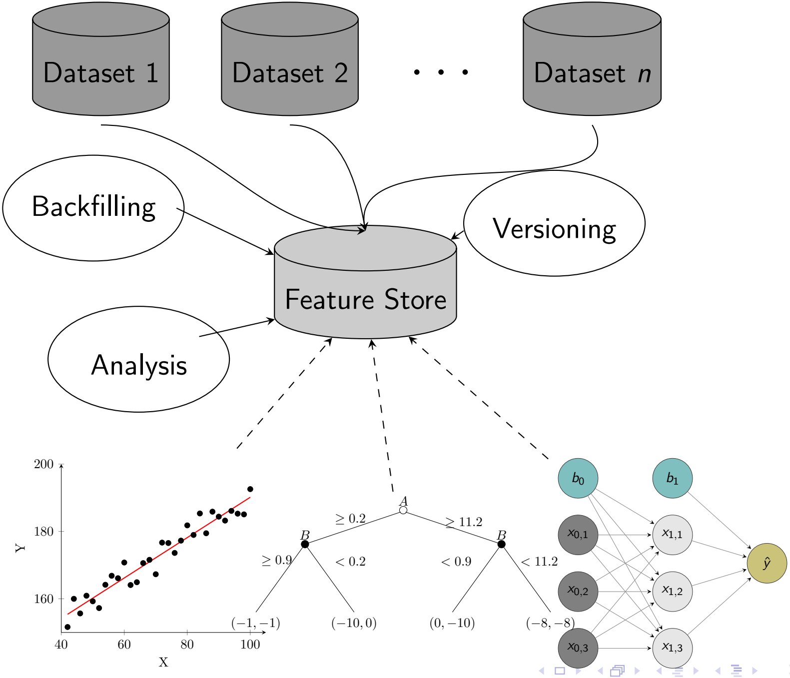

Dataset 1
Dataset 2
Dataset n
Feature Engineering
200
180
160
Y
40
60
80
100
X
A
B
B
≥ 0.2
≥ 11.2
≥ 0.9
< 0.2
< 0.9
< 11.2
(−1, −1)
(−10, 0)
(0, −10)
(−8, −8)
b0
b1
x0,1
x0,2
x0,3
x1,1
x1,2
x1,3
ŷ

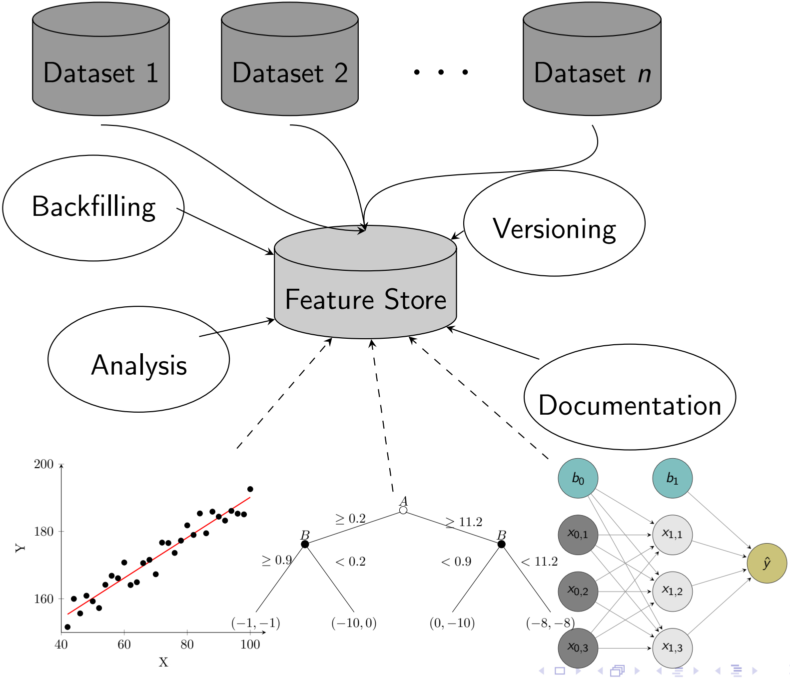

Dataset 1
Dataset 2
· · ·
Dataset n
Feature Store
200
180
160
Y
40
60
80
100
X
A
≥ 0.2
≥ 11.2
B
B
≥ 0.9
< 0.2
< 0.9
< 11.2
(−1, −1)
(−10, 0)
(0, −10)
(−8, −8)
b0
b1
x0,1
x0,2
x0,3
x1,1
x1,2
x1,3
ŷ



