# CX, MODEL TRAINING AUTOMATION AND THE FEATURE STORE PROBLEM

September 26, 2019

COMCAST

# WHAT WE WILL TALK ABOUT TODAY

- ML USE CASE - CUSTOMER EXPERIENCE
- DATA AT SCALE IS HARD
- WHY WE NEED A FEATURE STORE
- SOLVING THE FEATURE STORE PROBLEM



COMCAST

COMCAST
NBCUNIVERSAL

A global media and technology company with several businesses, including Comcast, NBCUniversal, and Sky.

COMCAST

| Products & Services | Cable Networks | Broadcast | Film | Parks | Products & Services |
| --- | --- | --- | --- | --- | --- |
| | | | | | Channels & Content |
| Comcast Spectacor | | | | | |

Minority interest and/or non-controlling interest.
Slide is not comprehensive of all Comcast NBCUniversal assets
Updated: December 03, 2018

# ML Lifecycle – Roles & Workflow

ML Operations

Inception

Exploration

Model Development

Candidate Model Selection

Model Operationalization

Model Evaluation

Go Live Phase

Define Use Case

Identify Model Types & Explore Features

Create and publish new features

Create & Validate Models

Iterate

Iterate

Model Review

Model Selection

- Define Online Feature Assembly
- Define pipeline to collect outcomes
- Model Deployment and Monitoring

Collect new data & retrain

Evaluate Live Model Performance

Go Live with Selected Models

Monitor Live Models

# What is the Business Problem?

- We have Inefficient Processes == **WASTE**
- The **WASTE** costs money
- We don't use all of the data we have to provide an ideal customer experience

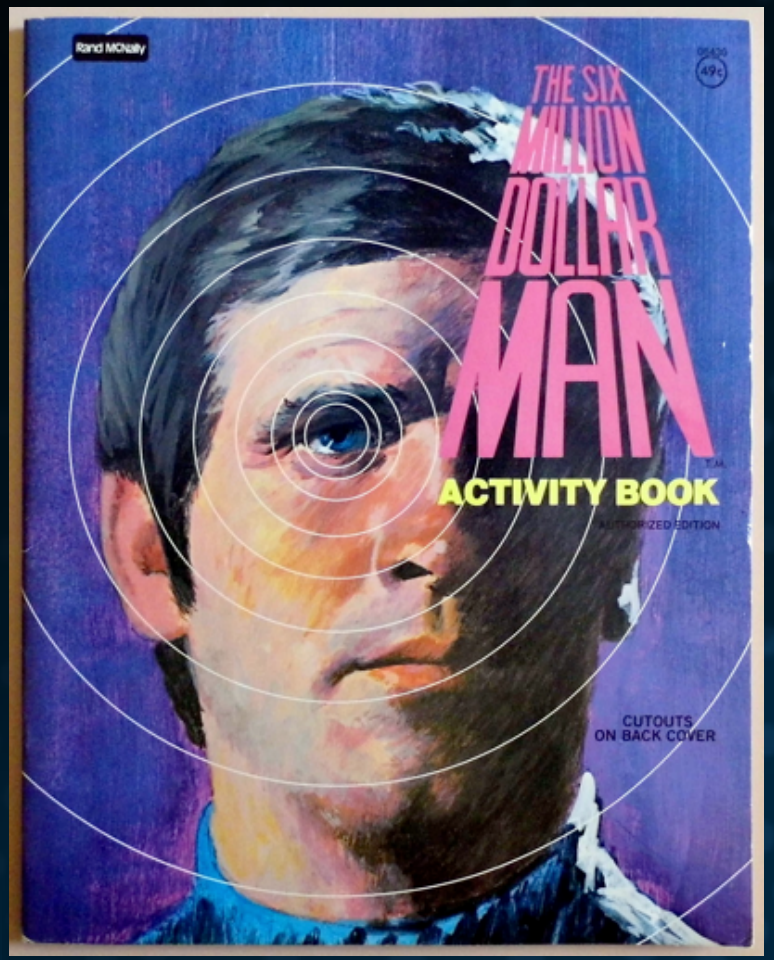
- GOAL: Take cost out of the business + drive KPIs
  - IVR / Digital Containment
  - XA / Chat Containment
  - Rework
  - Avoidable Truck Rolls

