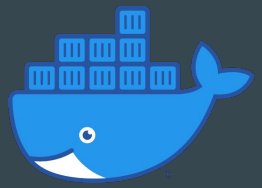
PyData

Gentle introduction to scaling up ML service with Kubernetes + Mlflow



PyData NYC, November 2022

Kei Nemoto

About me

- Kei Nemoto
- Data Scientist @ Montefiore Einstein Center for Health Data Innovations
  - MLOps
  - CI/CD
  - ETL
  - Backend Development
  - NLP

linkedin.com/in/kei-nemoto

github.com/box-key

All code is available on Github!

github.com/box-key/pydata-kubernetes-mlflow

ML Service Tech Stack

- Machine learning framework

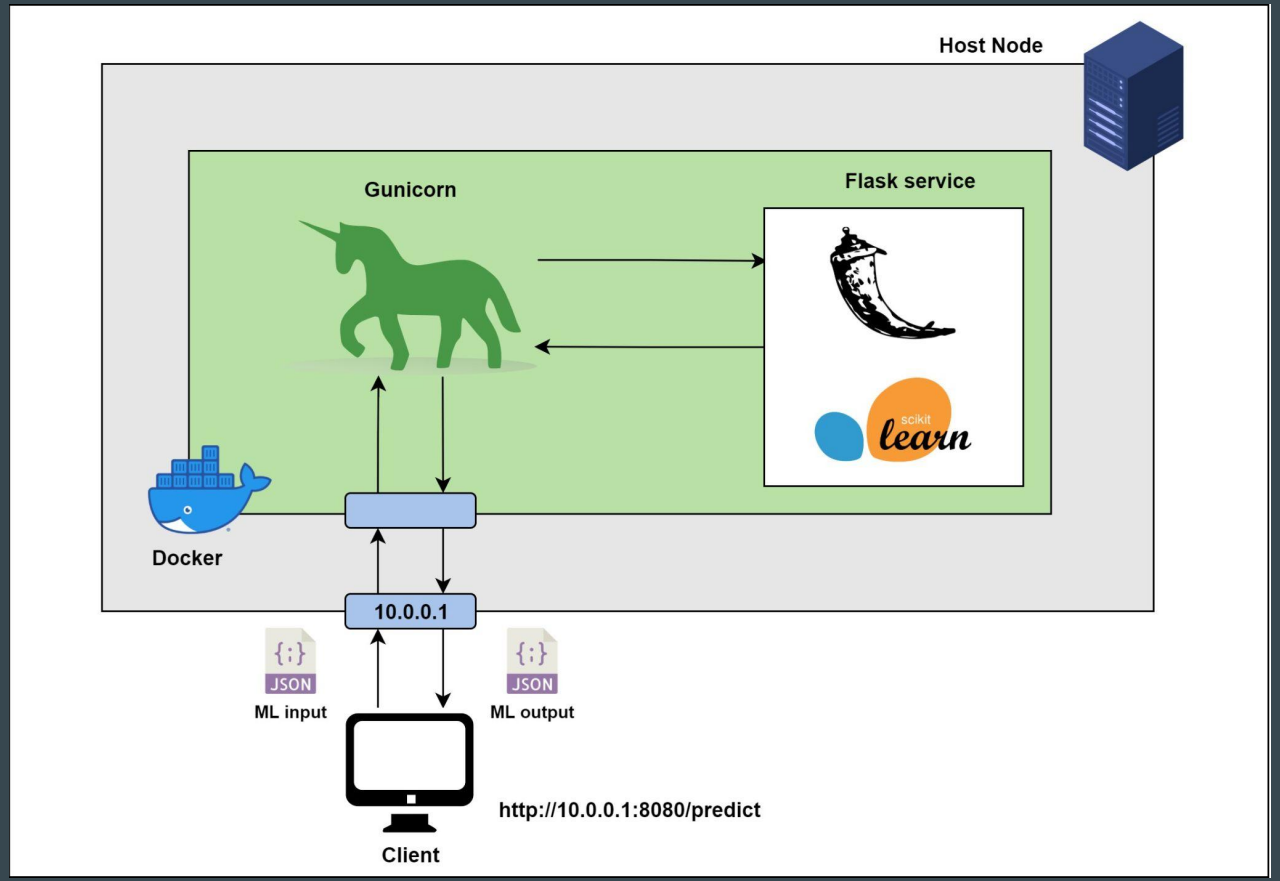
- Web framework

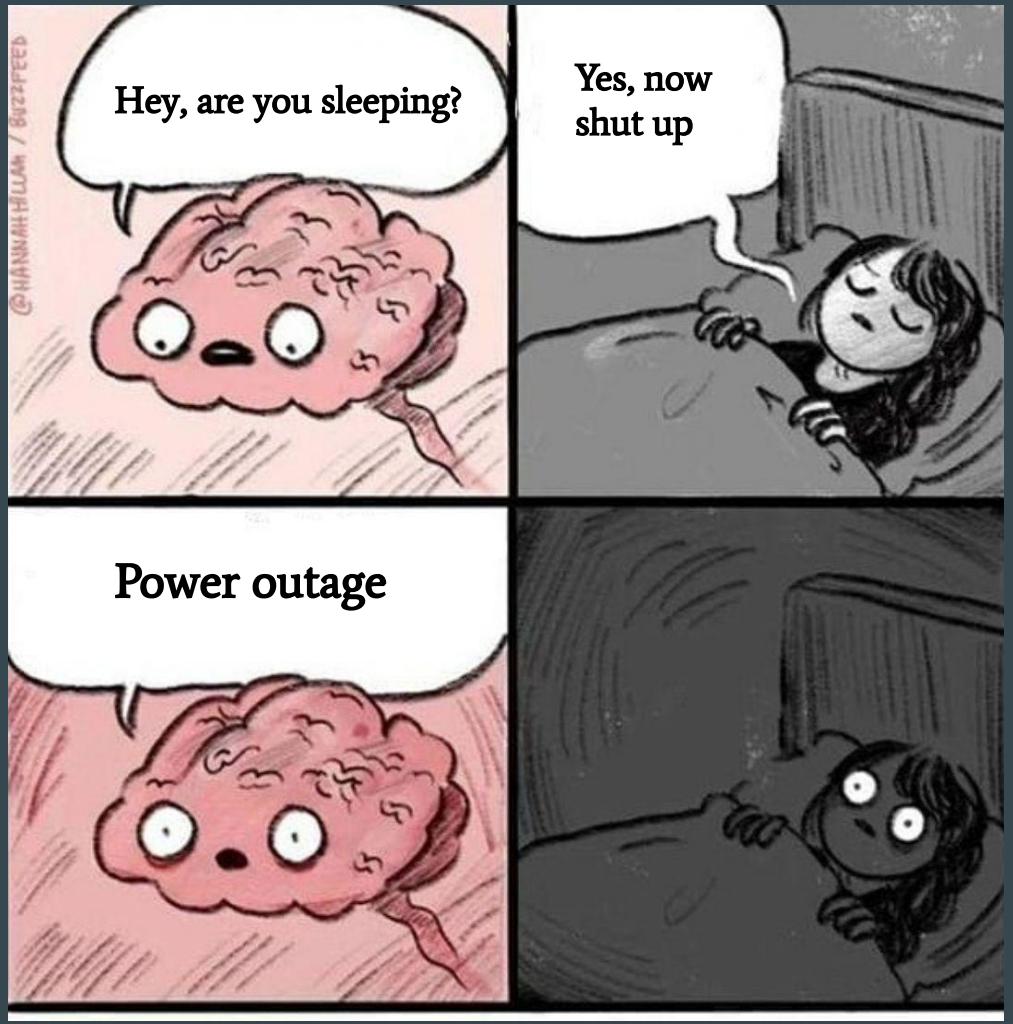
- WSGI Http server

- Container development tool

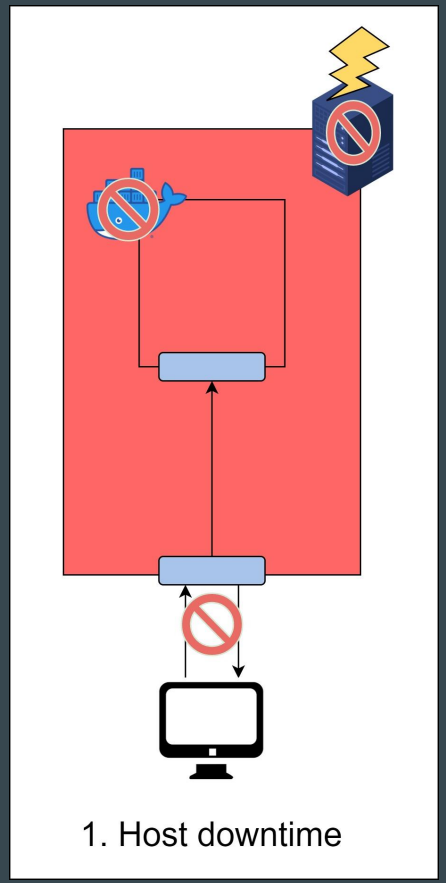
```python
from flask import Flask, request
from flask_restful import Resource, Api
from sklearn.ensemble import RandomForestClassifier


class IrisRandomForestPredictor(Resource):
    def __init__(self, **kwargs):
        self.model: RandomForestClassifier = kwargs["model"]

    def post(self):
        payload = request.get_json()
        input_arr = [
            float(payload["sepal_length"]),
            float(payload["sepal_width"]),
            float(payload["petal_length"]),
            float(payload["petal_width"])
        ]
        prediction = int(self.model.predict([input_arr])[0])
        return {
            "status": 200,
            "message": "Successfully predicted!",
            "data": {
                "prediction": prediction
            }
        }
```

