DATA+AI SUMMIT 2022

Distributed Machine Learning at Lyft

Anindya Saha

ML Platform Staff Engineer

Han Wang

ML Platform Tech Lead

Senior Staff Engineer

ORGANIZED BY databricks

Agenda

- Introduction
- The pains
- The solutions

Use Cases @ Lyft

Decisions by ML Models

Price Optimization

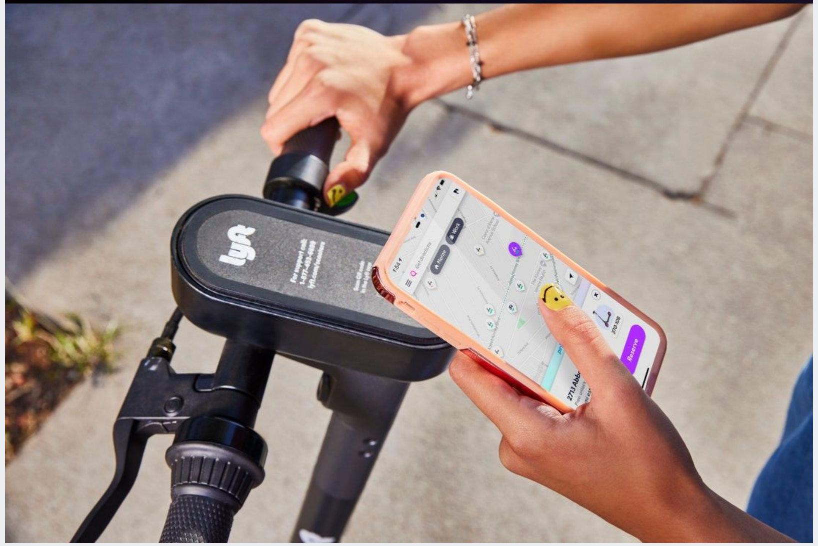
Safety

ETAs

Fraud Detection

Mapping

Incentives



What are distributed ML scenarios?

- Feature engineering on PB level dataset?
- Building ML pipelines using the largest Machines on AWS?
- Using 1000 GPUS to train a large model?
- Try 1 billion hyperparameter combinations to find the best?

They are only the edge cases of distributed machine learning.

The Sizes

All Sizes Matter!   Reducing Size Matters More!

