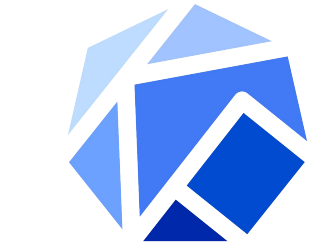



# Introducing KFServing : Cloud Native Serverless Inferencing

Dan Sun (dsun20@bloomberg.net)
Ellis Bigelow (ellisbigelow@google.com)

# Oh, you want to deploy your model on K8s?

## First, can you become an expert in ...

- Model Serialization
- Model Servers
- HTTP/GRPC
- Containerization
- GitOps
- Kubectl
- Deployments, Services
- HPAs, VPAs, KPAs
- Readiness/Liveness Probes
- Persistent Volumes
- Service Meshes
- Cloud Events
- GPUs

# Introducing KFServing

- Solve the model serving problem for **Data Scientists.**
- Create an intuitive and consistent experience.
- Simple enough for Day 1, Powerful enough for Day 100.
- Modern Production Features aren't just for WebApps.

# Experts fragmented across industry

- Seldon Core was pioneering Graph Inferencing.
- IBM and Bloomberg were exploring serverless ml lambdas.
- Google had built a common Tensorflow HTTP API for models.
- Microsoft Kubernetizing their Azure ML Stack


# Putting the pieces together

- Kubeflow created the conditions for collaboration.
- A promise of open code and open community.
- Shared responsibilities and expertise across multiple companies.
- Diverse requirements from different customer segments


# A Clean Interface

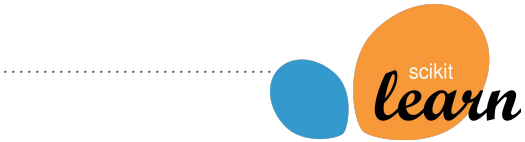
```yaml
apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "InferenceService"
metadata:
  name: "sklearn-iris"
spec:
  default:
    sklearn:
      storageUri: "gs://kfserving-samples/models/sklearn/iris"
```


```yaml
apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "InferenceService"
metadata:
  name: "flowers-sample"
spec:
  default:
    tensorflow:
      storageUri: "gs://kfserving-samples/models/tensorflow/flowers"
```

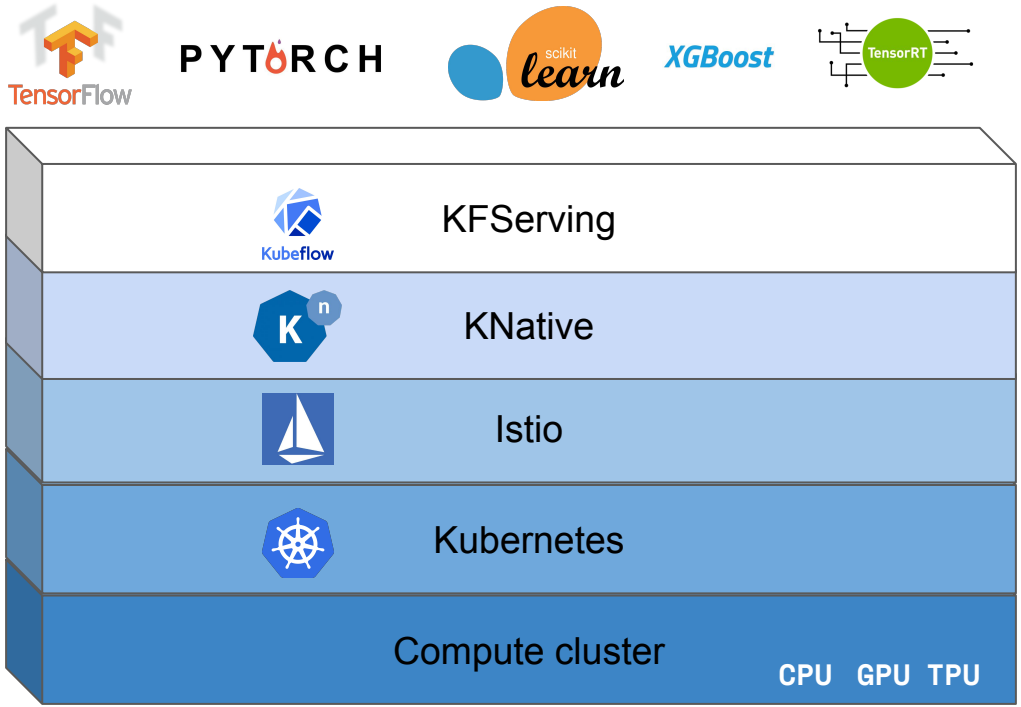

```yaml
apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "KFService"
metadata:
  name: "pytorch-cifar10"
spec:
  default:
    pytorch:
      storageUri: "gs://kfserving-samples/models/pytorch/cifar10"
      modelClassName: "Net"
```