Single Container Deployment

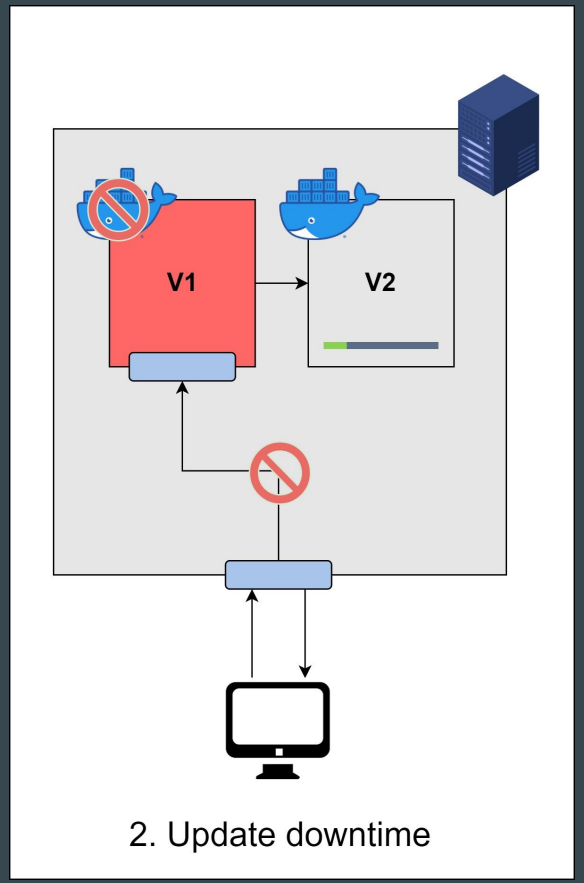



Hey, are you sleeping?
Yes, now shut up
Power outage

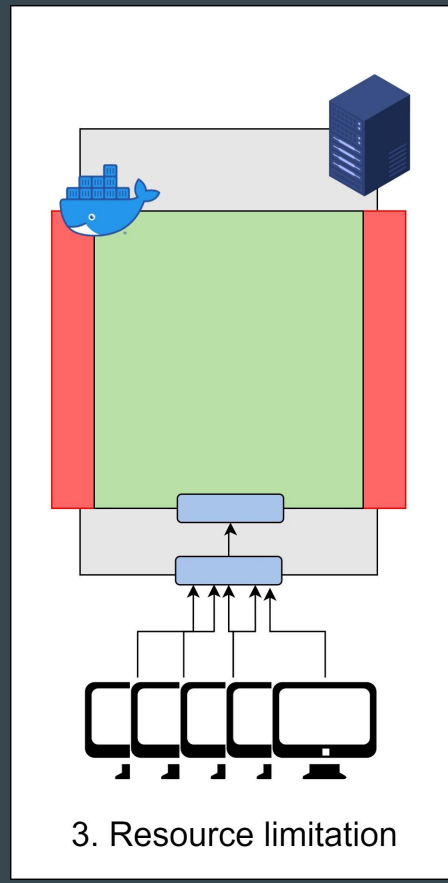
Problems of Single Container Deployment



1. Host downtime

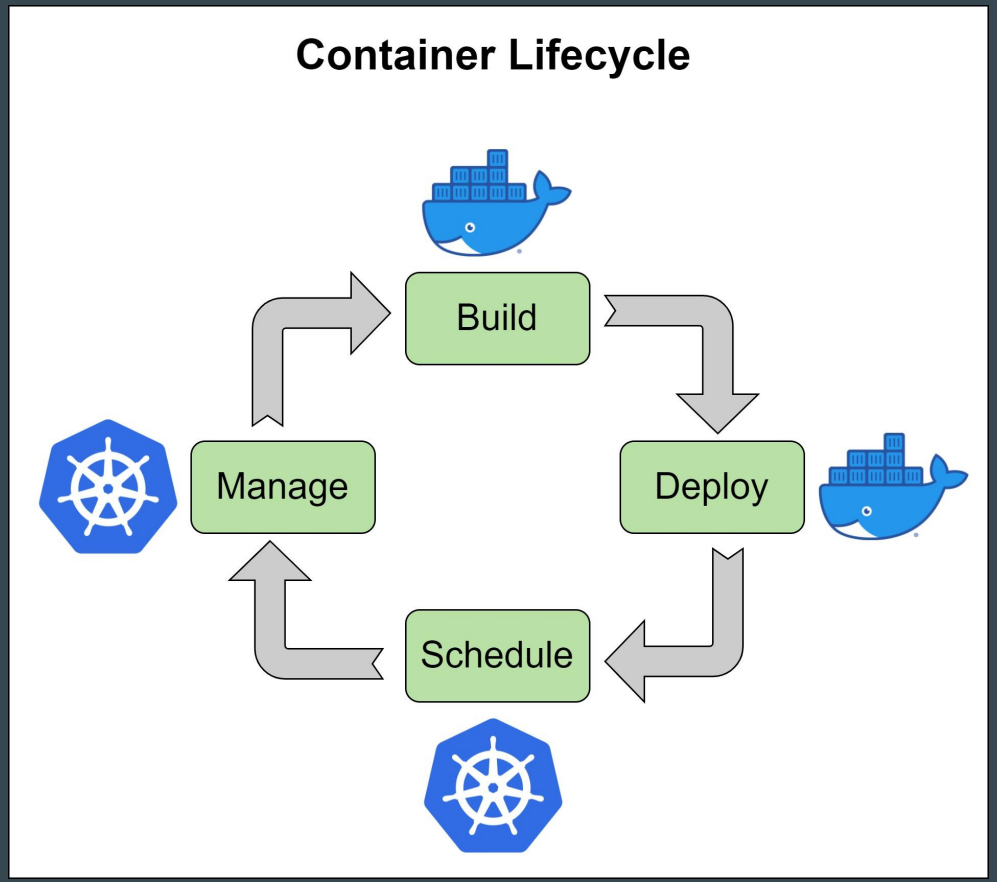


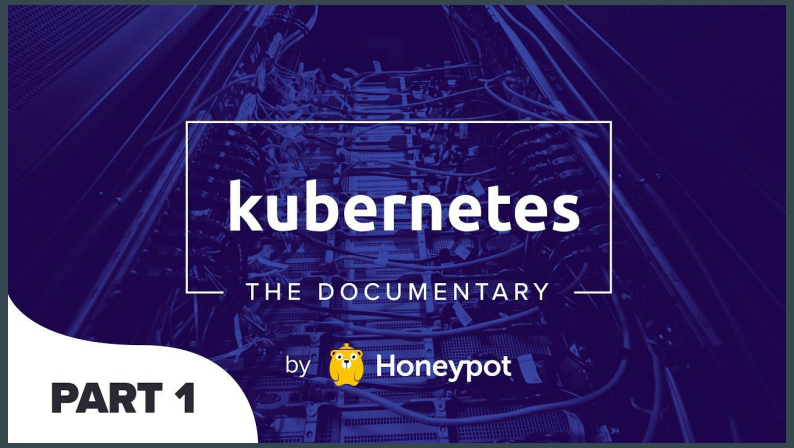
2. Update downtime



3. Resource limitation

Kubernetes = Container Management System

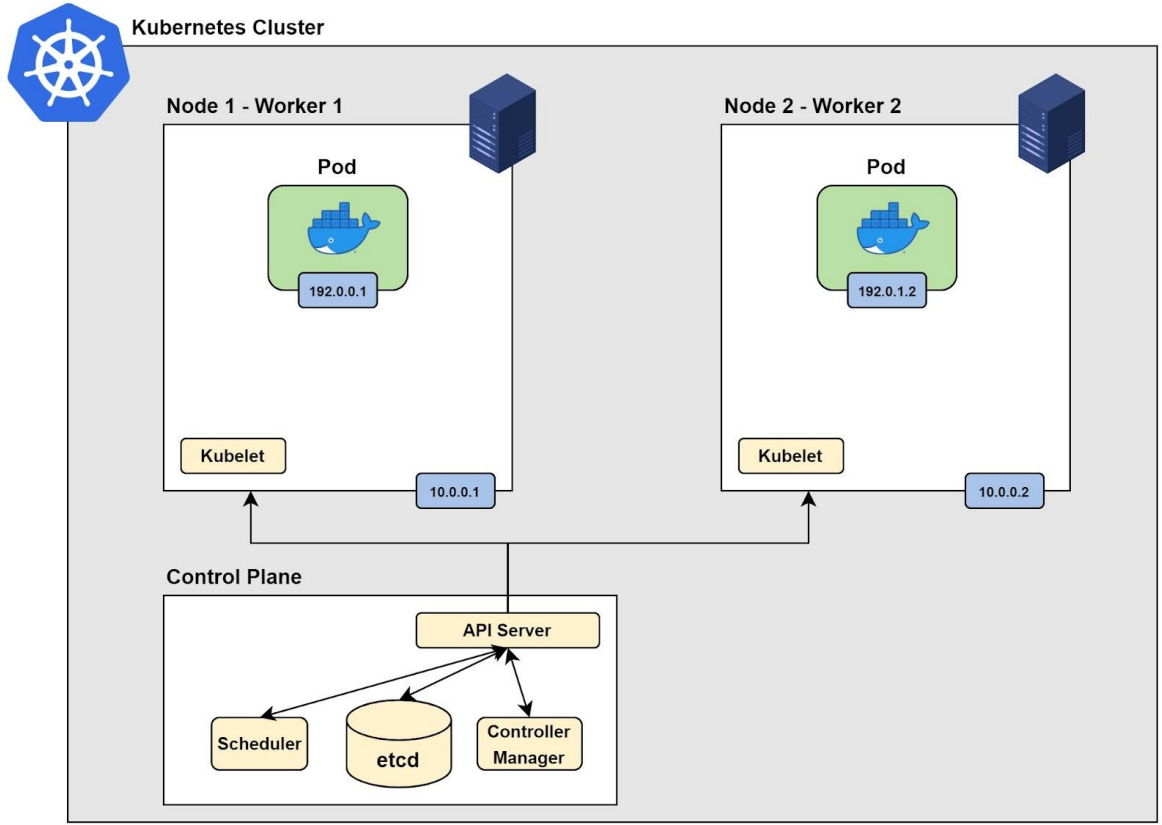




Let's invent the post office. Docker creates an envelope (container). Kubernetes is the postal system that delivers envelopes to the destination.

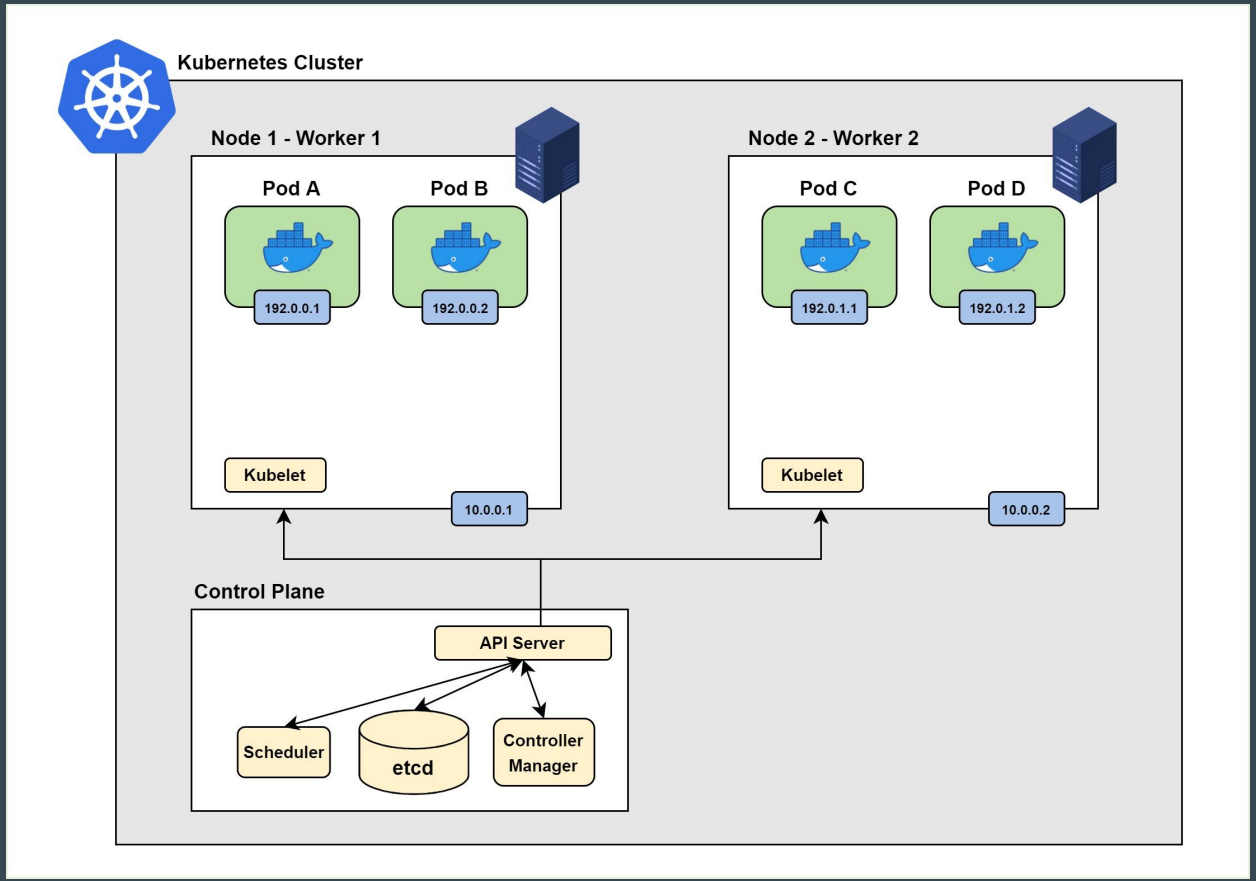
Reference: "Kubernetes: The Documentary [PART 1]" by Honeypot on Jan 21, 2022. https://www.youtube.com/watch?v=BE77h7dmoQU