	Small Data	Large Data
Small Compute	✓	✓
Large Compute	✓	✓

The Scope

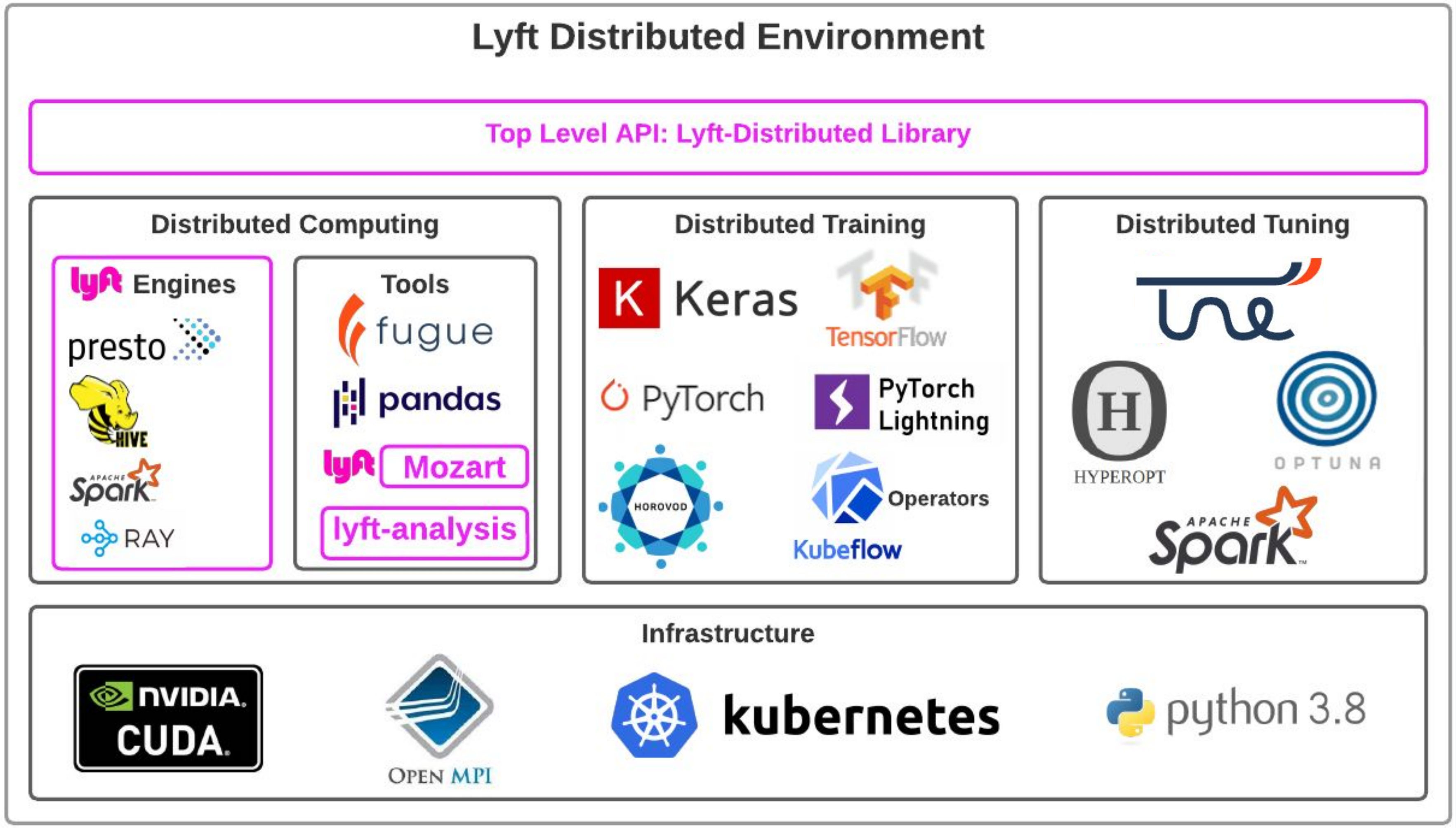
The entire ML Lifecycle needs distributed computing

Feature Engineering	Prototyping	Model Training	Serving	Monitoring
Easily prepare, ingest, retrieve features offline and online in real time using the same code solving online offline discrepancy	Enable rapid prototyping with easy access to features, compute environment and repos	Custom environments to train complex workflows on demand or scheduled	Deploy models to microservices based architecture and enable predictions through network calls.	Continuously monitoring models performance in production

Design Principles

- Making big data small
- Fast Iterations
- No restriction on modeling libraries and versions
- Layered-cake Approach
- Time/Cost Visibility
- Ease-of-use