Dataset 1
Dataset 2
. . .
Dataset n
Backfilling
Feature Store
200
180
160
Y
40
60
80
100
X
A
≥ 0.2
≥ 11.2
B
B
≥ 0.9
< 0.2
< 0.9
< 11.2
(−1, −1)
(−10, 0)
(0, −10)
(−8, −8)
$b_0$
$b_1$
$x_{0,1}$
$x_{0,2}$
$x_{0,3}$
$x_{1,1}$
$x_{1,2}$
$x_{1,3}$
$\hat{y}$


Dataset 1
Dataset 2
· · ·
Dataset n
Backfilling
Analysis
Feature Store
200
180
160
Y
40
60
80
100
X
A
≥ 0.2
≥ 11.2
B
B
≥ 0.9
< 0.2
< 0.9
< 11.2
(−1, −1)
(−10, 0)
(0, −10)
(−8, −8)
b0
b1
x0,1
x0,2
x0,3
x1,1
x1,2
x1,3
ŷ


Dataset 1
Dataset 2
· · ·
Dataset $n$
Backfilling
Versioning
Feature Store
Analysis
200
180
160
Y
40
60
80
100
X
$A$
$\geq 0.2$
$\geq 11.2$
$B$
$B$
$\geq 0.9$
$< 0.2$
$< 0.9$
$< 11.2$
$(-1, -1)$
$(-10, 0)$
$(0, -10)$
$(-8, -8)$
$b_0$
$b_1$
$x_{0,1}$
$x_{0,2}$
$x_{0,3}$
$x_{1,1}$
$x_{1,2}$
$x_{1,3}$
$\hat{y}$



Dataset 1
Dataset 2
. . .
Dataset n
Backfilling
Versioning
Feature Store
Analysis
Documentation
200
180
160
Y
40
60
80
100
X
A
B
B
≥ 0.2
≥ 11.2
≥ 0.9
< 0.2
< 0.9
< 11.2
(−1, −1)
(−10, 0)
(0, −10)
(−8, −8)
$b_0$
$b_1$
$x_{0,1}$
$x_{0,2}$
$x_{0,3}$
$x_{1,1}$
$x_{1,2}$
$x_{1,3}$
$\hat{y}$

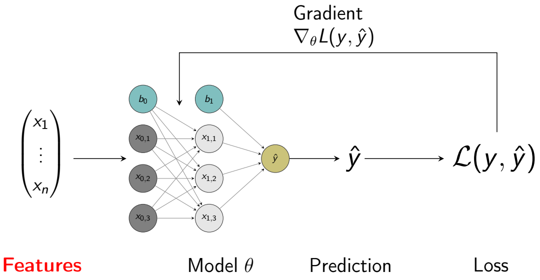
# What is a Feature?

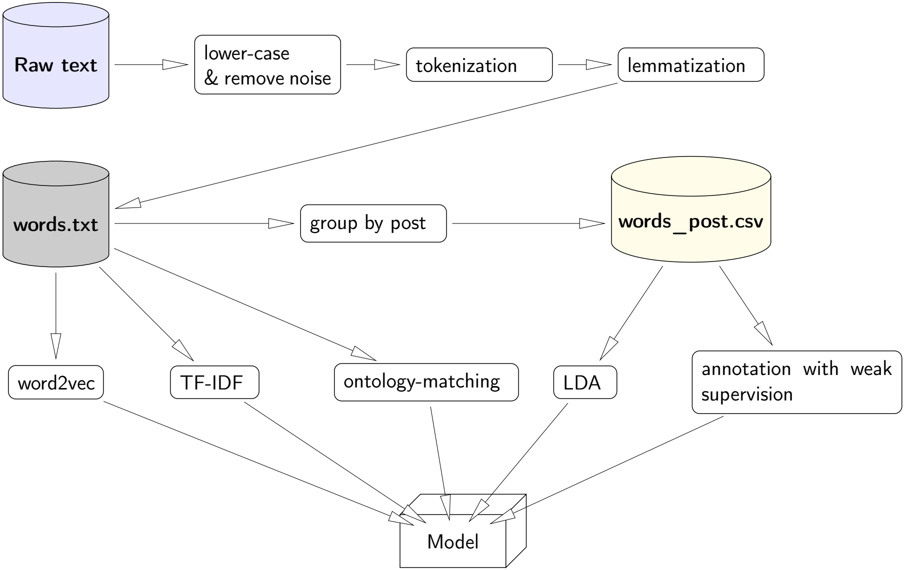
*A feature is a measurable property of some data-sample*

A feature could be..