Kubernetes 101

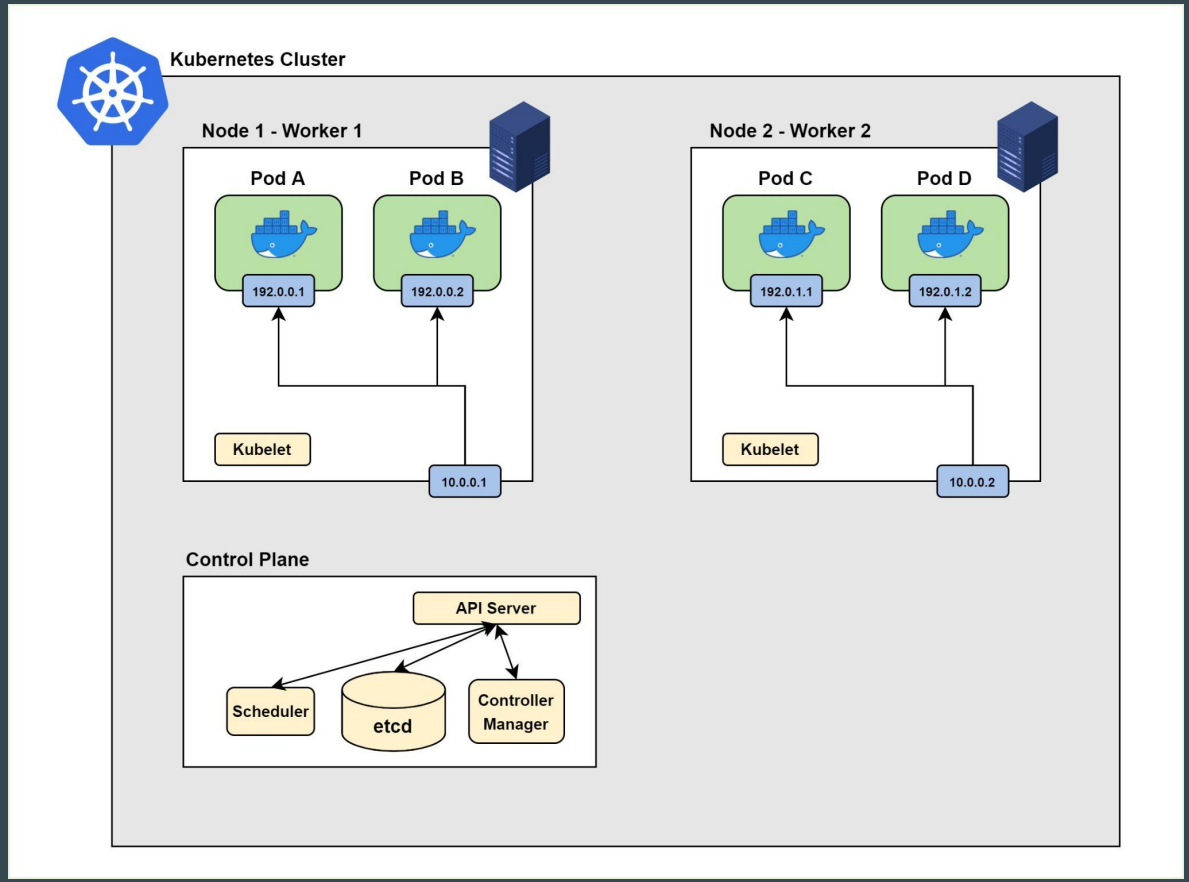


- Kubernetes cluster consists of *Control Plane* and *Workers*
- *Pod* executes container
- Each *pod* gets unique IP address

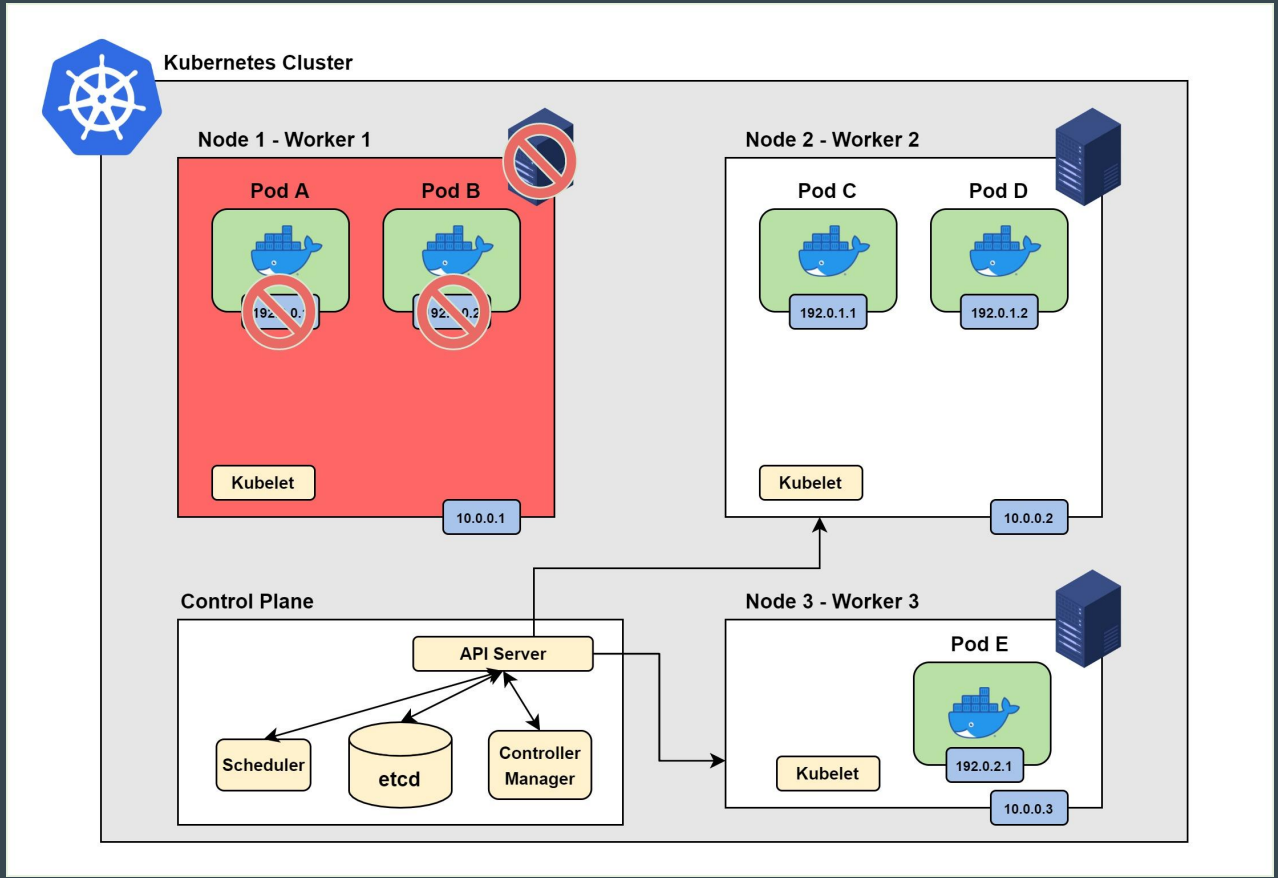
Deployment


Kubernetes Cluster
Node 1 - Worker 1
Pod A
Pod B
192.0.0.1
192.0.0.2
Kubelet
10.0.0.1
Node 2 - Worker 2
Pod C
Pod D
192.0.1.1
192.0.1.2
Kubelet
10.0.0.2
Control Plane
API Server
Scheduler
etcd
Controller Manager