# Machine Learning and Human Empathy

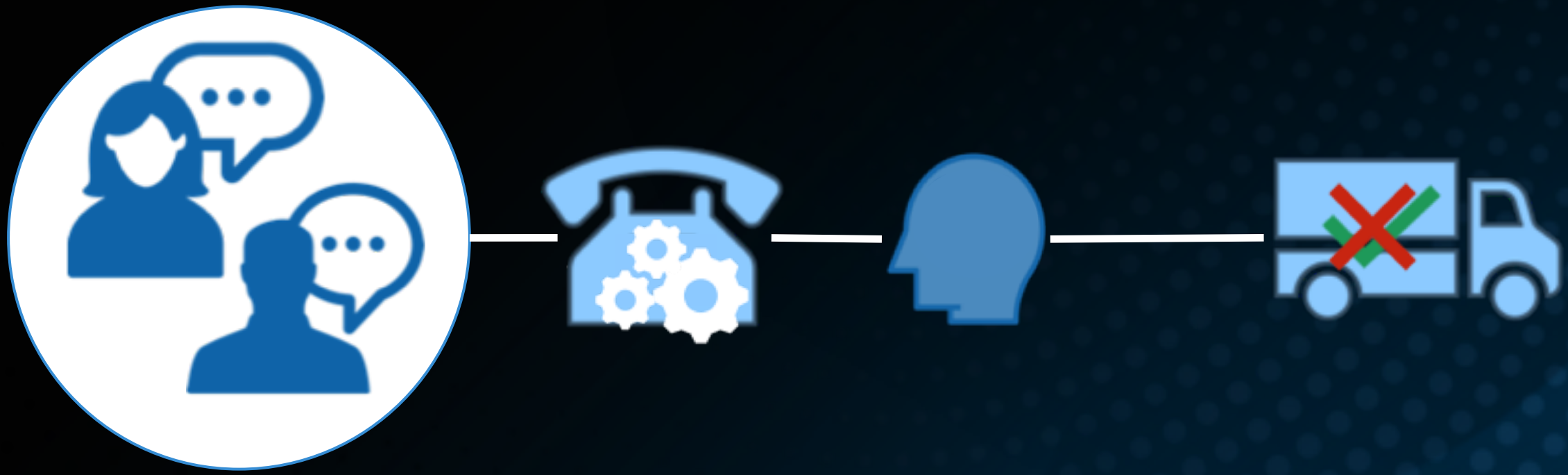
## Six Million Dollar Man

- The Six Million Dollar Man is a better analogy
  - Start with Quality People
  - Augmenting them with Technology
  - Up-skilling the Workforce