# A Clean Interface

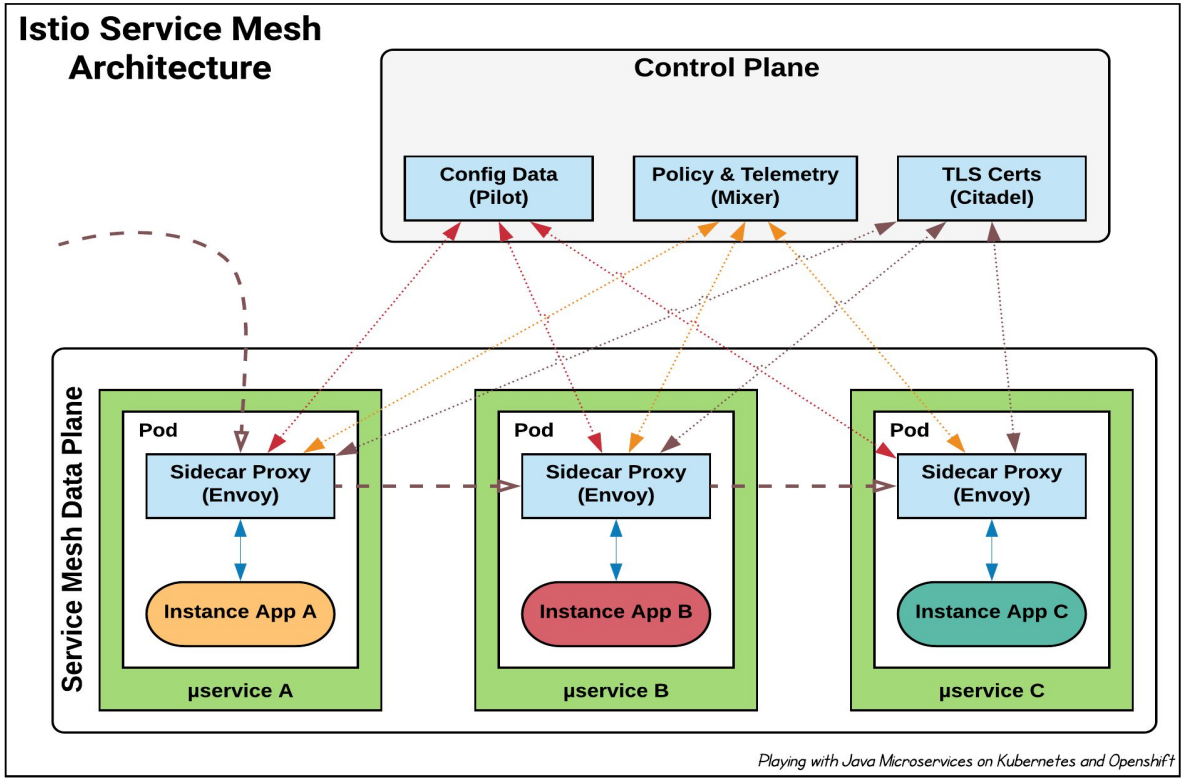
```
apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "KFService"
metadata:
  name: "sklearn-iris"
spec:
  default:
    sklearn:
      storageUri: "gs://kfserving-samples/models/sklearn/iris"
      serviceAccount: inferencing-robot
      minReplicas: 3
      maxReplicas: 10
      resources:
        requests:
           cpu: 2
           gpu: 1
           memory: 10Gi
  canaryTrafficPercent: 25
  canary:
    sklearn:
      storageUri: "gs://kfserving-samples/models/sklearn/iris-v2"
      serviceAccount: inferencing-robot
      minReplicas: 3
      maxReplicas: 10
      resources:
        requests:
           cpu: 2
           gpu: 1
           memory: 10Gi
```