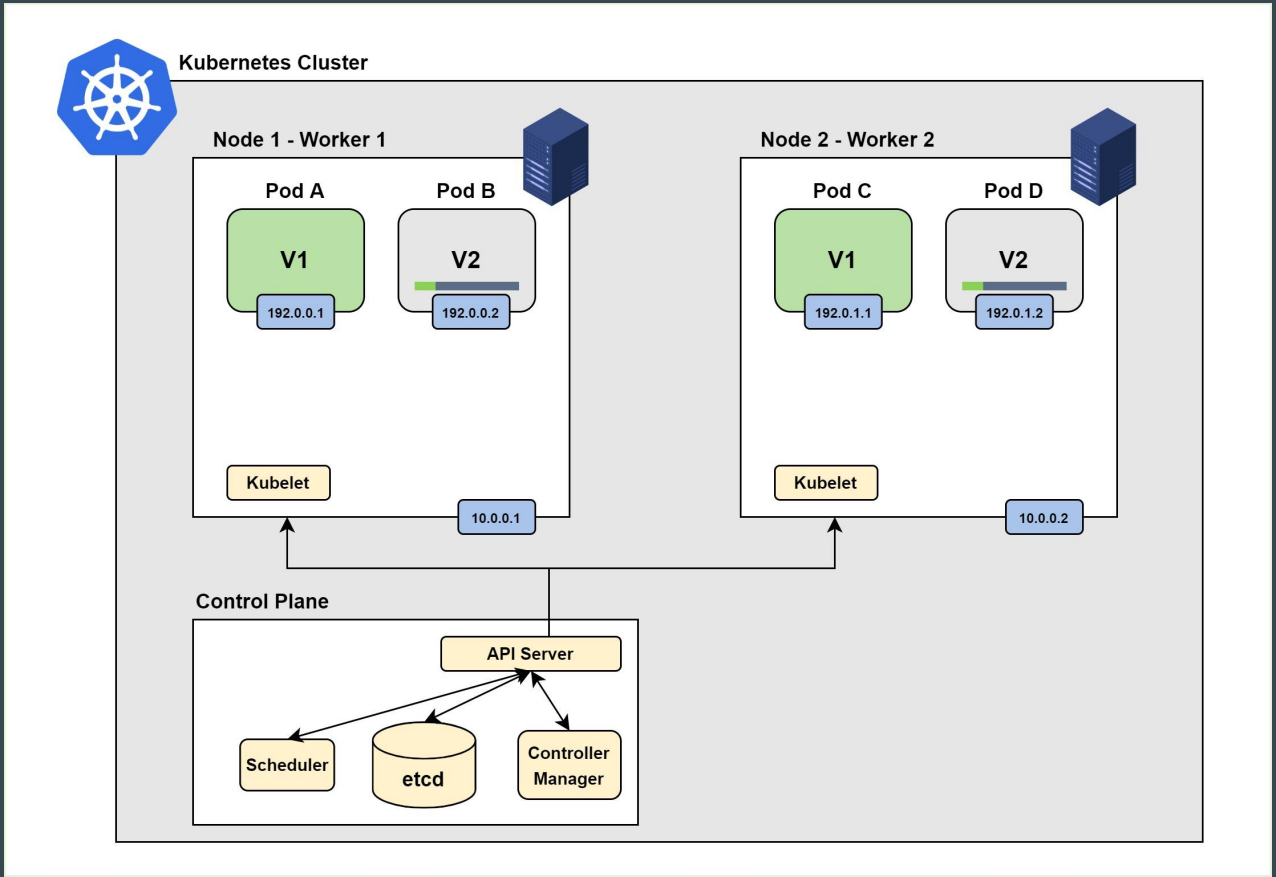
Deployment - Scalability



Deployment - High Availability



Deployment - Rolling update



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: iris-rf
  namespace: deployment-demo
  labels:
    app: iris-rf
spec:
  replicas: 3
  selector:
    matchLabels:
      app: iris-rf
  template:
    metadata:
      labels:
        app: iris-rf
    spec:
      containers:
        - name: iris-rf
          image: boxkey/iris-rf:beta
          ports:
            - containerPort: 5000
```

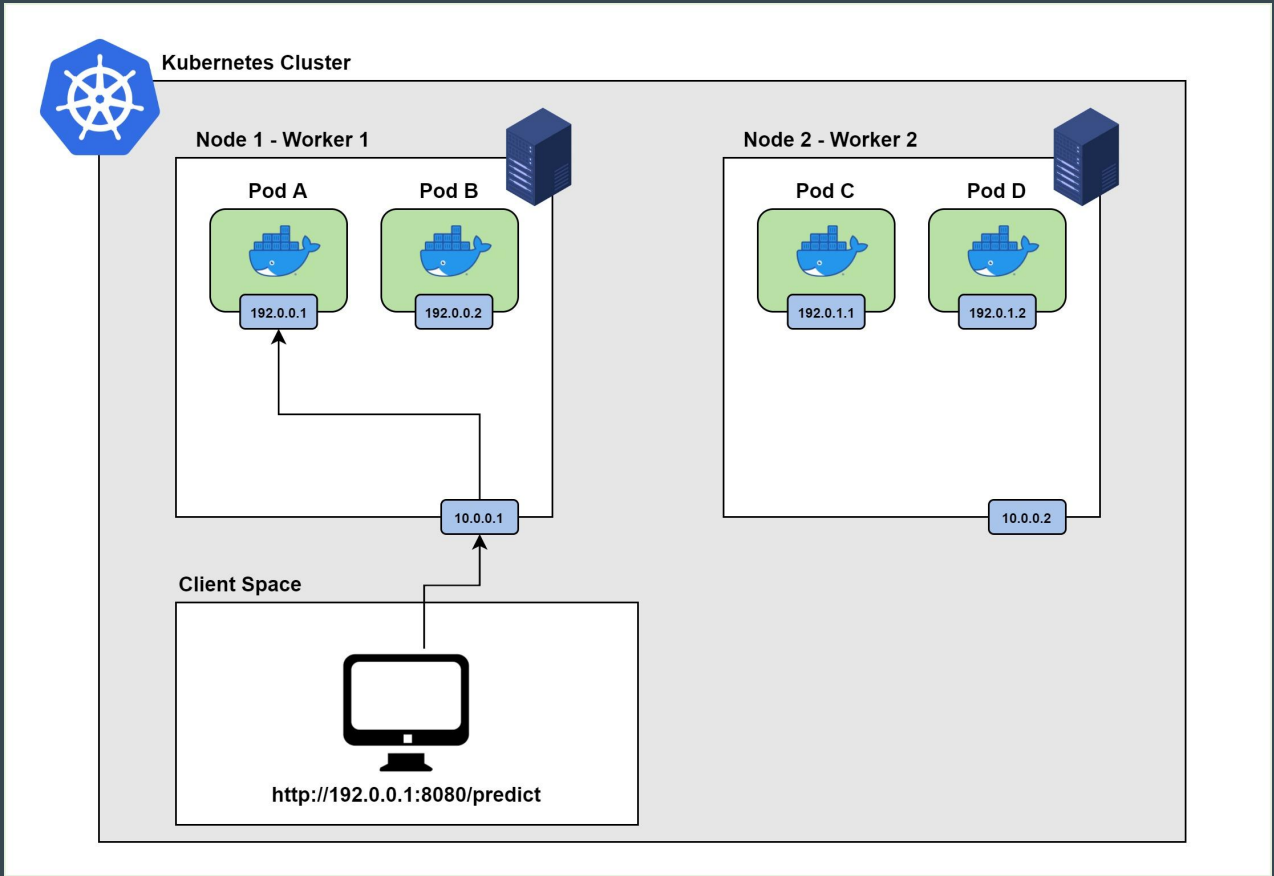
Create deployment

```
> kubectl apply -f kubernetes/deployment-demo.yaml


deployment.apps/iris-rf created
```

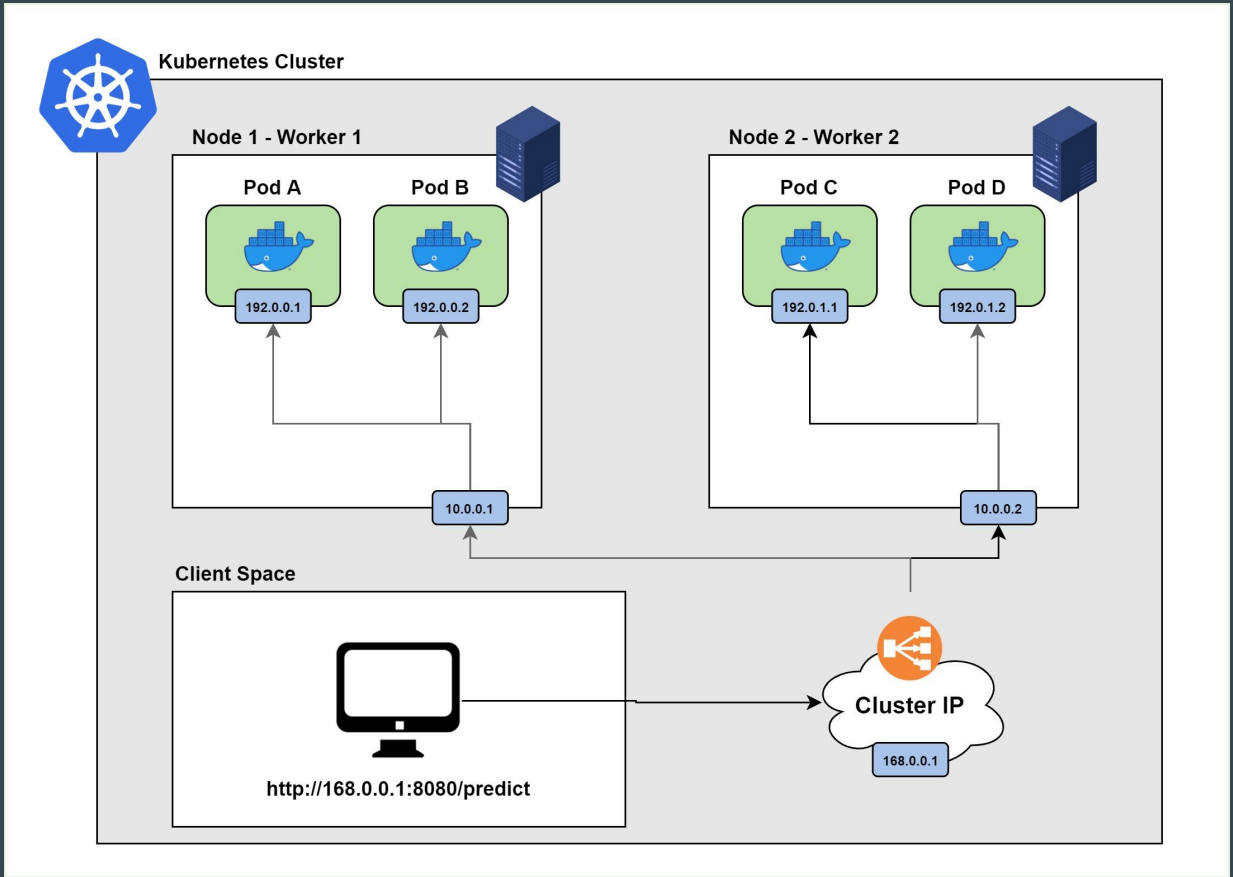
Print pods in deployment

```
> kubectl get pods -n deployment-demo \
> -o=custom-columns=NAME:.metadata.name,\
> IP:.status.podIP,STATUS:.status.phase,\
> IMAGE:.spec.containers[0].image