- Machine Learning + Human Decision Making = Better Results

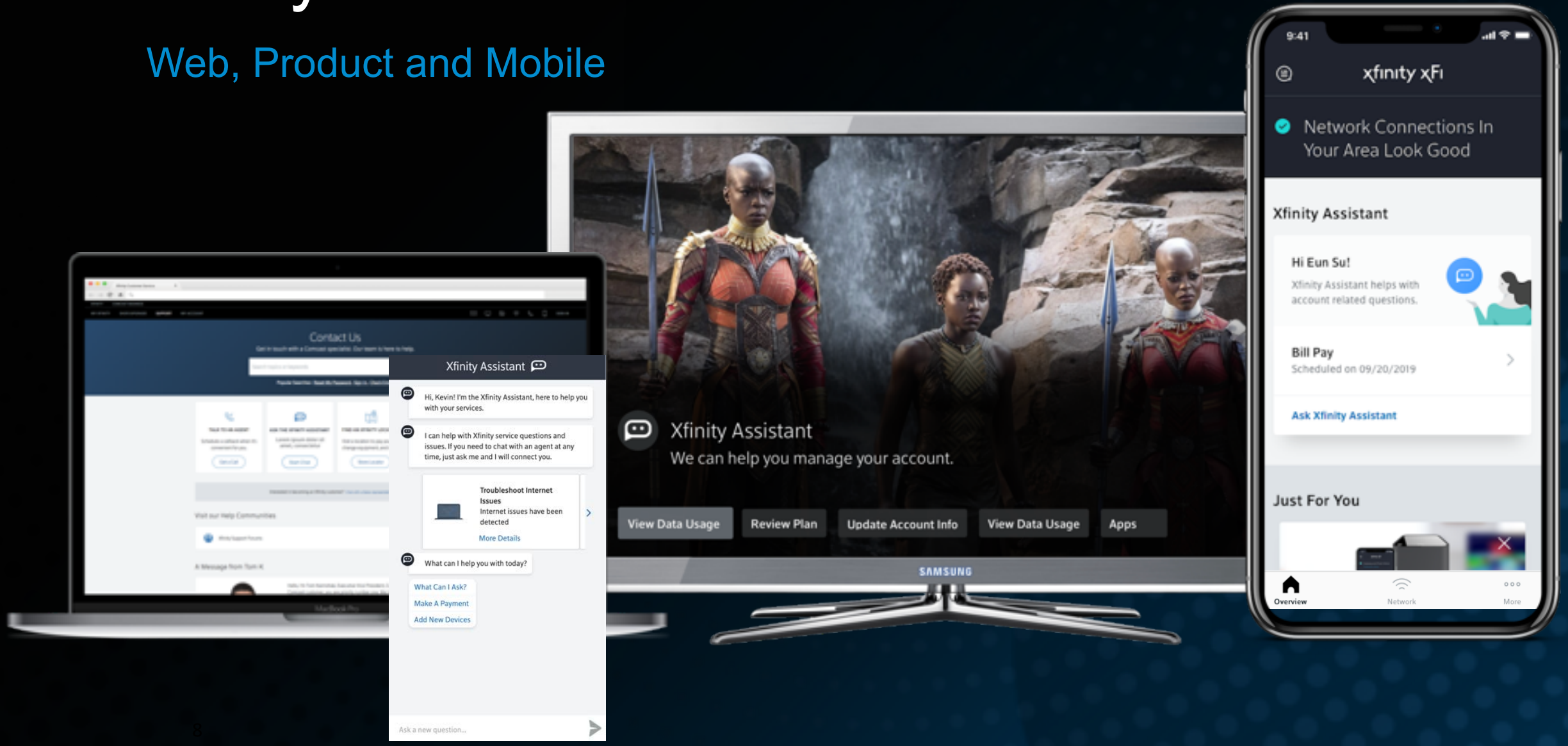


# Operational Waste During Service Calls

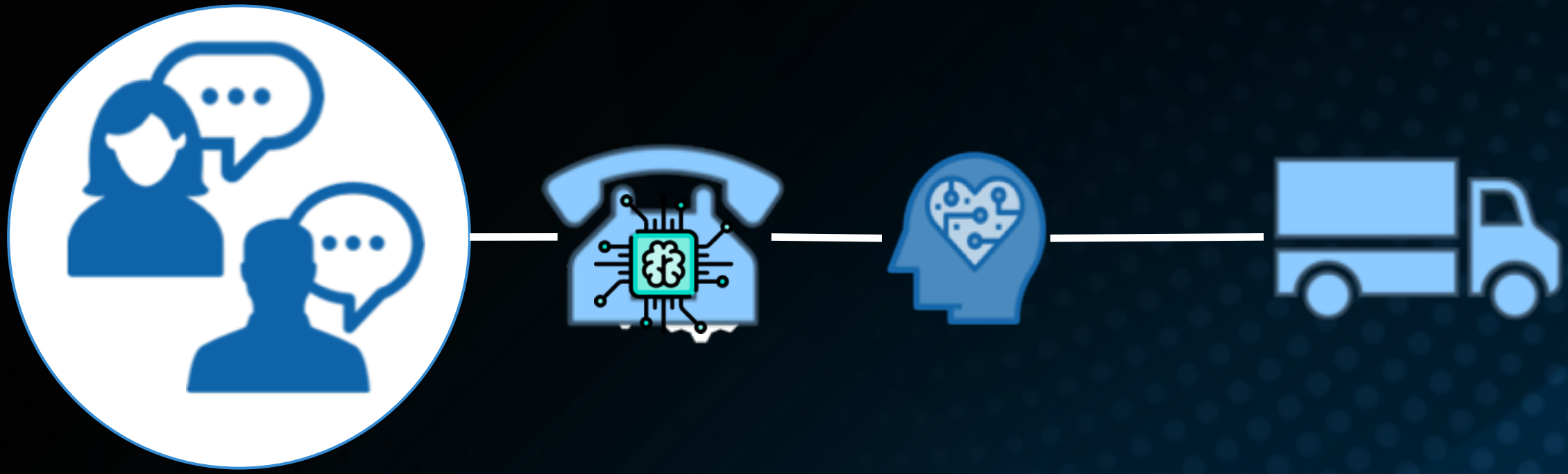


# Xfinity Assistant

## Web, Product and Mobile



# ML Powered Service Calls



# Machine Learning at Scale

BILLIONS AND BILLIONS

## STREAMING EVENTS

Product supply chain produces billions of features are engineered and enriched to create the ML dataset

HUNDREDS OF MILLION

## RECORDS UPDATES

Multiple feature sets are made available in the online feature store

TENS OF MILLION

## CUSTOMER ACCOUNTS

Predictions are ready to be made for Comcast customers that use Internet and Video products



COMCAST

# Customer Impacting Predictions YTD

**Millions of Predictions**

**60% HSD Predictions**

**40% X1 Predictions**

**85% of customers that used the recommendations did not call for repair in a 24-hour period**

# TECHNICAL OVERVIEW

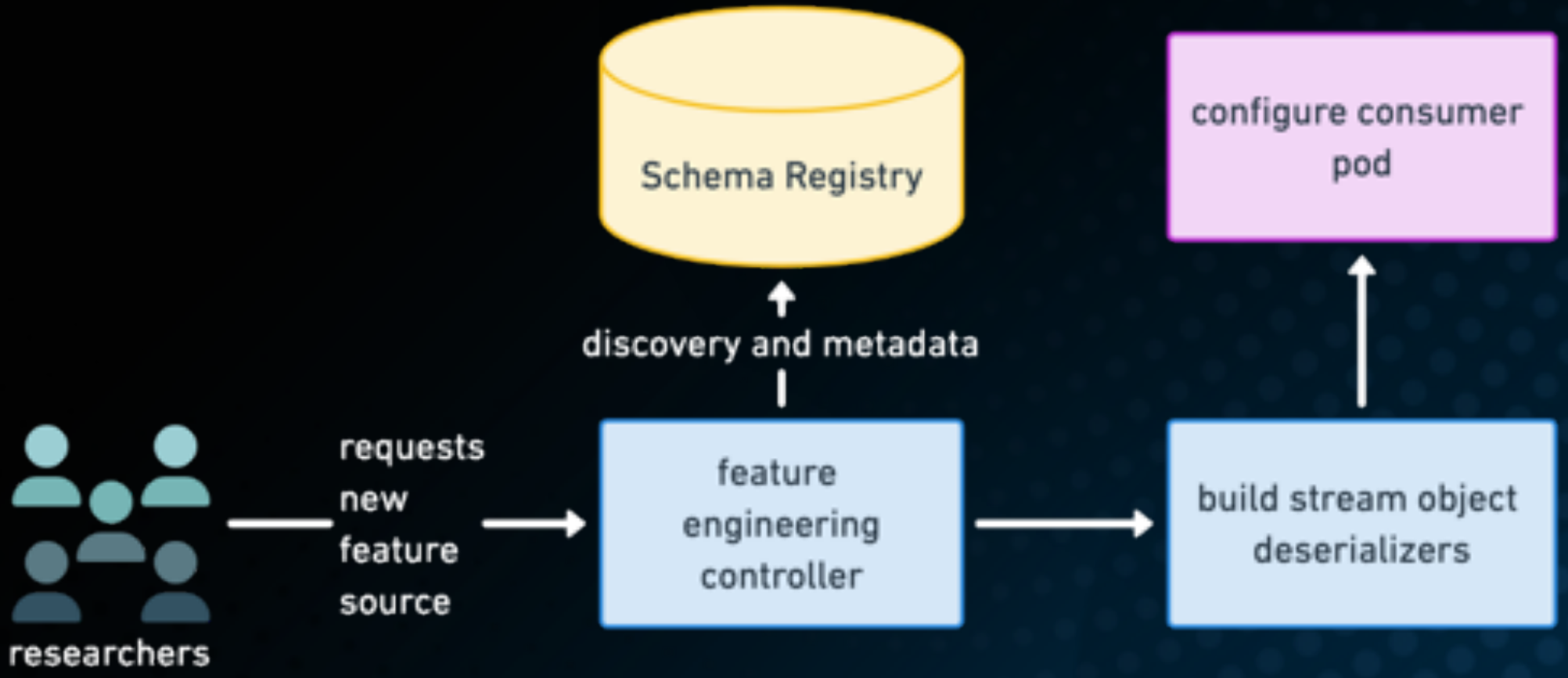
- METADATA
- DATA INGESTION
- FEATURE ENGINEERING
  - ”Feature store problem”
- MODEL TRAINING
- MODEL DEPLOYMENT
- MODEL SERVING
- AUTOMATION