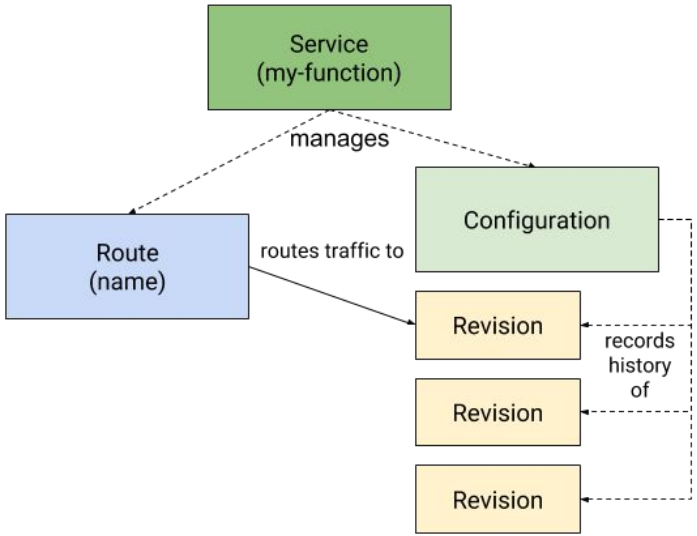

# Cloud Native Layers




# ML Service Meshes?

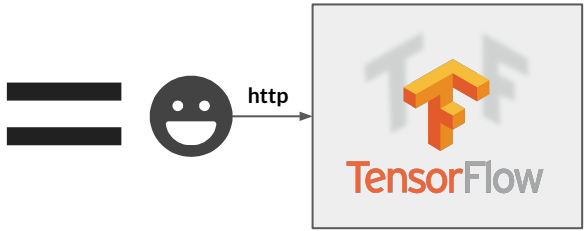



# Serverless Inferencing?


# But the Data Scientist Sees...

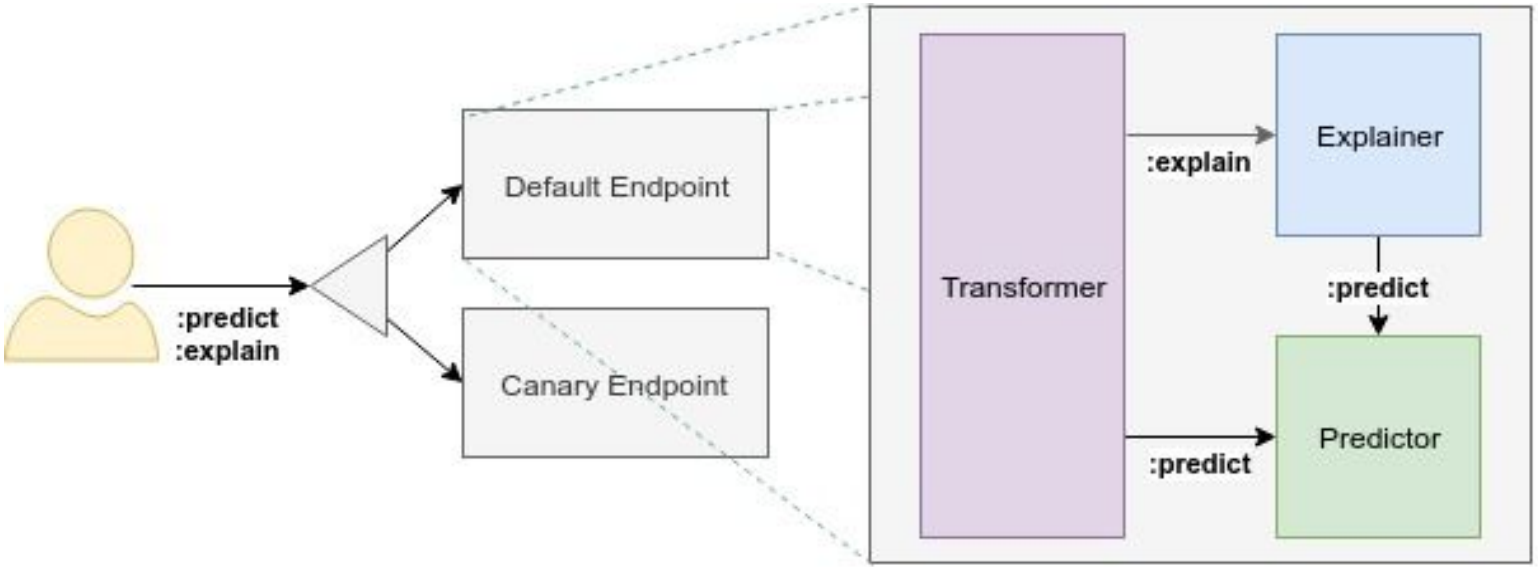
```
apiVersion: "serving.kubeflow.org/v1alpha1"
kind: "KFService"
metadata:
  name: "flowers-sample"
spec:
  default:
    tensorflow:
      modelUri: "gs://kfserving-samples/models/tensorflow/flowers"
```