Lyft Distributed Environment



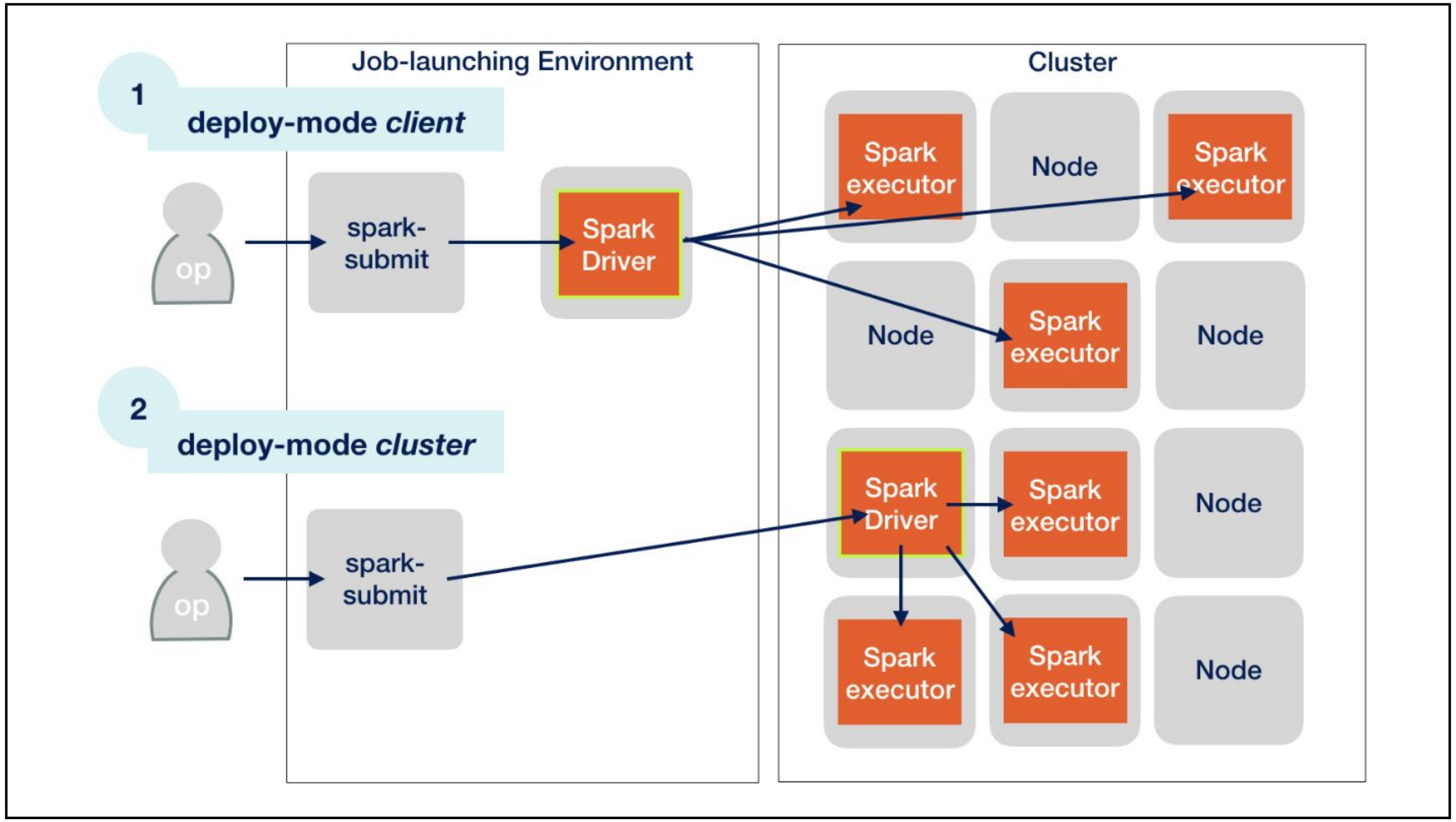
Distributed ML Platform @ Lyft

Motivation

- Efficiently utilizing compute resources
- Inadequate infrastructure leads to suboptimal modeling
    - Scaling vertically (powerful machines) is easier as opposed to scaling horizontally(more machines)
    - Underutilized resources in vertical scaling
    - Users need to sample down large datasets to fit into one machine
    - Larger engineering effort to set up