- **BRINGING IT ALL TOGETHER**



COMCAST

# METADATA






COMCAST

# START SIMPLE - ABSTRACT

DATA INGESTION AND FEATURE ENGINEERING

## HOW TO MANIPULATE AND STORE DATA?




COMCAST

# ABSTRACT TO CONCRETE

## DATA INGESTION AND FEATURE ENGINEERING

|  | | |
|---|---|---|
| • Kafka<br>• Kinesis<br>• HTTP requests<br>• batch, files, etc. | • Windowing<br>• Aggregation<br>• Normalization<br>• Transformation<br>• etc. | Features ready to be passed into ML models |



COMCAST

# WHAT IS THE FEATURE STORE PROBLEM?

- WHAT IS A FEATURE STORE
- WHY DO YOU NEED ONE
- WHAT IS THE PROBLEM?



COMCAST

# FEATURE STORE

## TYPES OF FEATURE STATES TO PERSIST

- **WINDOWED DATA**
- **AGGREGATE DATA**
- **OTHER DATA RELATED TO INTERMEDIATE STEPS FOR FEATURE ENGINEERING**
  - Common indexes
  - Inference and predictions from other models
  - Etc.

Can be many things:

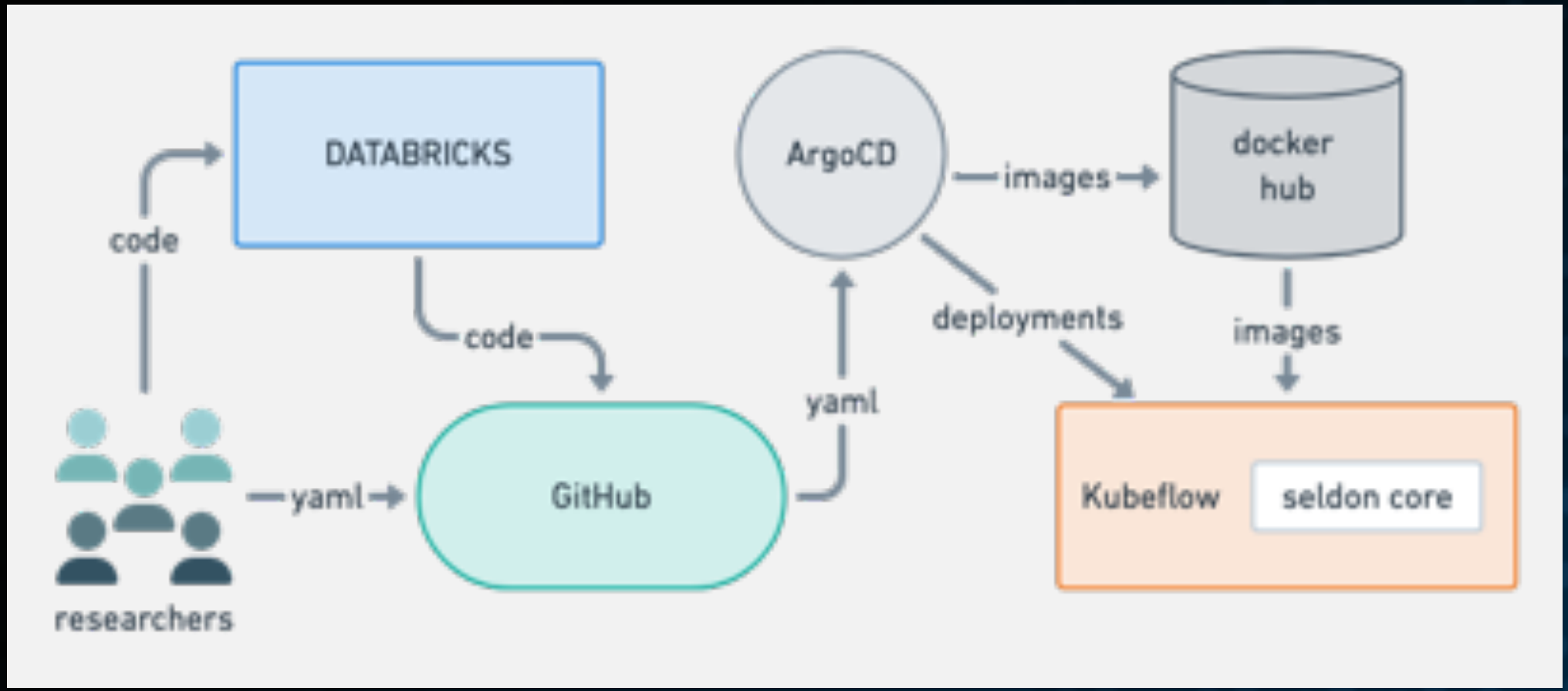
- Spark state store (Cassandra, RocksDB)
- Flink state backend (Memory, RocksDB)
- Custom: Hadoop (HDFS, HBase)
- Custom: Redis