NAME                      IP           STATUS    IMAGE
iris-rf-8c7857fd6-v52qc   10.42.1.80   Running   boxkey/iris-rf:beta
iris-rf-8c7857fd6-s9dmp   10.42.0.85   Running   boxkey/iris-rf:beta
iris-rf-8c7857fd6-kgx72   10.42.2.70   Running   boxkey/iris-rf:beta
```

Which pod to send request to?



Cluster IP Service

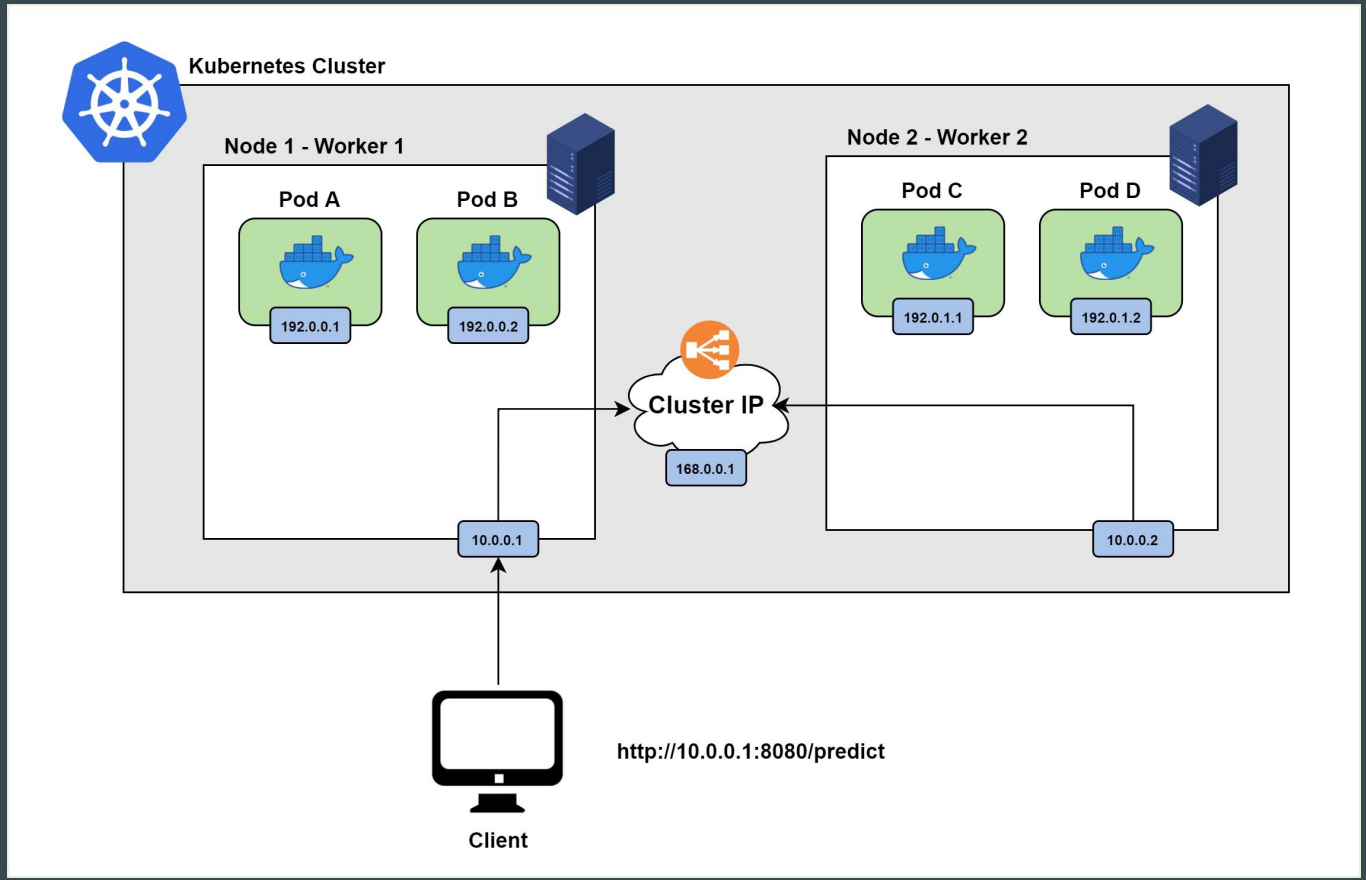


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: iris-rf
  namespace: cluster-ip-demo
  labels:
    app: iris-rf
spec:
  replicas: 3
  selector:
    matchLabels:
      app: iris-rf
  template:
    metadata:
      labels:
        app: iris-rf
    spec:
      containers:
        - name: iris-rf
          image: boxkey/iris-rf:beta
          ports:
            - containerPort: 5000
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: iris-svc
  namespace: cluster-ip-demo
spec:
  ports:
    - port: 5000
      protocol: TCP
      targetPort: 5000
  selector:
    app: iris-rf
  type: ClusterIP
```

```
> kubectl get services -n cluster-ip-demo

NAME       TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
iris-svc   ClusterIP   10.43.251.45   <none>        5000/TCP   26s
```

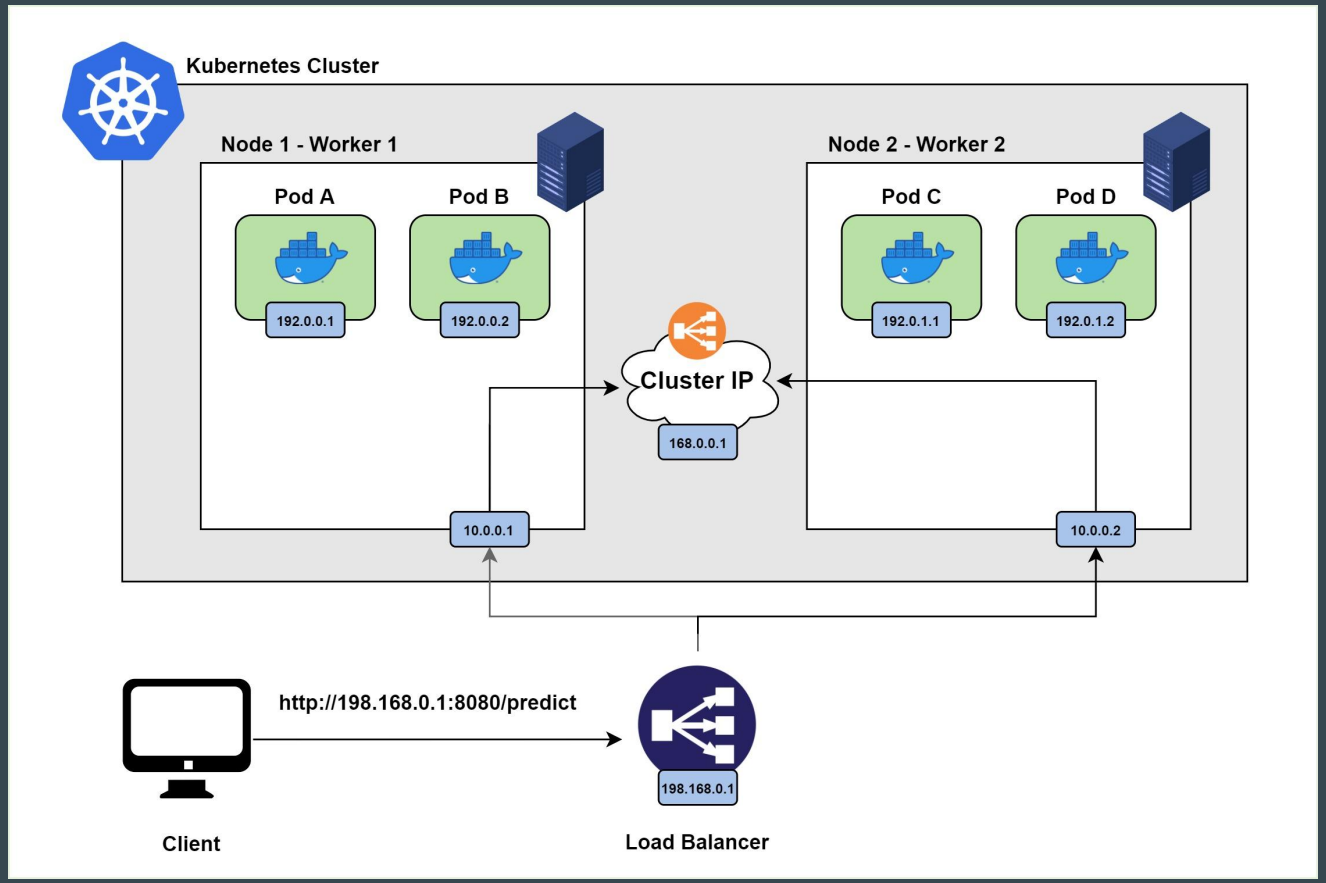
NodePort Service



```yaml
apiVersion: v1
kind: Service
metadata:
  name: iris-svc
  namespace: node-port-demo
spec:
  ports:
    - port: 5000
      protocol: TCP
      targetPort: 5000
  selector:
    app: iris-rf
  type: NodePort
```

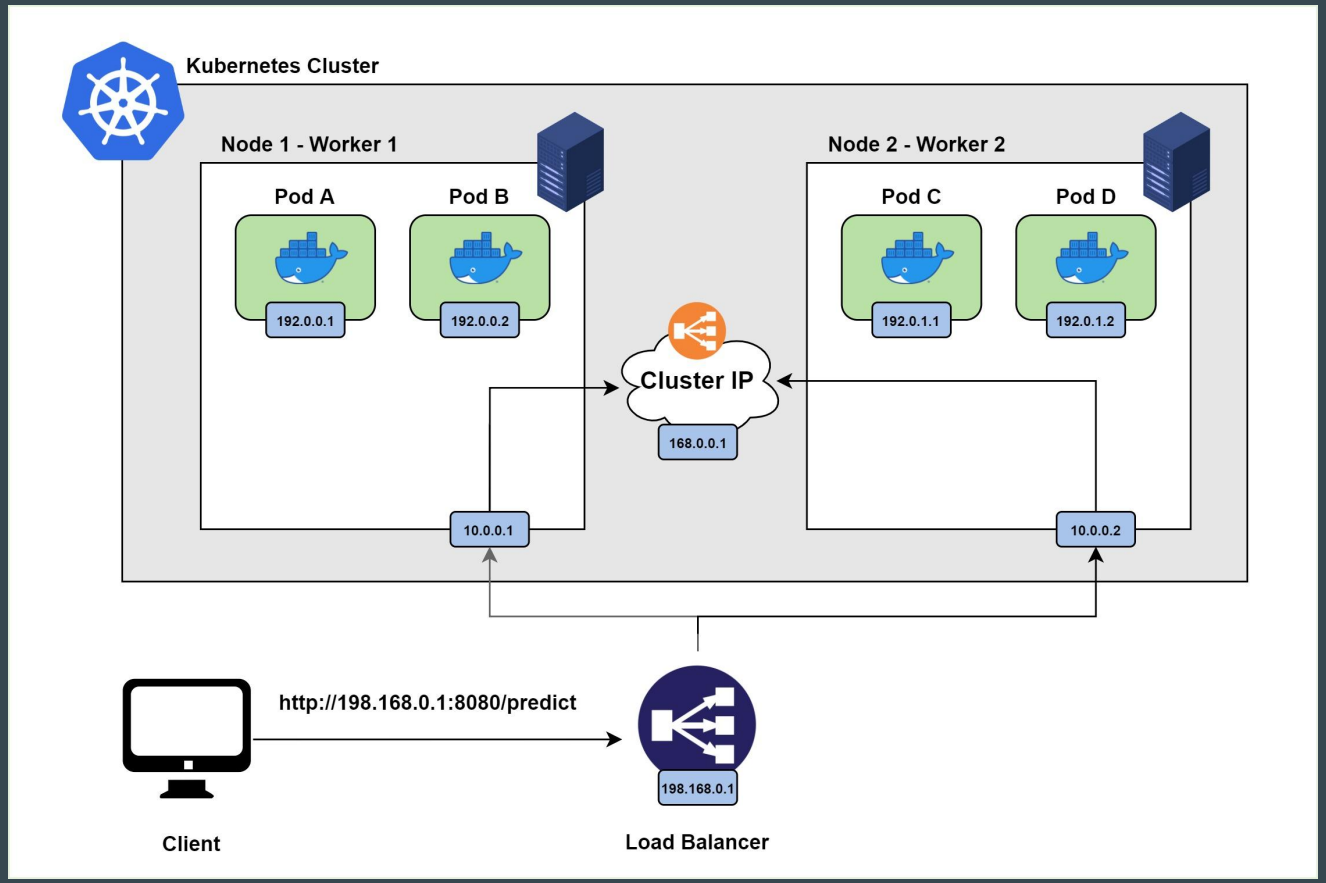
```
> kubectl get services -n node-port-demo

NAME        TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
iris-svc    NodePort    10.43.11.78   <none>        5000:31304/TCP   4h48m
```