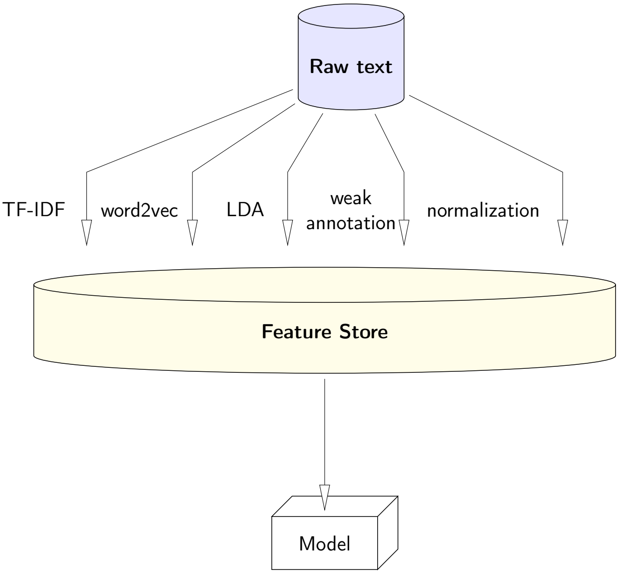
- An aggregate value (min, max, mean, sum)
- A raw value (a pixel, a word from a piece of text)
- A value from a database table (the age of a customer)
- A derived representation: e.g an embedding or a cluster

**<u>Features</u> are the fuel for AI systems:**



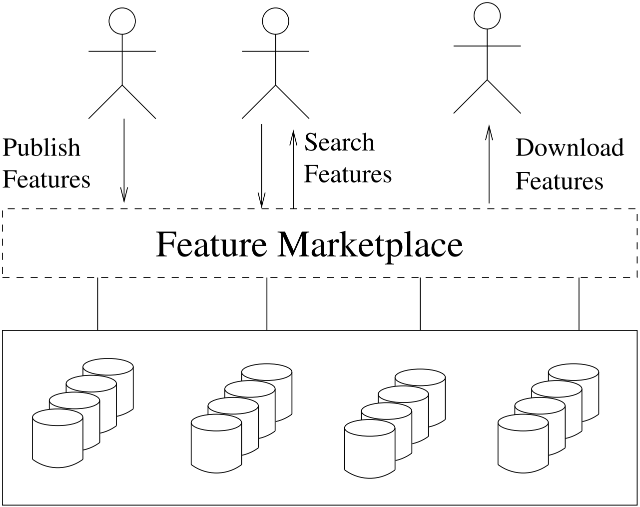


Raw text
lower-case
& remove noise
tokenization
lemmatization
words.txt
group by post
words_post.csv
word2vec
TF-IDF
ontology-matching
LDA
annotation with weak supervision
Model


Raw text
TF-IDF
word2vec
LDA
weak annotation
normalization
Feature Store
Model

# How to Encourage Feature Reusage?

# Feature Marketplace



# Feature Store API Service

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

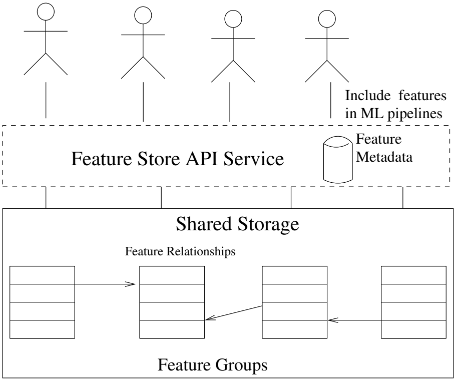
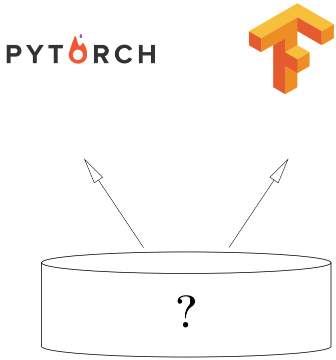