- A pointer to a Serialized Model File
- 8 lines of YAML
- A live model at an HTTP endpoint

- Scale to Zero
- GPU Autoscaling
- Safe Rollouts
- Optimized Containers
- Network Policy and Auth
- Tracing
- Metrics
- ...

# Opinionated ML Microservices




# KFServing on Bloomberg's Data Science Platform

- Data science platform for Bloomberg's ML practitioners
- Runs on computing clusters with GPU nodes
- Bloomberg has a sizeable number of data science/AI teams and many of them need to run production inference services

## Why KFServing?

- Out-of-box model serving, standardize model serving across teams
- Model explainability, Inference graph (A/B Testing, Multi-Arm Bandit)
- Open community often leads to better product
- Easily run on our existing Istio/Knative stack

TechAtBloomberg.com



Bloomberg

Engineering

# Serving Models on Production


Larger data

Sample data exploration

Local development

Debugging (tier 1)

Larger-scale training

Debugging (tier 2)

PROD data

PROD training

PROD model QA

PROD serving

TechAtBloomberg.com

Bloomberg

Engineering

# Bloomberg Production Requirements

- Prod SLA with 24/7 support
- Production safe rollout procedure
- Build process for images (e.g., pre/post processing)
- Approval process for model serving deployment
- User debugging ability and logging, tracing
- Metrics monitoring for latency/throughput
- Alerting on inference service issues
- Separate inference server cluster from model training
- ...

  KFServing is a production grade inference solution.