LoadBalancer Service

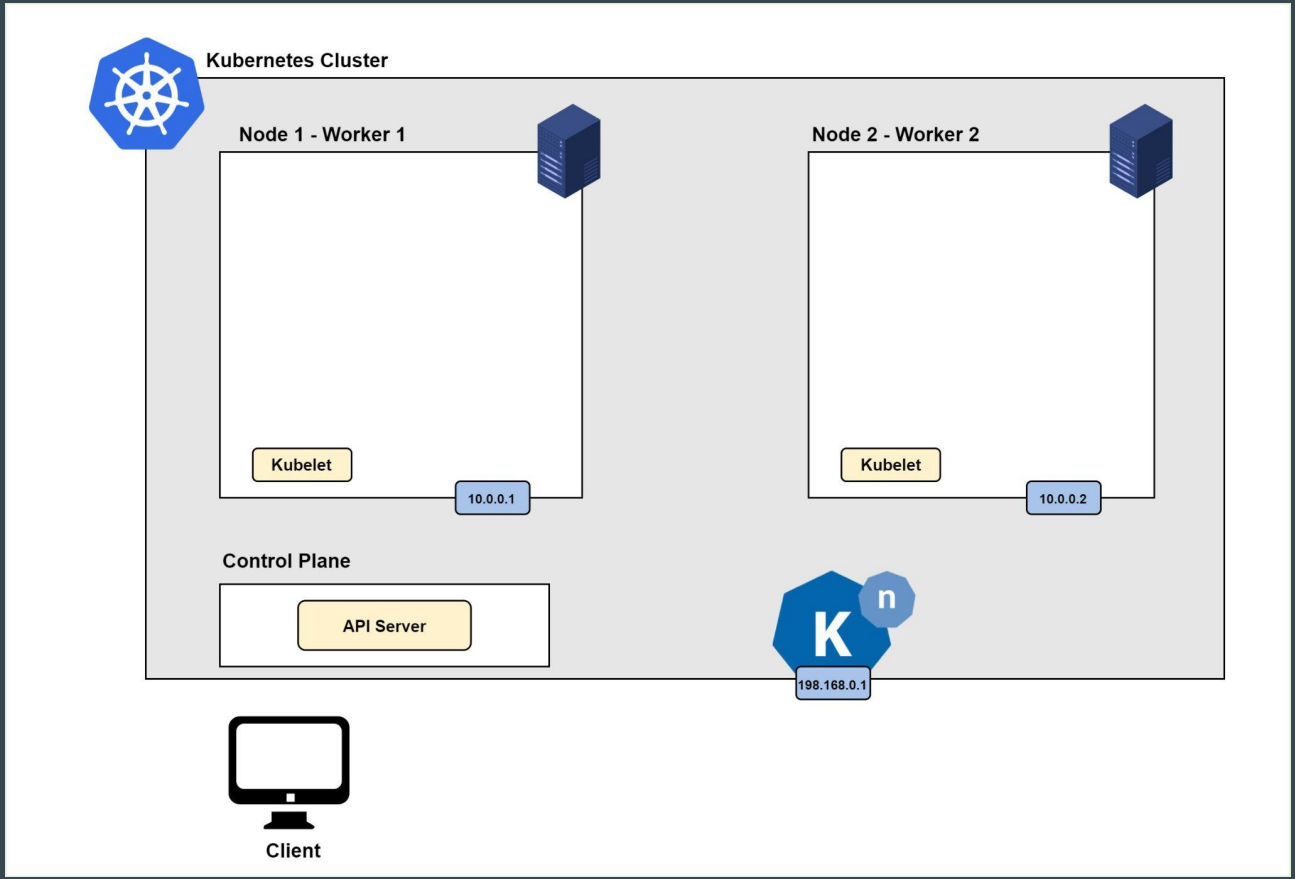


```yaml
apiVersion: v1
kind: Service
metadata:
  name: iris-svc
  namespace: load-balancer-demo
spec:
  ports:
    - port: 5000
      protocol: TCP
      targetPort: 5000
  selector:
    app: iris-rf
  type: LoadBalancer
```

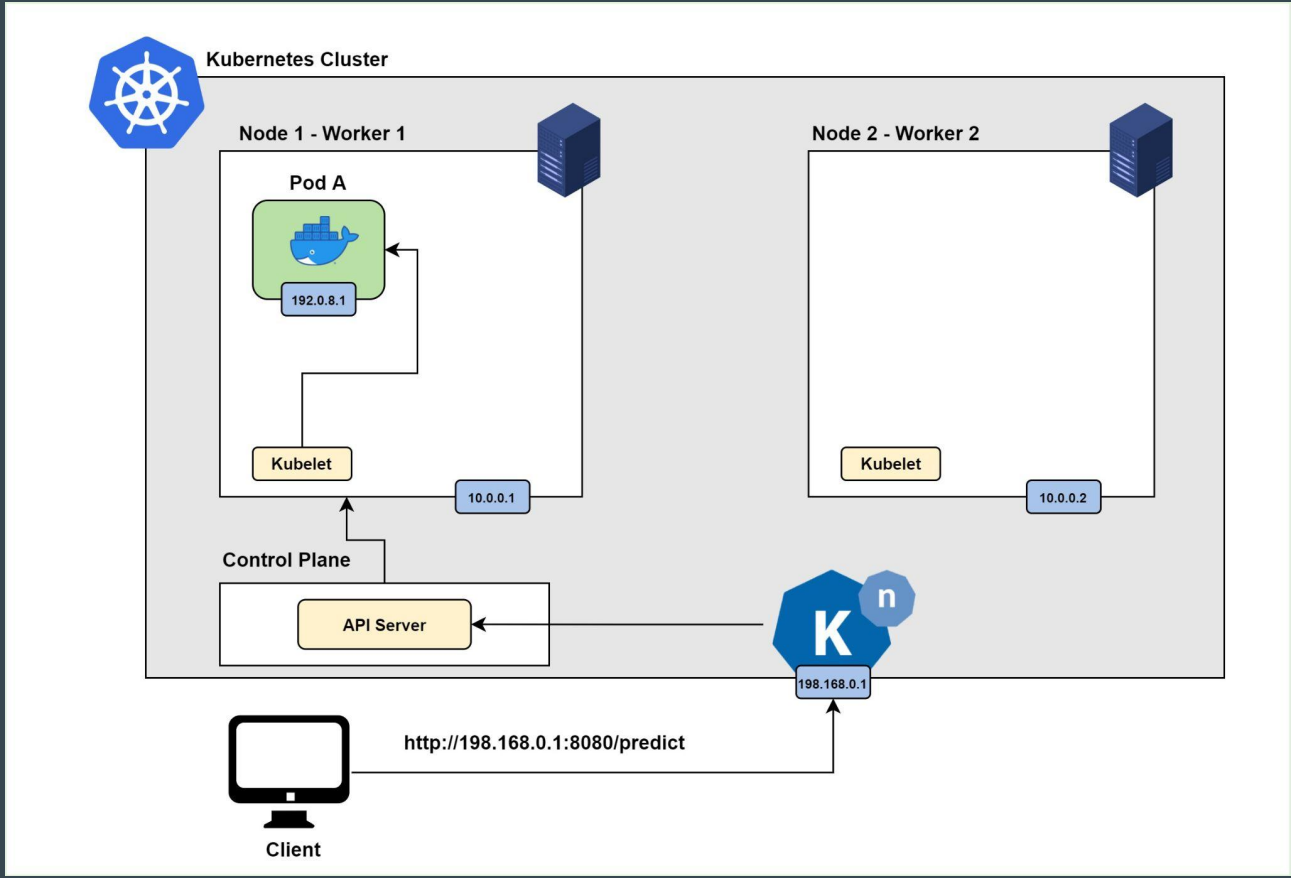
Should we run 4 pods?