Figure: Feature Store API Service

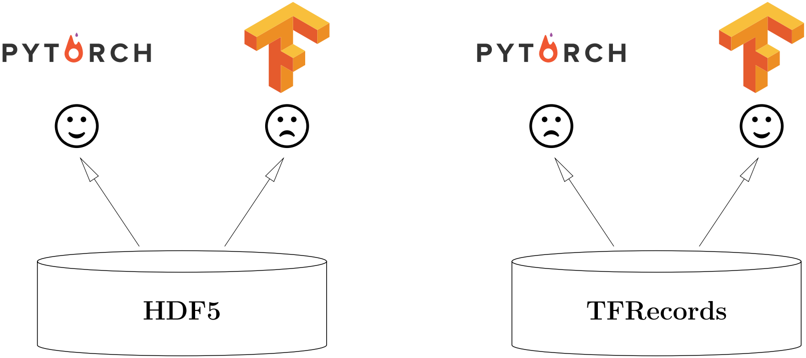
# How to Store Datasets for Deep Learning?

# How to Store Datasets for Deep Learning?

- Should be framework agnostic
- Need to be able to store tensor datasets
- Should support sharding for distributed training
- Advanced features: row-predicate filtering, SQL interface, columnar selection.

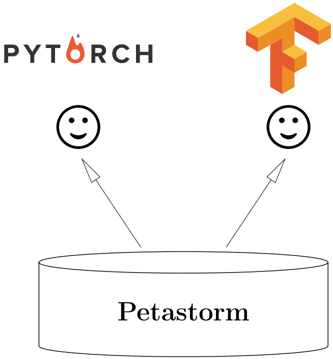


# How to Store Datasets for Deep Learning?



# How to Store Datasets for Deep Learning?

- Petastorm is a dataset format designed for deep learning
- Petastorm stores data as parquet files with extra metadata to handle multi-dimensional tensors
- Petastorm contains readers for the popular machine learning frameworks such as SparkML, Tensorflow, PyTorch



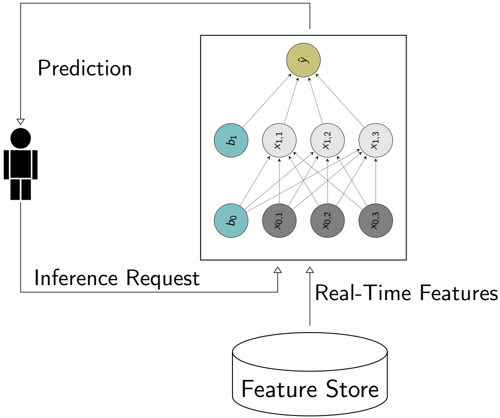

5

[5] Robbie Gruener, Owen Cheng, and Yevgeni Litvin. *Introducing Petastorm: Uber ATG's Data Access Library for*

# How to Serve Features for Inference?

# Delivering Features for Training and Serving is Different

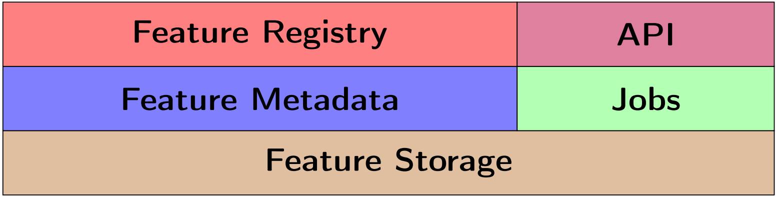
- Serving can require real-time features
- Ideally we want consistency between real-time features and batch features used for training
- Complex engineering problem



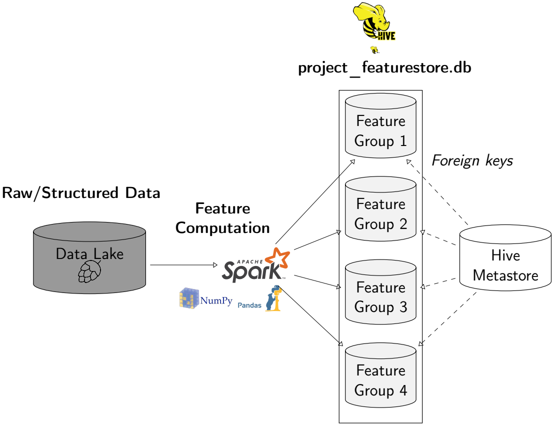
# How to Implement a (batch) Feature Store?