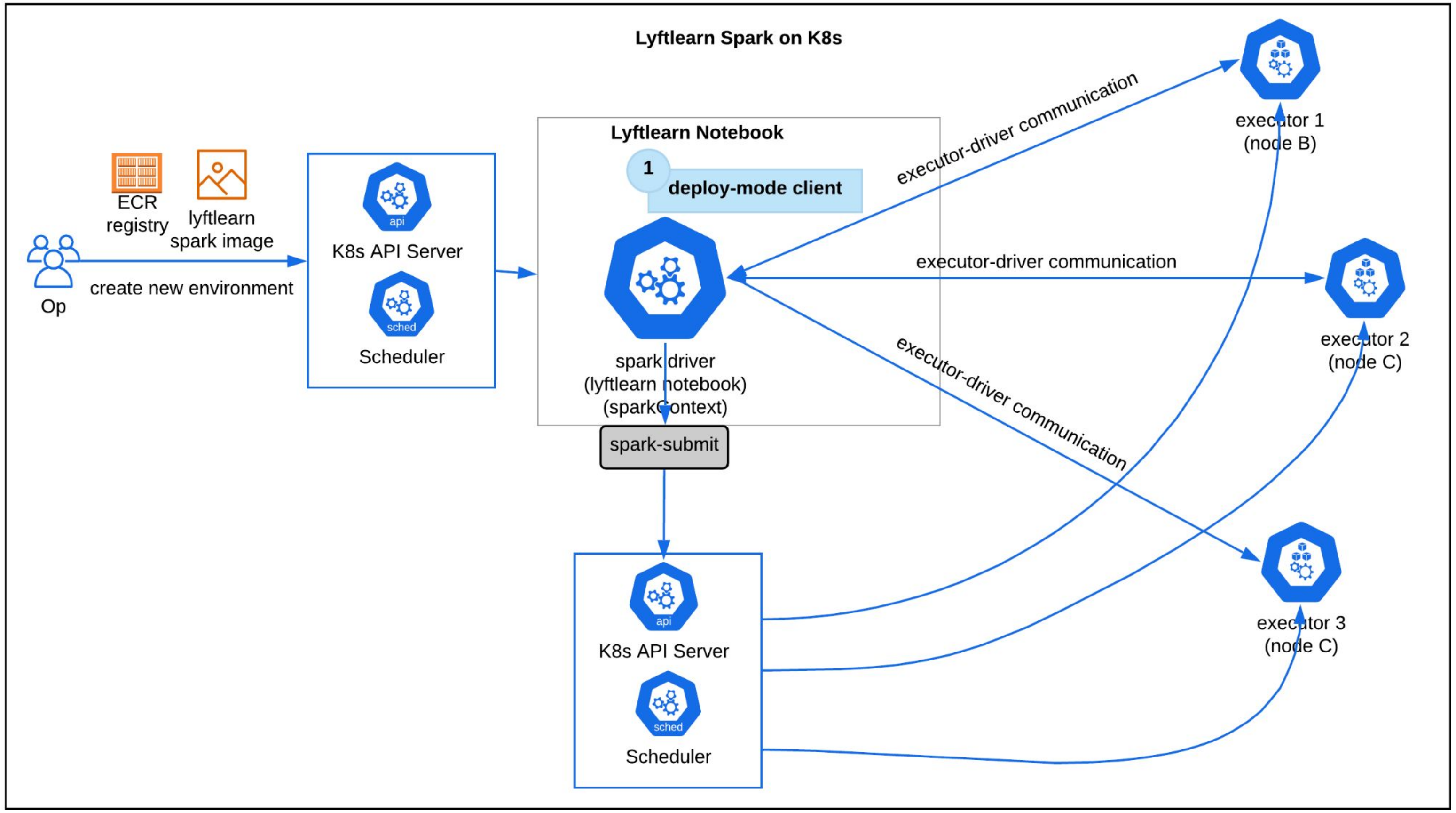
Distributed ML Platform @ Lyft

Spark on Kubernetes



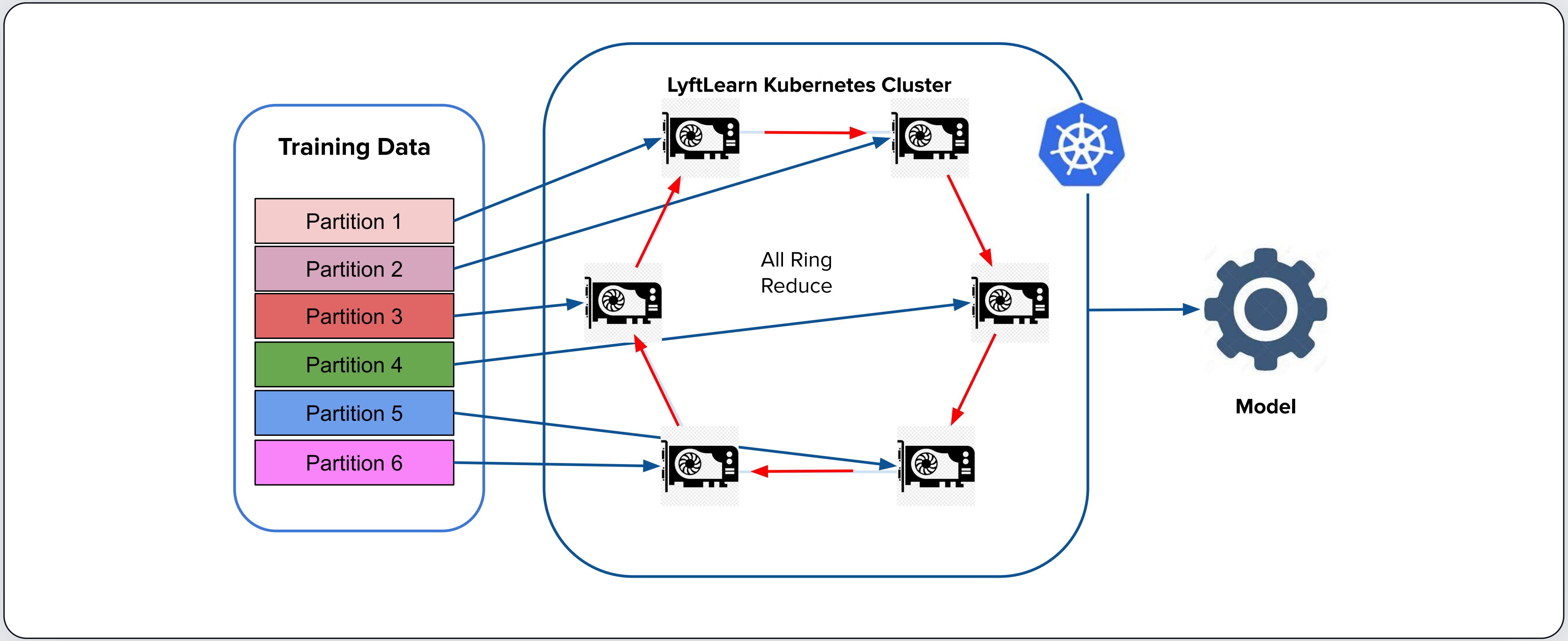
Distributed ML Platform @ Lyft

Spark on Kubernetes



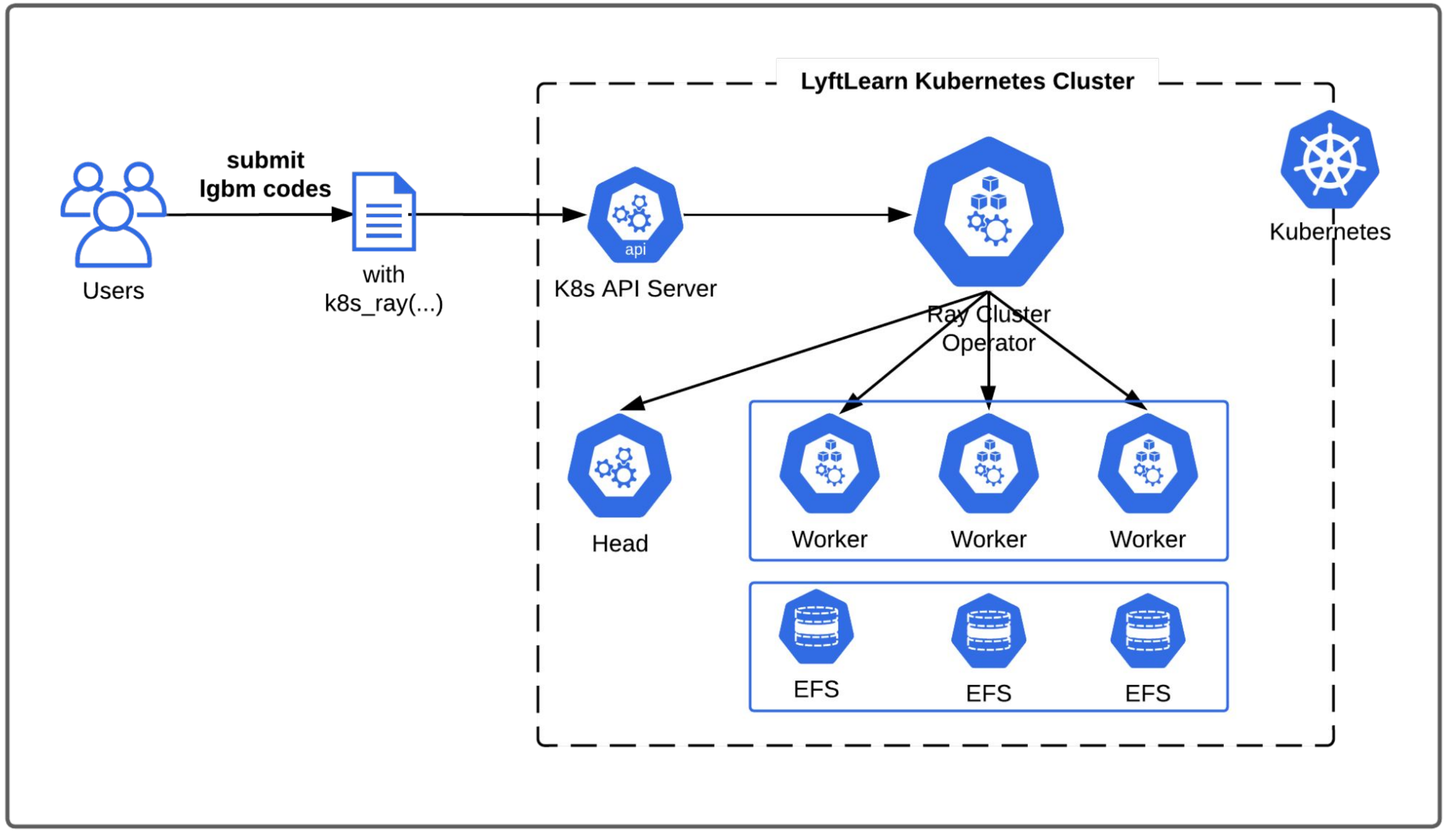
Distributed ML Platform @ Lyft

Distributed Deep Learning



Distributed ML Platform @ Lyft

Distributed LightGBM on Ray



LyftLearn Abstractions

Fugue with Spark on Kubernetes

```python
# schema: *,result:str
def compute_tasks(tasks:pd.DataFrame) -> Iterable[Dict[str,Any]]:
    for task in tasks.to_dict("records"):
        # do some expensive compute here
        task["result"] = str(task)
        yield task
```

```python
from lyft_distributed import k8s_spark, local_spark
from fugue import transform

with k8s_spark({"cluster":"4*4*4g"}) as session:
    sdf = session.createDataFrame([[1,"a"], [2, "b"]], "a:int,b:string")
    # current_spark means current active spark session
    result = transform(sdf, compute_tasks, engine="current_spark")
    print(type(result))
    result.show()
```