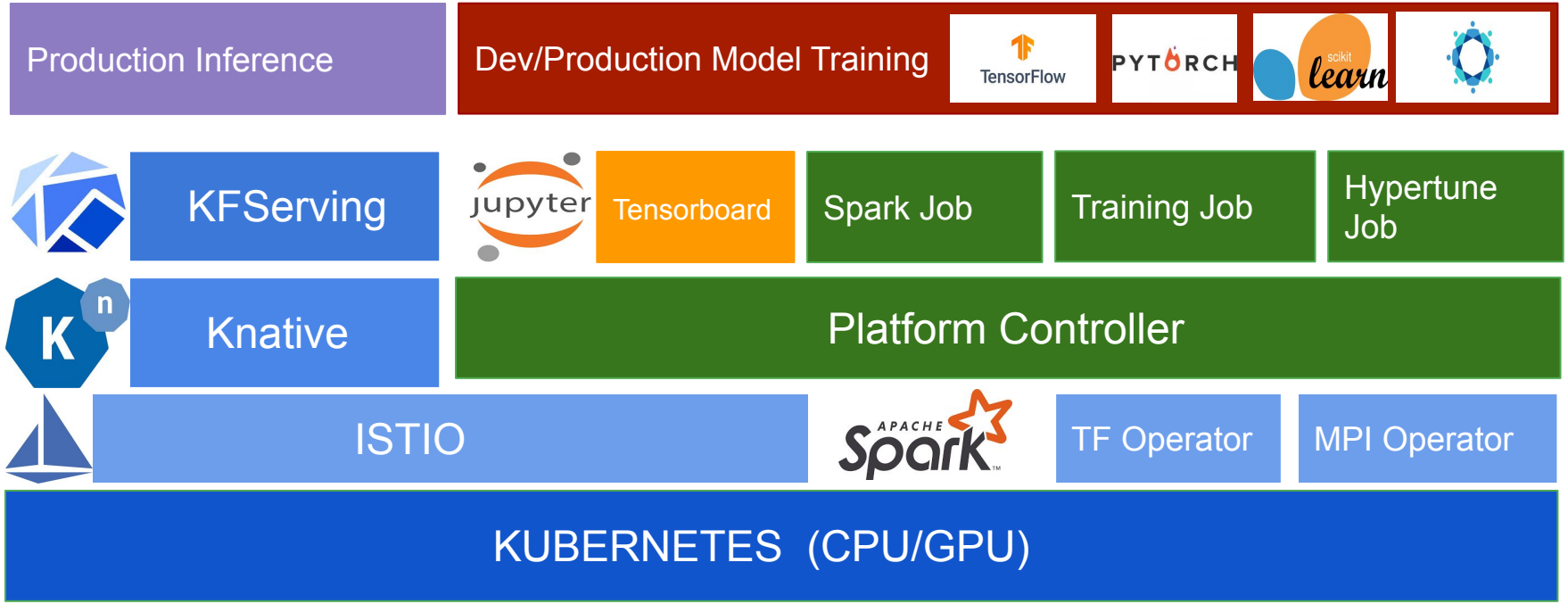




Bloomberg

Engineering

# Platform Architecture




TechAtBloomberg.com



Bloomberg

Engineering

# Bloomberg Use Cases

- Standardized model serving across ML frameworks
- Safe production rollout with KFServing canary strategy
- Pre/Post processing before/after model prediction with KFServing transformer
- Text classification model explainability with KFServing Alibi explainer integration
- Inference on news kafka streams
- Tensorflow/PyTorch GPU Inference
- GPU Sharing for models (Proposal, Ellis Bigelow [Google]/Dan Sun)
- A/B Testing (Proposal, Kai-Zhan Lee & Dan Sun)





Bloomberg

Engineering

# Inference Service with Canary Rollout

```
apiVersion: serving.kubeflow.org/v1alpha2
kind: InferenceService
metadata:
  name: canary-example
spec:
  default
    predictor:
      tensorflow:
        storageUri: s3://examples/bert/v1
  canary
    predictor:
      tensorflow:
        storageUri: s3://examples/bert/v2
  canaryTrafficPercent: 10
```

```
apiVersion: serving.kubeflow.org/v1alpha2
kind: InferenceService
metadata:
  name: canary-example
spec:
  default
    predictor:
      tensorflow:
        storageUri: s3://examples/bert/v2
```

# Inference Service with Transformer

```
apiVersion: serving.kubeflow.org/v1alpha2
kind: InferenceService
metadata:
  name: text-classification-example
spec:
  default
    transformer:
      custom:
        container:
          image: text-transformer:v1
    predictor:
      pytorch:
        modelClassName: TorchTextClassifier
        modelClassKwargs: {"embedding_dim": 200}
        storageUri: s3://examples/text-classifier
```

Pre/Post Processing

PyTorch Model Server

```
class TextTransformer(Transformer):

    def preprocess(self, inputs: Dict) -> Dict:

        return {'instances': [text_transform(instance) for
instance in inputs['instances']]}


    def postprocess(self, inputs: Dict) -> Dict:

        return np.argmax(inputs)
```

TechAtBloomberg.com



Bloomberg

Engineering

# Custom Inference Service with Image Build Job

```
class InhouseModel(KFServing.KFModel):

   def load(self):

       //load model


   def predict(self, request: Dict) -> Dict:

       instances = request['instances']

       inputs = [Document(instance) for instance in
instances]

       return model.predict(inputs)
```

Build Image

```
apiVersion: serving.kubeflow.org/v1alpha2
kind: InferenceService
metadata:
  name: custom-prediction
spec:
  default
    predictor:
      custom:
        image: ds/custom-prediction:v1
        env:
          - name: STORAGE_URI
            value: s3://model/v1
```