COMCAST

# RESEARCH AND MODEL FLOW

## MODEL TRAINING AND DEPLOYMENT






COMCAST

# MLFLOW

## MODEL TRAINING

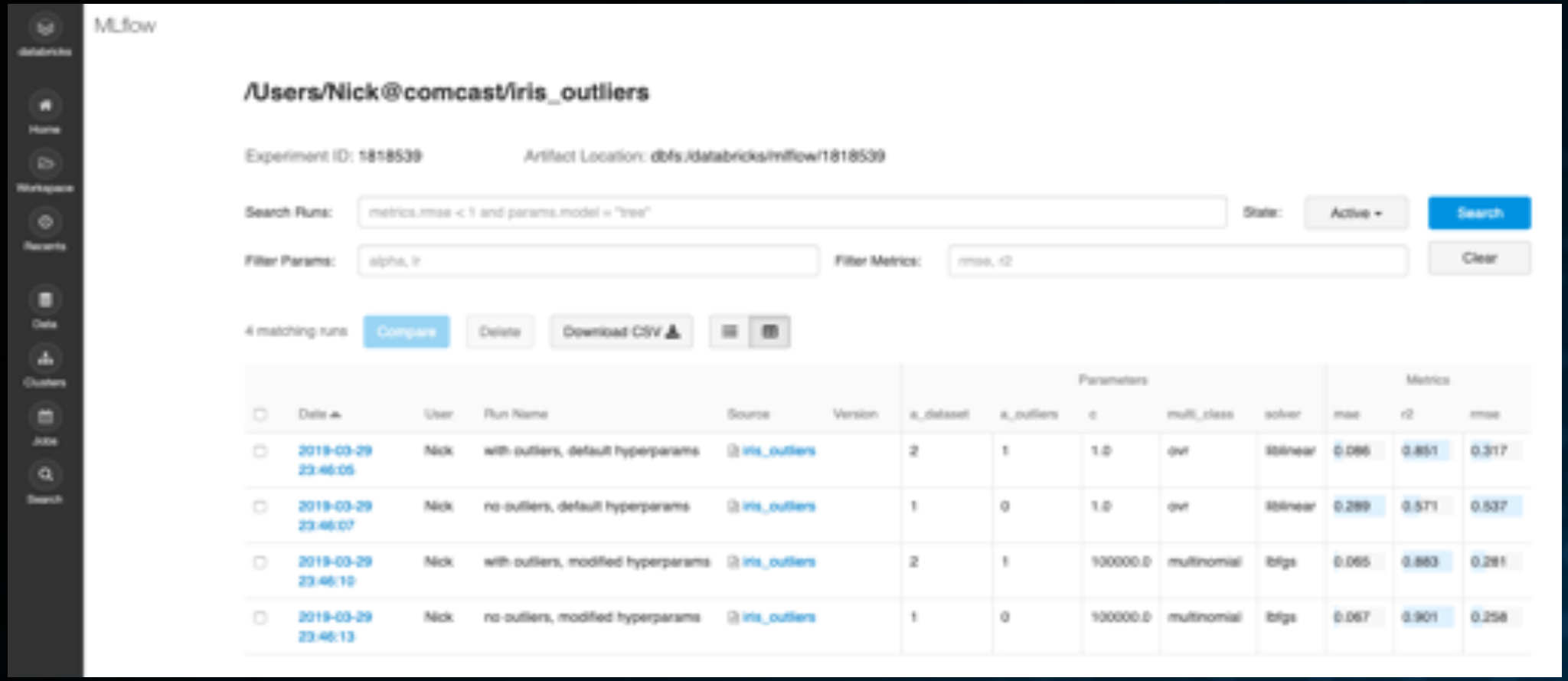
```
23      # Start a MLFlow experiment and label it
24      with mlflow.start_run(run_name="no outliers, default hyperparams"):
25          # train
26          clf = train(train_x, train_y, solver, C, multi_class)
27          # predict
28          predict = clf.predict(test_x)
29          # eval metrics
30          rmse, mae, r2 = eval_metrics(test_y, predict)
31
32          # log some params
33          mlflow.log_param("c", C)
34          mlflow.log_param("multi_class", multi_class)
35          # log some metrics
36          mlflow.log_metric("rmse", rmse)
37          mlflow.log_metric("r2", r2)
38          mlflow.log_metric("mae", mae)
39
40          # finally log the model and end
41          mlflow.sklearn.log_model(clf, "model")
42          mlflow.end_run()
```