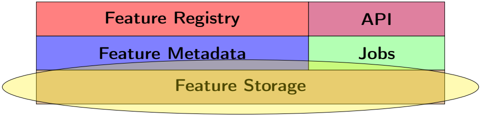
# The Components of a Feature Store

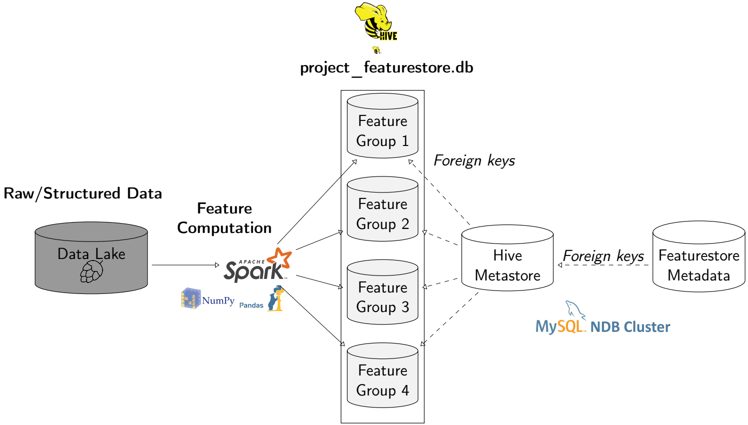
- **The Storage Layer:** For storing feature data in the feature store
- **The Metadata Layer:** For storing feature metadata (versioning, feature analysis, documentation, jobs)
- **The Feature Engineering Jobs:** For computing features
- **The Feature Registry:** A user interface to share and discover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

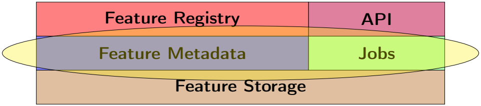


# Feature Storage

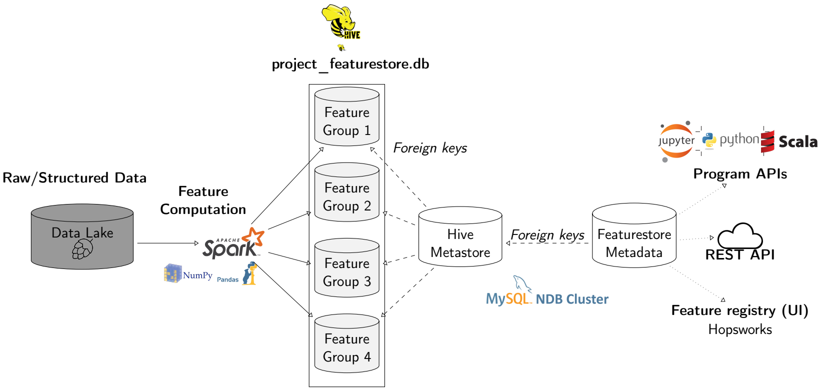


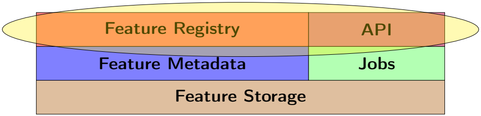




project_featurestore.db
Feature Group 1
Feature Group 2
Feature Group 3
Feature Group 4
Foreign keys
Raw/Structured Data
Data Lake
Feature Computation
APACHE Spark
NumPy
Pandas
Hive Metastore
Foreign keys
Featurestore Metadata
MySQL NDB Cluster