TechAtBloomberg.com



Bloomberg

Engineering

# Inference Service with Text Explainer

ALIBI

- Alibi : library from Seldon for ML model explainability/interpreting and monitoring
- Black box instance based model explanation with high precision rules called anchor, representing local sufficient conditions for prediction
- Support multiple use cases e.g tabular, text and  image data classification.
- Implemented based on Anchors: High-Precision Model-Agnostic Explanations

# Inference Service with Text Explainer

ALIBI

```
apiVersion: serving.kubeflow.org/v1alpha2
kind: InferenceService
metadata:
  name: text-explainer
spec:
  default:
    explainer:
      alibi:
        type: AnchorText
        storageUri: s3://spacy-model
        config:
          use_unk: false
          sample_prob: 0.5
    predictor:
      custom:
        container:
          image: ds/text-classifier:v1
        storageUri: s3://text-model
```

Alibi Text Explainer

In-House NLP Model Server

**Anchor consists of the words that need to be present to ensure a prediction, regardless other words.**

Anchor: flashy
Precision: 0.99
Examples where anchor applies and model predicts negative:
a visually flashy but psychologically preferable and emotionally unintelligible action behind style and laziness .

Examples where anchor applies and model predicts positive:
each visually flashy but narratively opaque and psychologically asinine diet while style and mystification