LyftLearn Abstractions

Fugue with Spark on Kubernetes

```
# schema: *,pred:double
def predict(df: pd.DataFrame) -> pd.DataFrame:
    region = df.region.iloc[0]
    model = load_model(region)
    return df.assign(pred = model.predict(df))
```

```
from fugue import transform

all_region_data = spark_session.read.parquet("<s3 location>")
result = transform(
    all_region_data, # data
    predict,         # logic
    partition={"by": "region"}, # logical partition
    engine= spark_session # run on current spark session
)
result.write.parquet("<s3 location>")
```

LyftLearn Abstractions

Unifying Spark & Ray

```
import ray
from lyft_distributed import k8s_spark, k8s_ray
```

```
# pre processing
with k8s_spark({"cluster":"16*16*4g"}) as spark:
    raw_data = spark.read.parquet("<s3 raw data>")
    features = transform(raw_data, feature_processor, engine=spark)
    features.write.parquet("<s3 train data>")
```

```
# training
with k8s_ray({"cluster":"4*12*8g"}):
    ray_bst = ray.get(train_lgbm_remote("<s3 train data>"))
```

```
# save model
ray_bst.booster_.save_model("<model path>")
```

```
# post processing
with k8s_spark({"cluster":"64*16*4g"}) as spark:
    test_data = spark.read.parquet("<s3 test data>")
    ray_bst_model = load_model("<model path>")
    transform(test_data, predict, params={"model":ray_bst_model}, engine=spark)
```

Distributed ML Platform @ Lyft

Challenges & Learnings - User Experience & Efficiency

- **Pain Points**
  - Having a standalone cluster is expensive.
  - Most of our DS experiments are bursty in nature.
  - Users want their custom docker images as Spark executor.
- **Mitigation**
  - Spark on K8s blended in well. With our existing ML infra was on K8s.
  - Users can use custom spark docker images.
  - On-demand on K8s is fast and cost efficient.
  - Spark usage increased by 60%. Cost dropped by 50%. Time reduced by 90%.