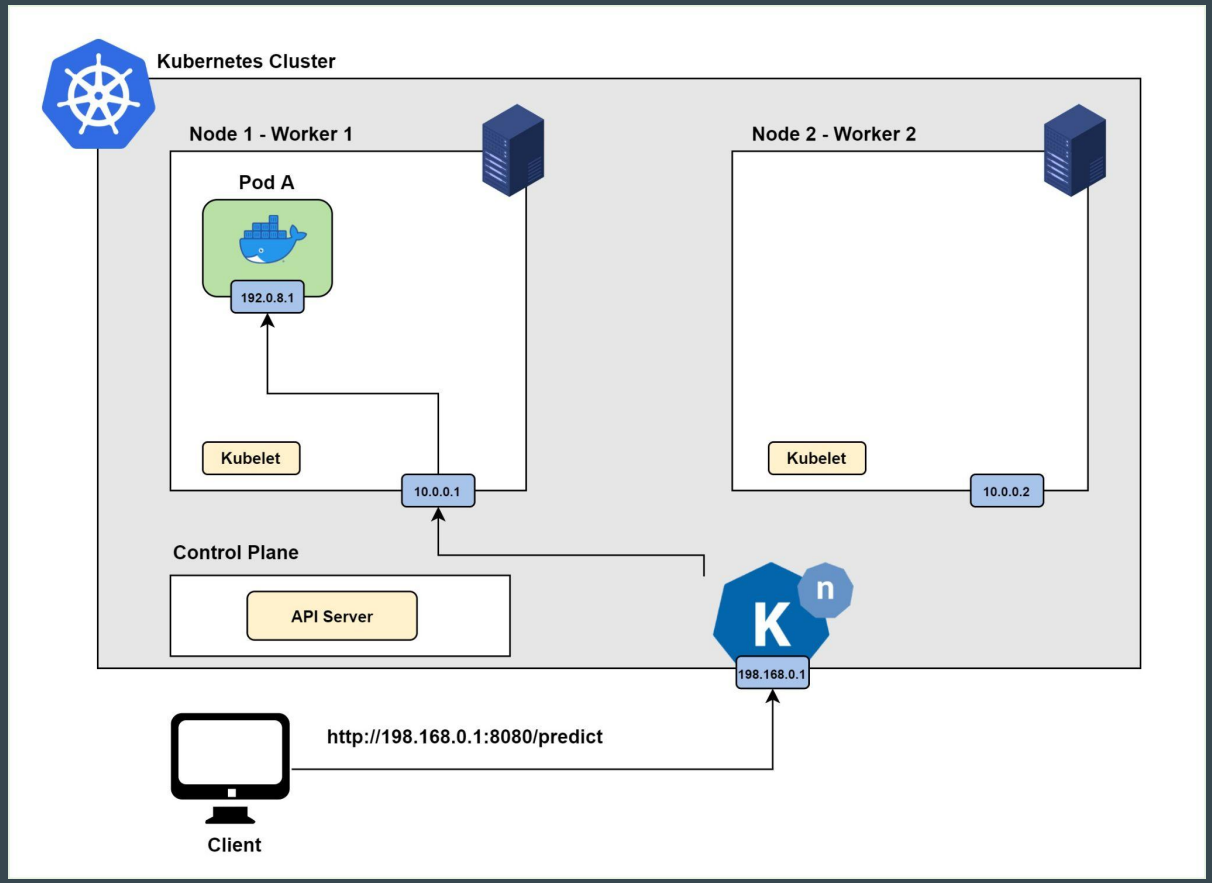
Serverless Service by Knative



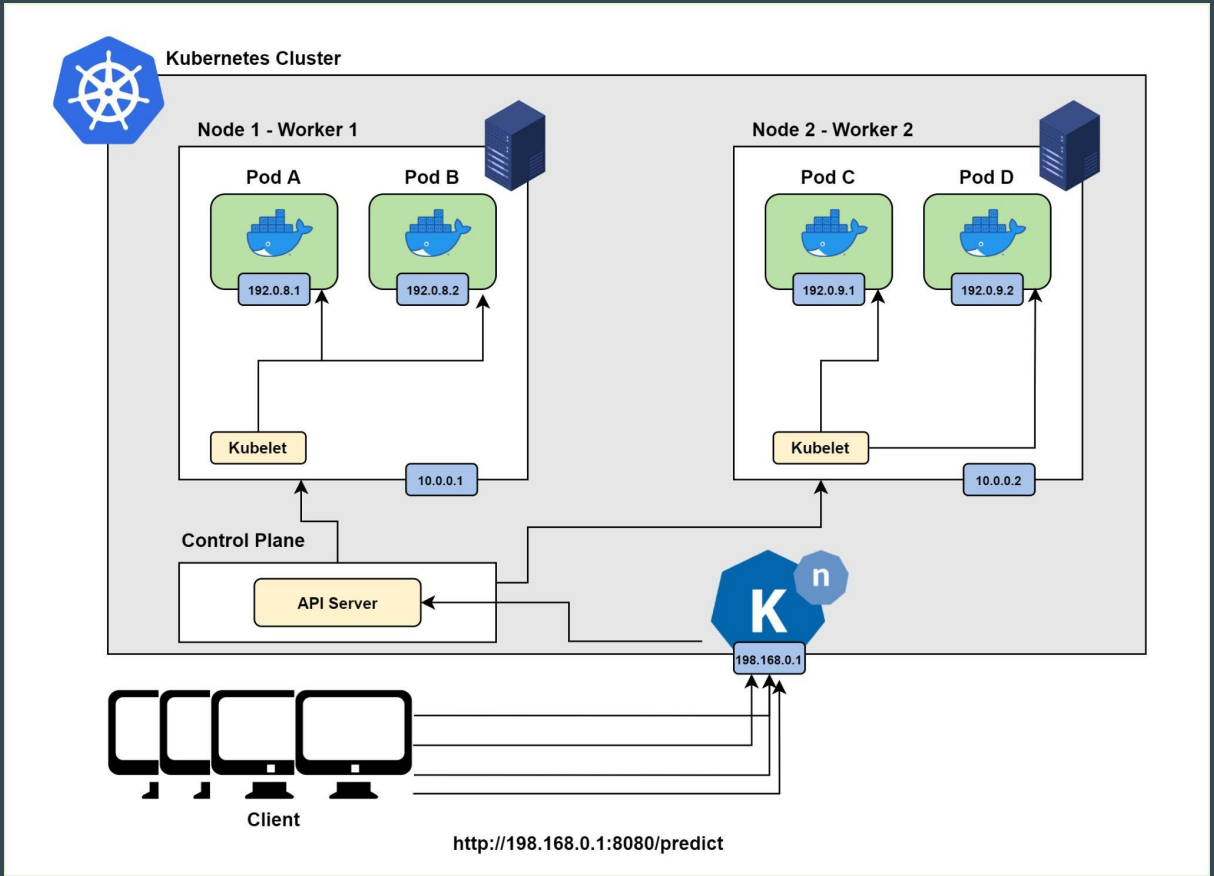
Serverless Service by Knative - Pod Creation



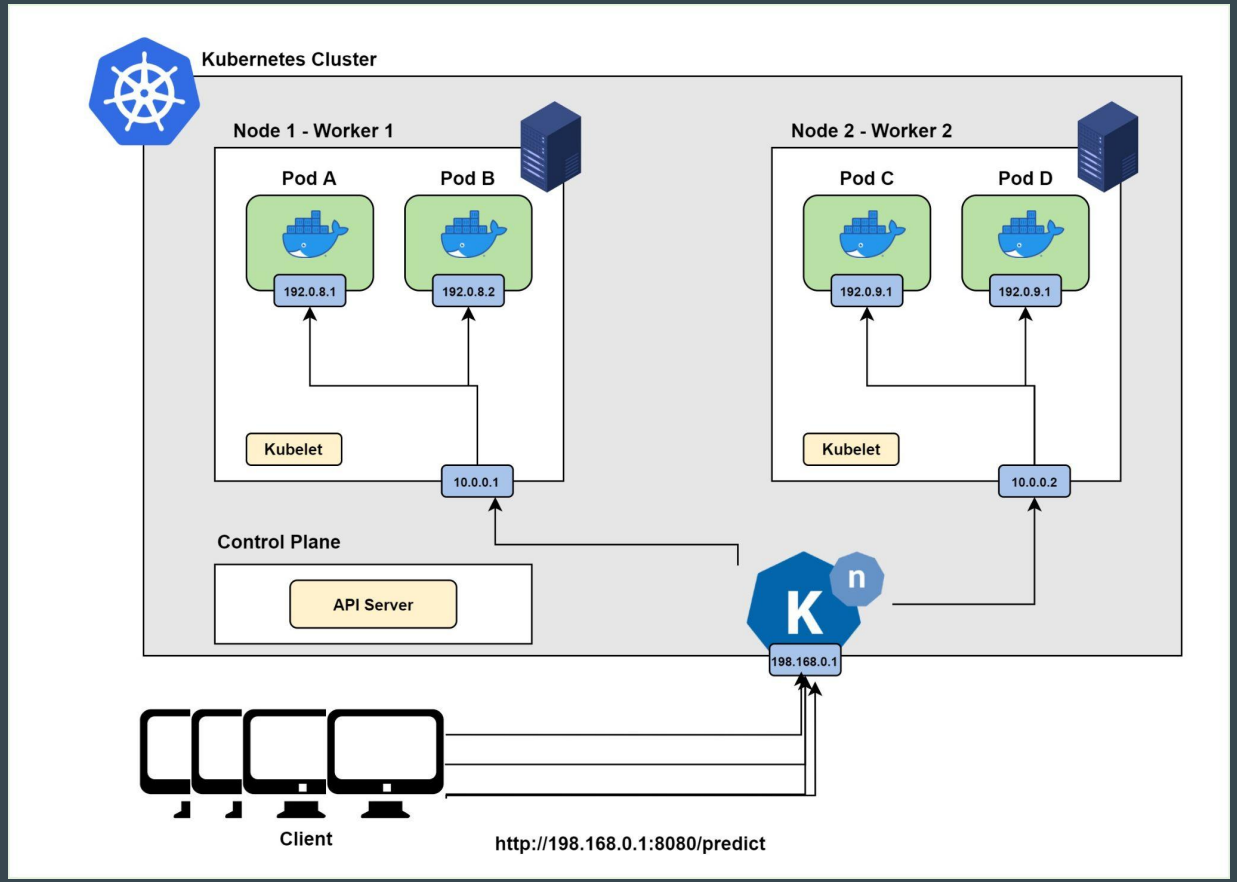
Serverless Service by Knative - Forward request