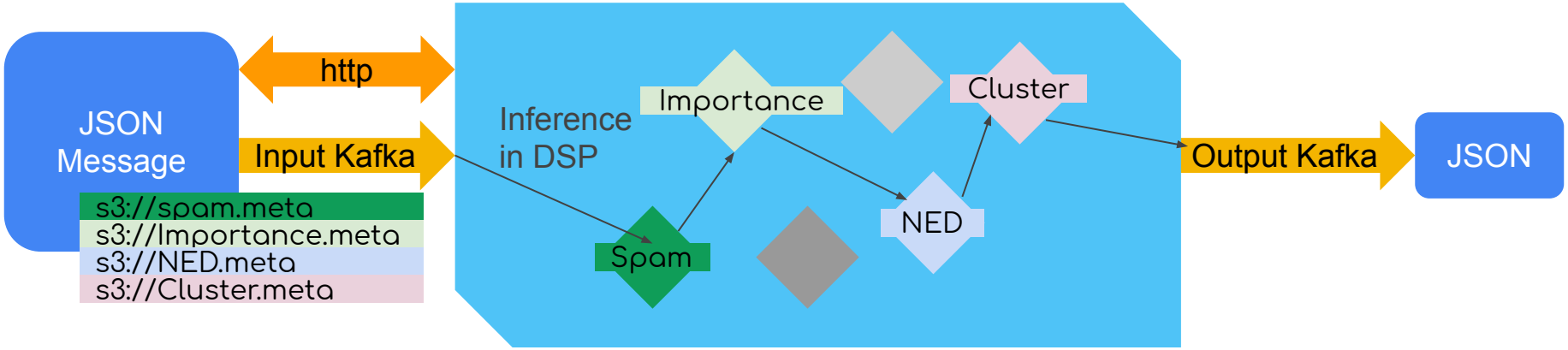
TechAtBloomberg.com



Bloomberg

Engineering

# Inference on Kafka Streams



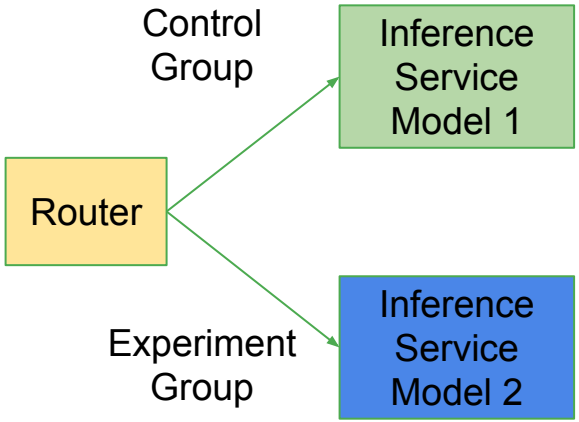

TechAtBloomberg.com

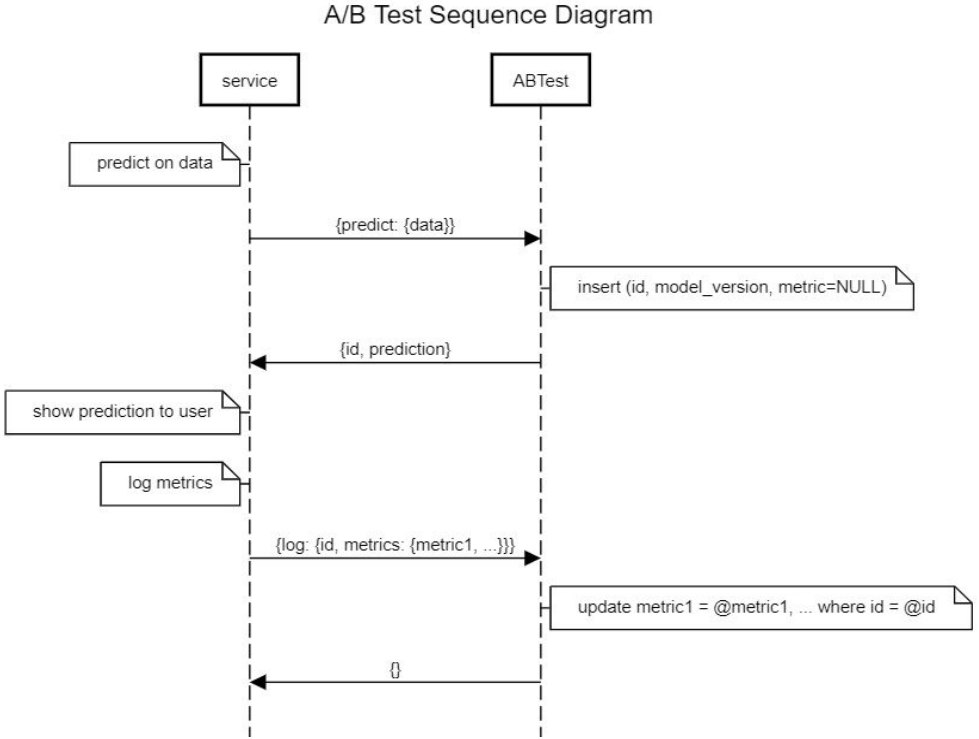


Bloomberg

Engineering

# Inference Router A/B Testing Proposal




TechAtBloomberg.com



Bloomberg

Engineering

# Production Model Deployment Process


Kubeflow

Jenkins

Knative/Tekton Build Job

Serving Image version

ML model Source Code

CI Pipeline

Image Repo

KFServing Deployment (Git Ops)

Approval process

Inference Service

Training Job

Model version

Model Store

Registered Model Artifacts

TechAtBloomberg.com

Bloomberg

Engineering

# Current KFServing Status

- Have been running Istio/Knative stack for a year
- Started with Function as a Service on Knative/Istio, now migrating over to KFServing v0.2.1
- Starting to have users trying out on DEV; currently working on Production deployment process, aiming to get Production-ready for early next year





Bloomberg

Engineering

# Our Working Group is Open

- We couldn't have done it without the community.
- Steering committee from five different companies.
- Diverse perspectives lead to a better product.
- Weekly working group open to the public.
- Special topics meetings to discuss upcoming designs.

# Community Designs & The Future

- A Coherent Architecture for Prediction, Explainability, and Pre/Post Processing.
- Finalized V1 and driving conversations for a V2 Inference Protocol.
- Proposals for out of the box integration with Feature Stores.
- Active Discussions on Async and Batch Inferencing.
- Exploring a new CR for Ensembling, AB Testing, Multi Arm Bandits .
- Investigating TCO improvements via GPU sharing.
- Available in Kubeflow 0.7

# Come Help!

- website: https://kubeflow.org
- github: https://github.com/kubeflow/kfserving
- slack: kubeflow (http://kubeflow.slack.com)
- twitter: @kubeflow
- email: ellisbigelow@google.com, dsun20@bloomberg.net


# Thank You
# www.kubeflow.org