COMCAST

# MLFLOW

## MODEL EXPERIMENT TRACKING






COMCAST

# MODEL SERVING IN DETAIL

## MODEL SERVING

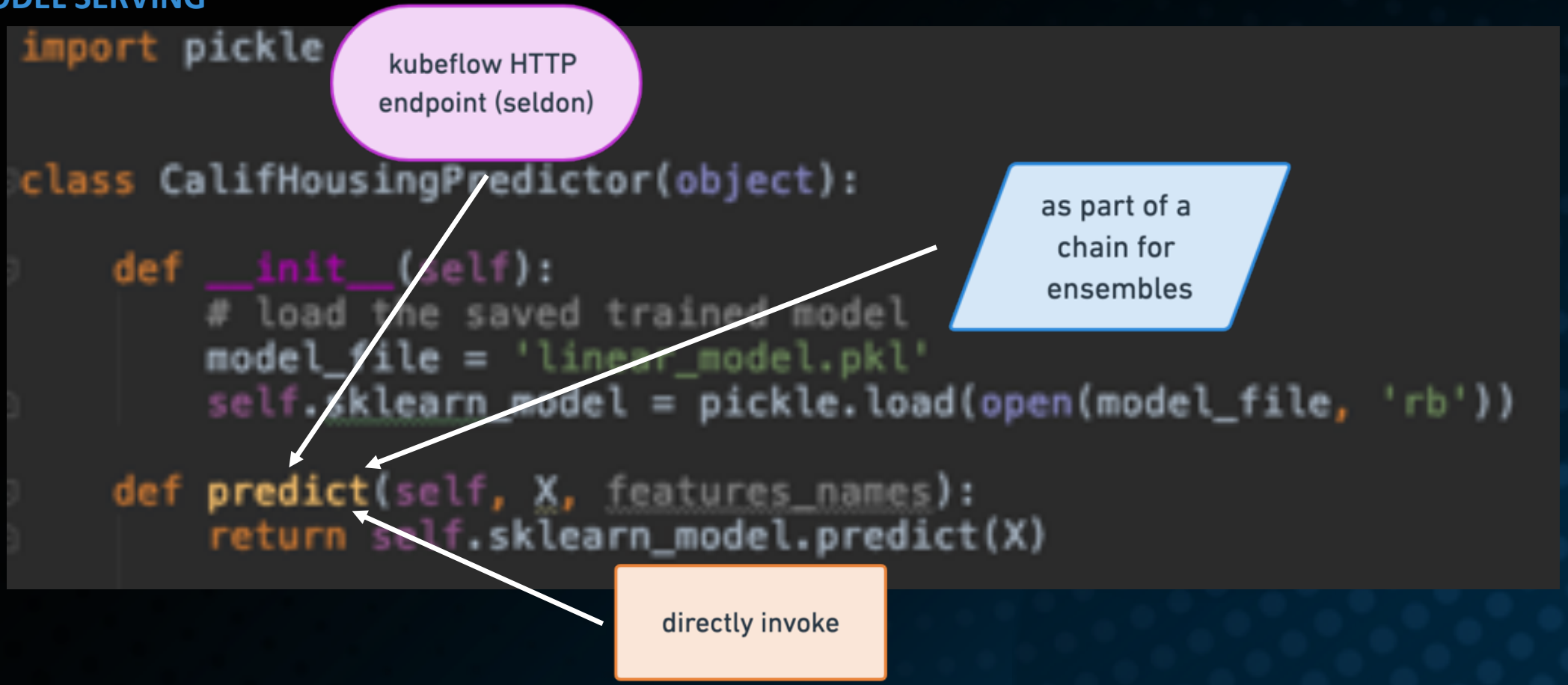
```
import pickle


class CalifHousingPredictor(object):

    def __init__(self):
        # load the saved trained model
        model_file = 'linear_model.pkl'
        self.sklearn_model = pickle.load(open(model_file, 'rb'))

    def predict(self, X, features_names):
        return self.sklearn_model.predict(X)
```