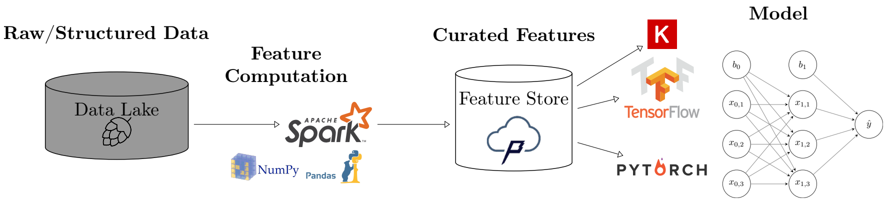

Feature Registry
API
Feature Metadata
Jobs
Feature Storage

# Feature Registry and API



# Demo-Setting



# 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*[6] 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

7

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

[7] 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

- Hopsworks' feature store[8] (**the only open-source one!**)
- Uber's feature store[9]
- Airbnb's feature store[10]
- Comcast's feature store[11]
- GO-JEK's feature store[12]
- HopsML[13]
- Hopsworks[14]


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

[9] 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.

[10] 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.

[11] 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.

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

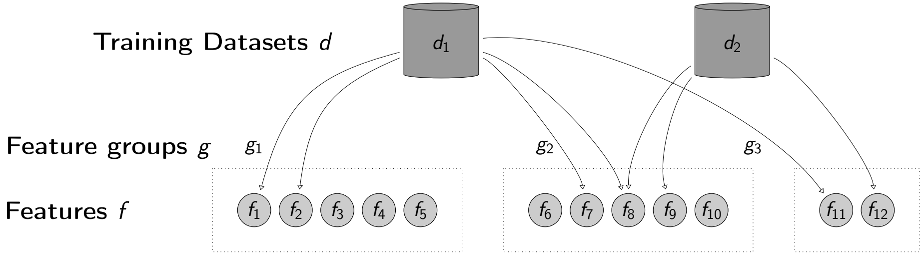
[13] Logical Clocks AB. *HopsML: Python-First ML Pipelines*. https://hops.readthedocs.io/en/latest/hopsml/hopsML.html. 2018.

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

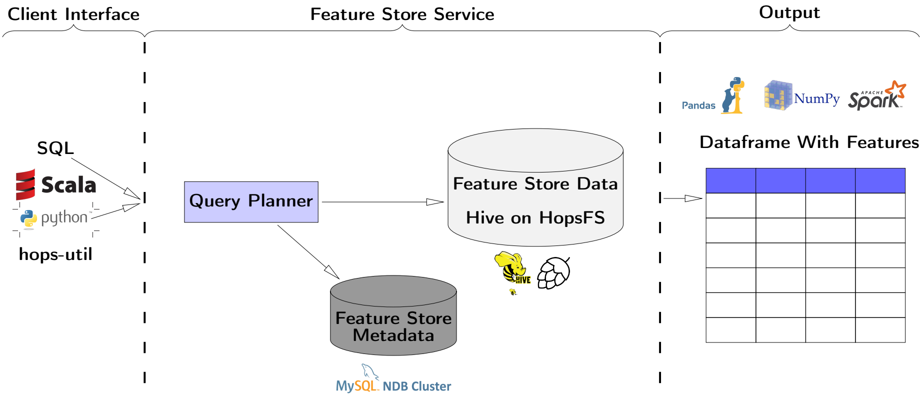
# Backup Slides

# 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.

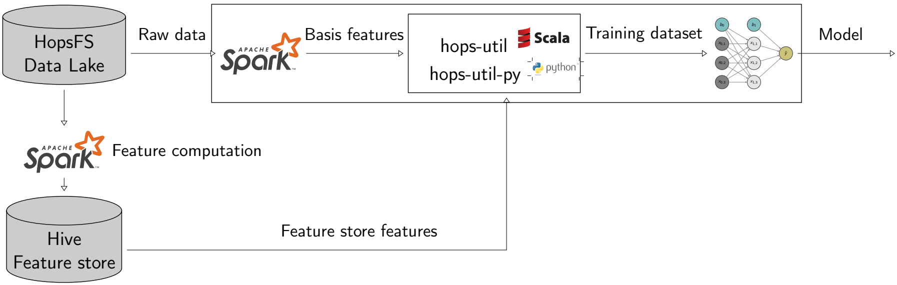


# Hopsworks Feature Store API Service



# 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")
```