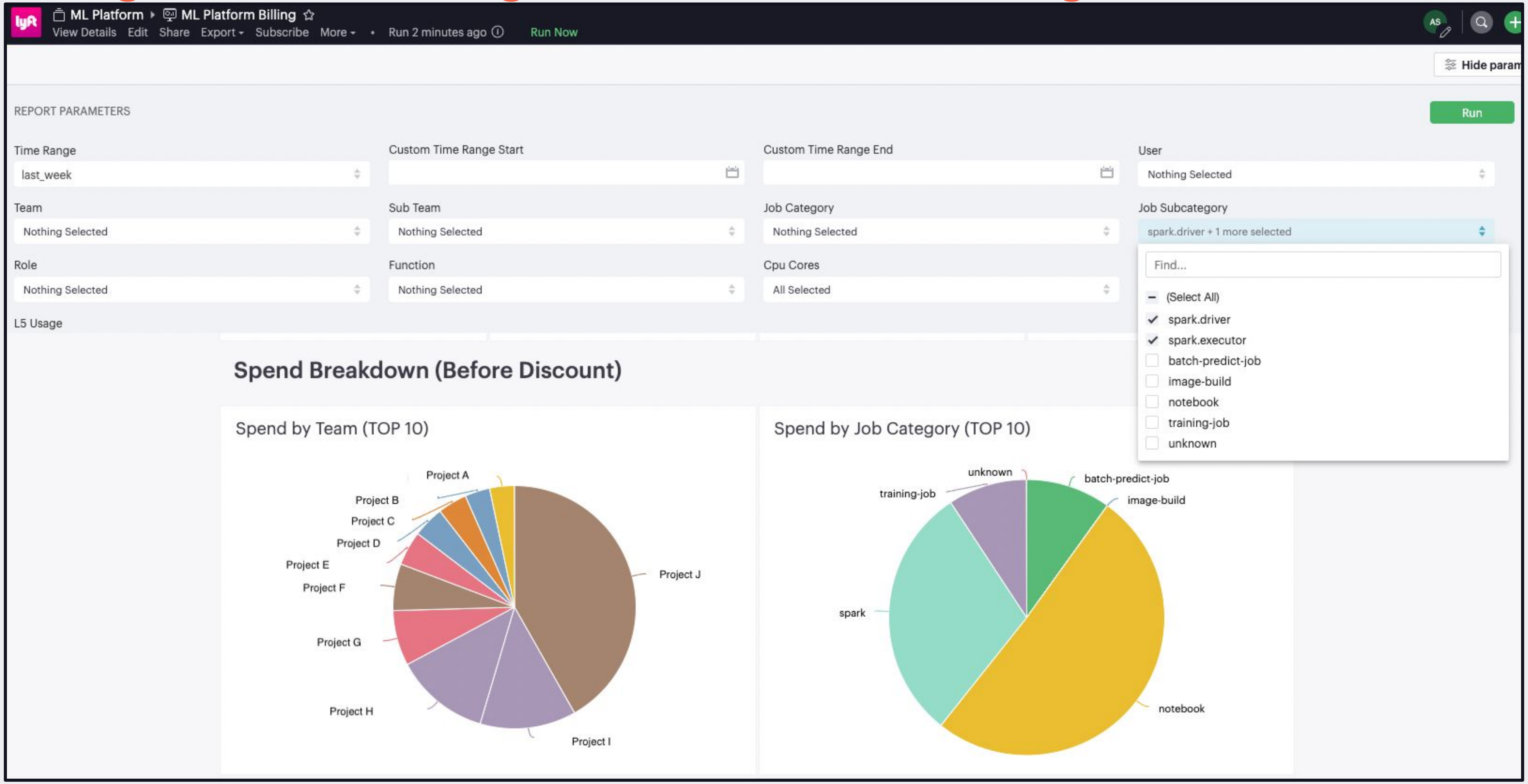
Distributed ML Platform @ Lyft

Challenges & Learnings - Cost Monitoring

- **Pain Points**
  - How to keep the cost low?
  - Preventing runaway clusters.
- **Mitigation**
  - Use K8s namespaces to isolate users.
  - Use custom labels and annotations on pods to detect spark executor pods.
  - Use K8s event listeners to track spark pod events and build cost dashboards.
  - Use custom Python SDK context managers for graceful shutdown of spark session.

Distributed ML Platform @ Lyft

Challenges & Learnings - Cost Monitoring



Distributed ML Platform @ Lyft

Challenges & Learnings - Preventing Job Starvation

- Pain Points
  - A user spawned 100s of spark jobs (1000s of executor pods) in a loop causing our K8s API to choke.
  - Caused DDOS for other namespaces. K8s etcd went down bringing down the availability of ML platform.
- Mitigation
  - Implemented K8s Resource Quota per namespace.
  - Backup K8s Job Controller uses custom labels on spark executors to destroy orphan pods.

ML Platform @ Lyft

Connect with us

- Engineering Blogs
  LyftLearn: ML Model Training Infrastructure built on Kubernetes
  How LyftLearn Democratizes Distributed Compute through Kubernetes Spark and Fugue
  Full-Spectrum ML Model Monitoring at Lyft

- Other Talks at Data + AI Summit 2022
  FugueSQL—The Enhanced SQL Interface for Pandas and Spark DataFrames
  Fugue Tune: Distributed Hybrid Hyperparameter Tuning

- Interested in working with us?
  Shiraz Zaman, Han Wang, Anindya Saha, Hakan Baba, Martin Liu, Mihir Mathur

Thank You