COMCAST

# MODEL SERVING - MANY POSSIBILITIES

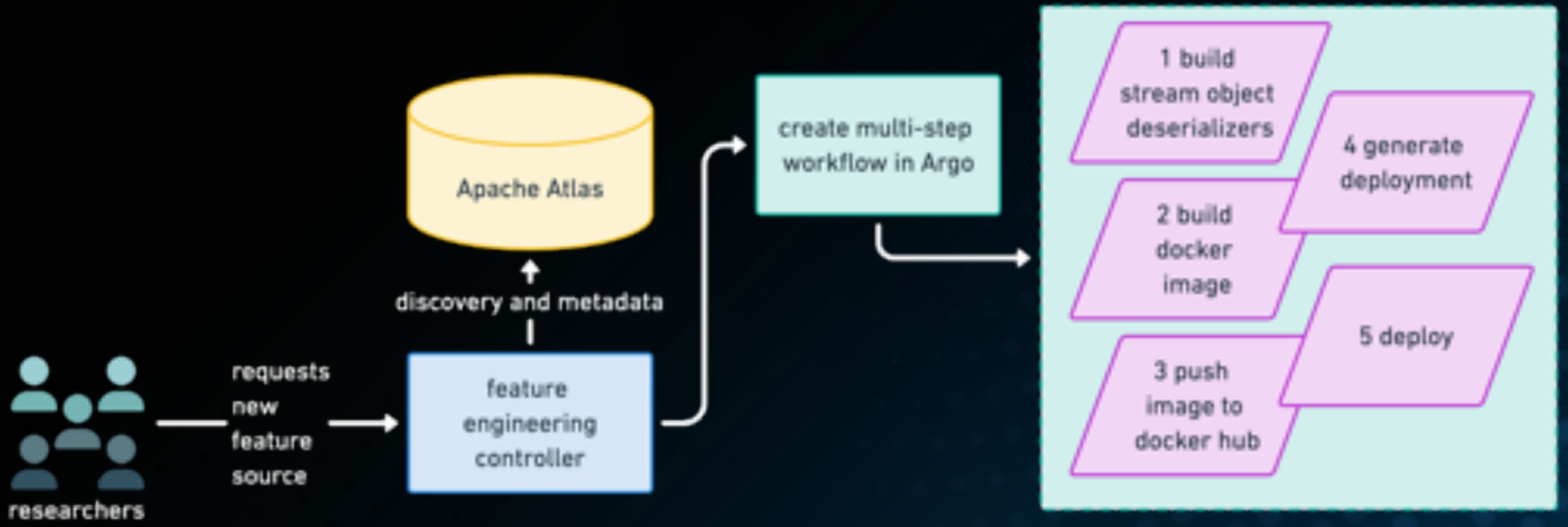
## MODEL SERVING






COMCAST

# PIPELINE AUTOMATION

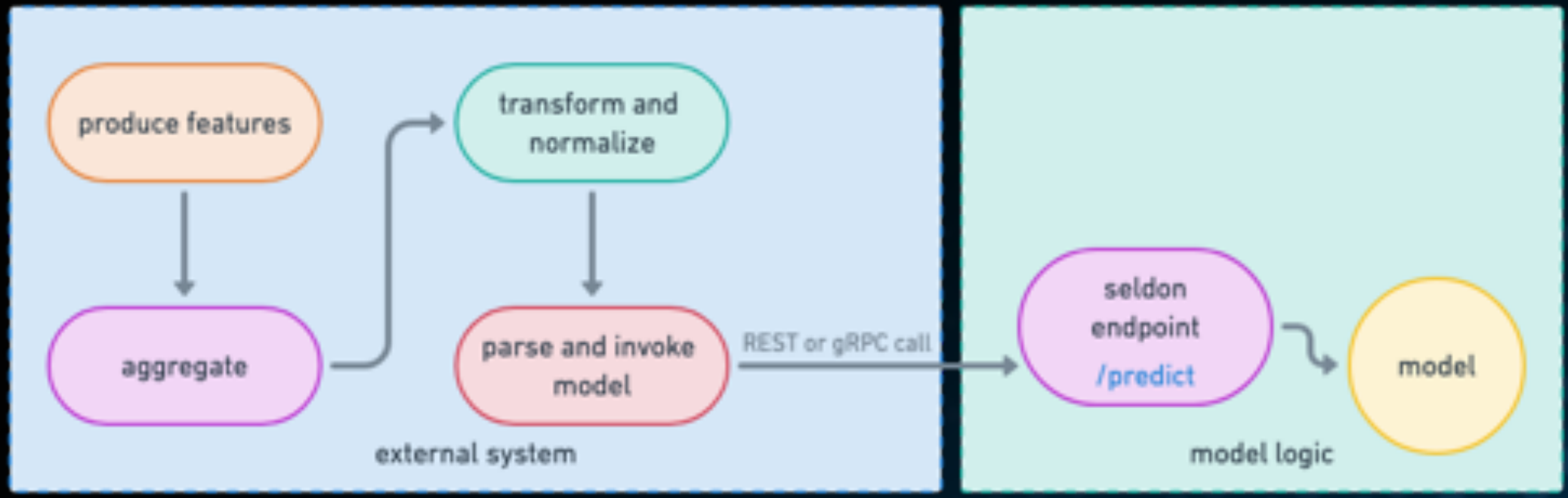





COMCAST

# BRINGING IT ALL TOGETHER

## MODEL SERVING WITH SEPARATE FEATURE ENGINEERING

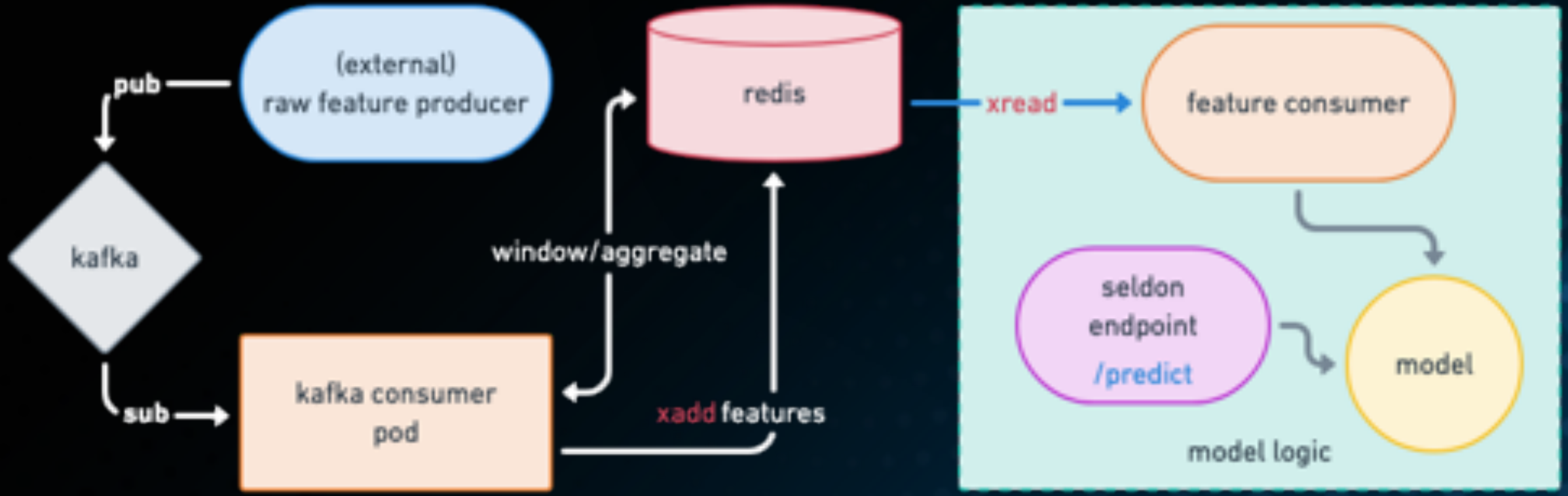





COMCAST

# PERFORMANCE PIPELINE (SOLVING THE FEATURE STORE PROBLEM)

## MODEL SERVING WITH INTEGRATED FEATURE ENGINEERING

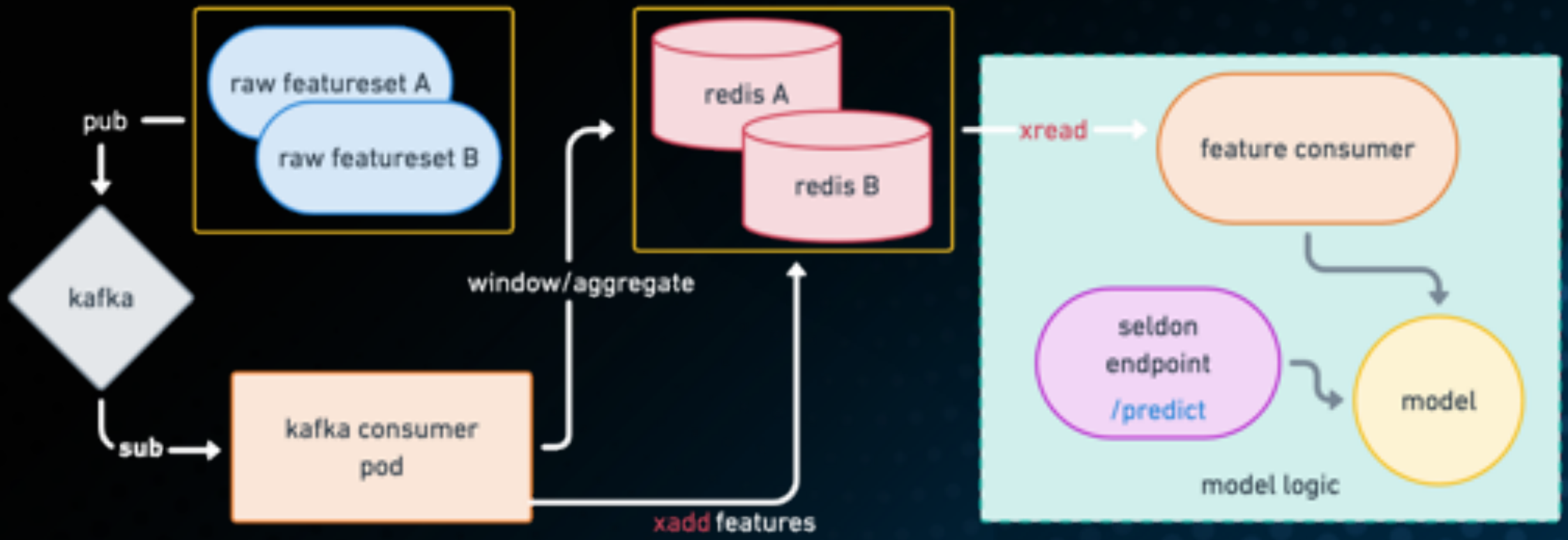





COMCAST

# BUT IN-MEMORY IS SMALL?

## MODEL SERVING: SPLIT UP THE IN-MEMORY DB BY FUNCTION






COMCAST

# DEMO CODE

EXAMPLE CODE  USING KUBEFLOW, KAFKA,

PYTHON, REDIS AND A SIMPLE MODEL

HTTPS://GITHUB.COM/COMCASTSAMPLES

“END TO END ML FEATURE STREAMING WITH KUBEFLOW KAFKA AND REDIS”



COMCAST


COMCAST