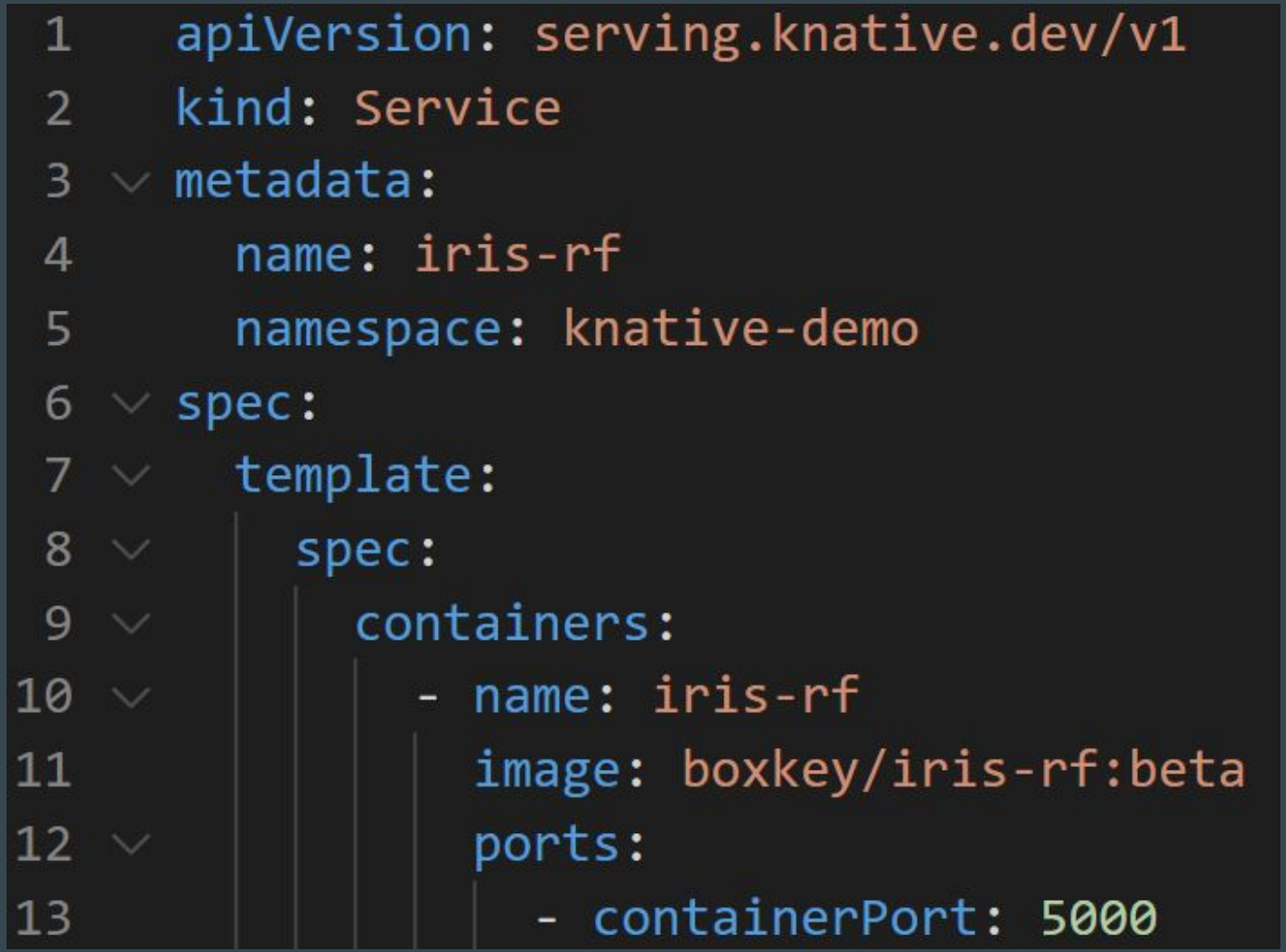


Automatic Scaling - Pod Creation

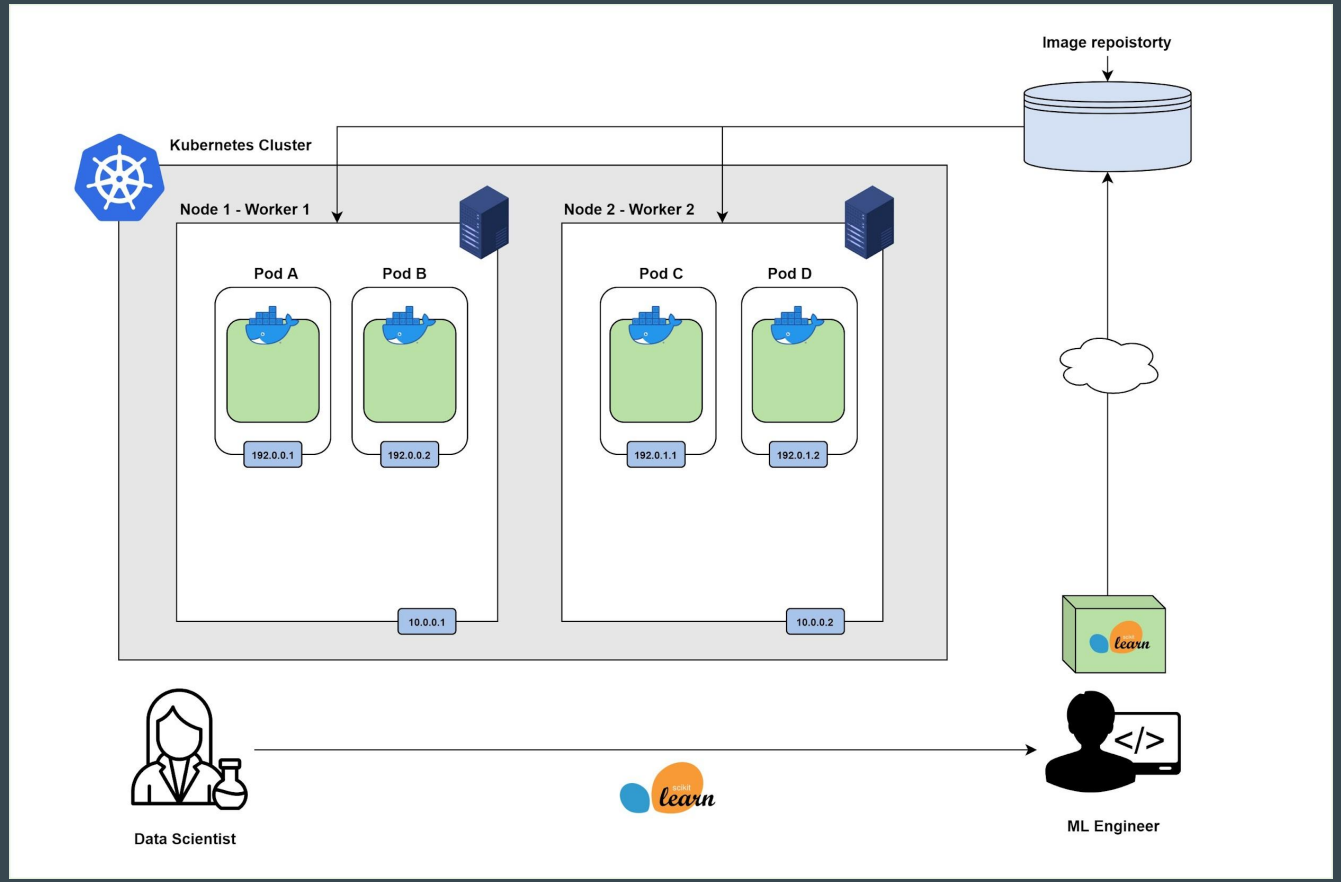


Automatic Scaling - Distribute Requests

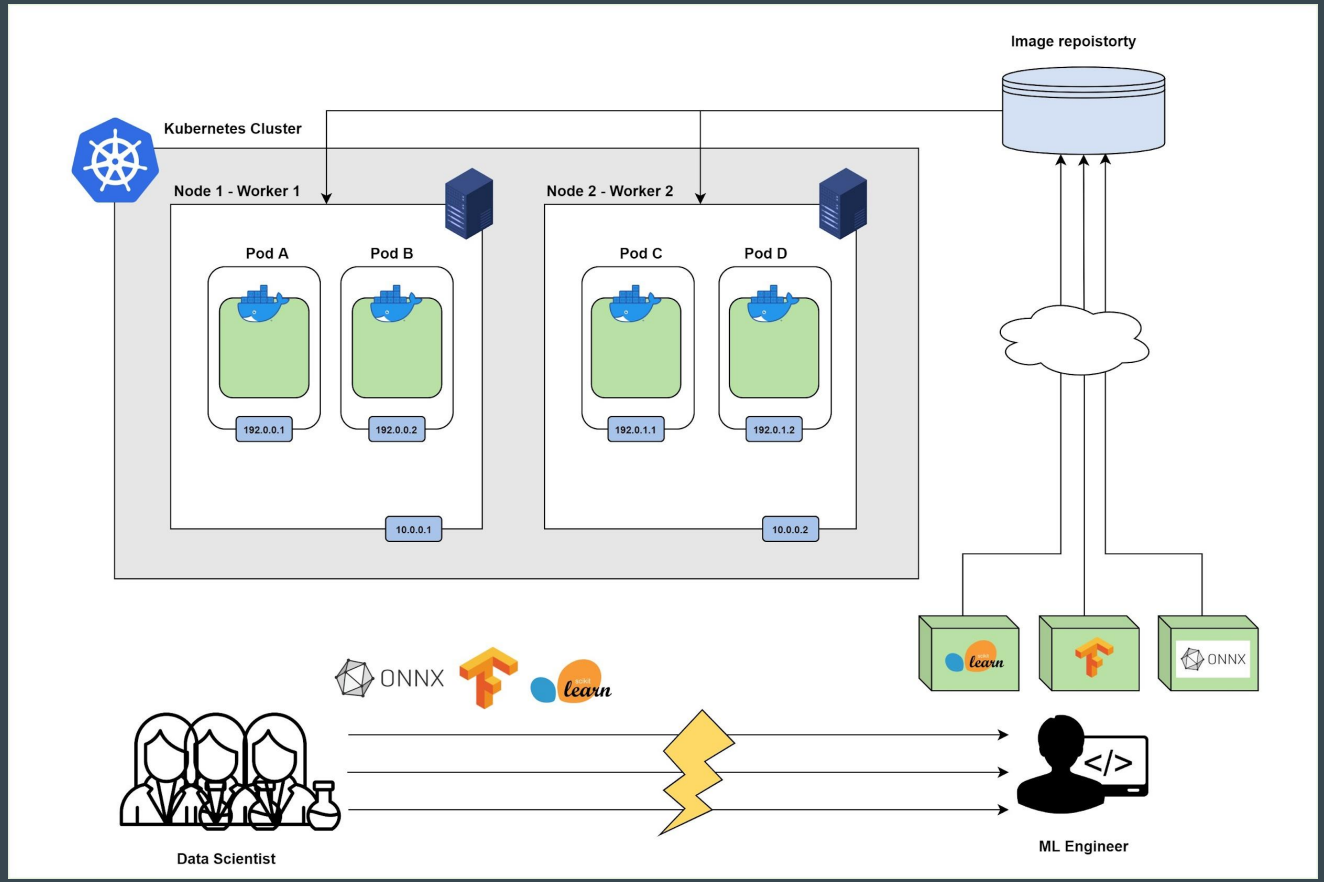



apiVersion: serving.knative.dev/v1
kind: Service
metadata:
name: iris-rf
namespace: knative-demo
spec:
template:
spec:
containers:
- name: iris-rf
image: boxkey/iris-rf:beta
ports:
- containerPort: 5000

Model in Kubernetes

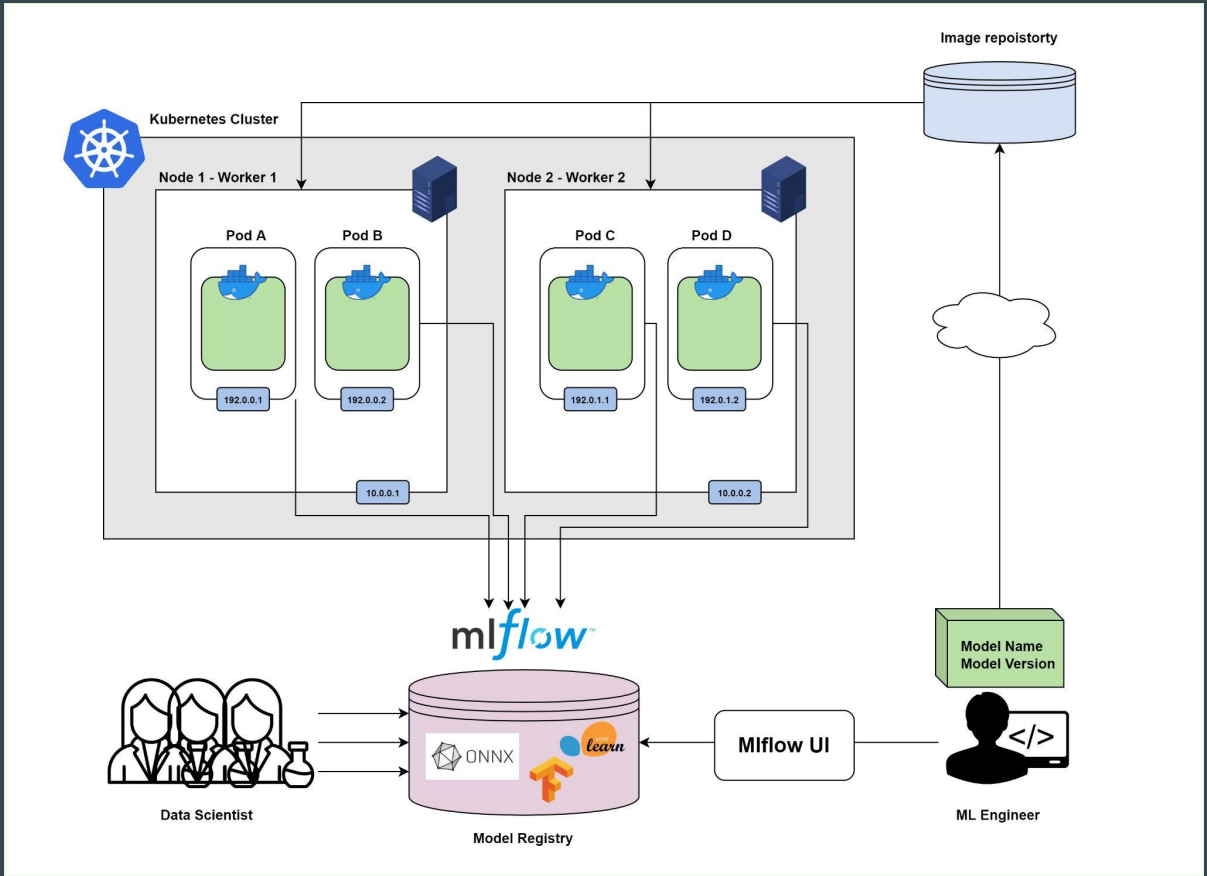


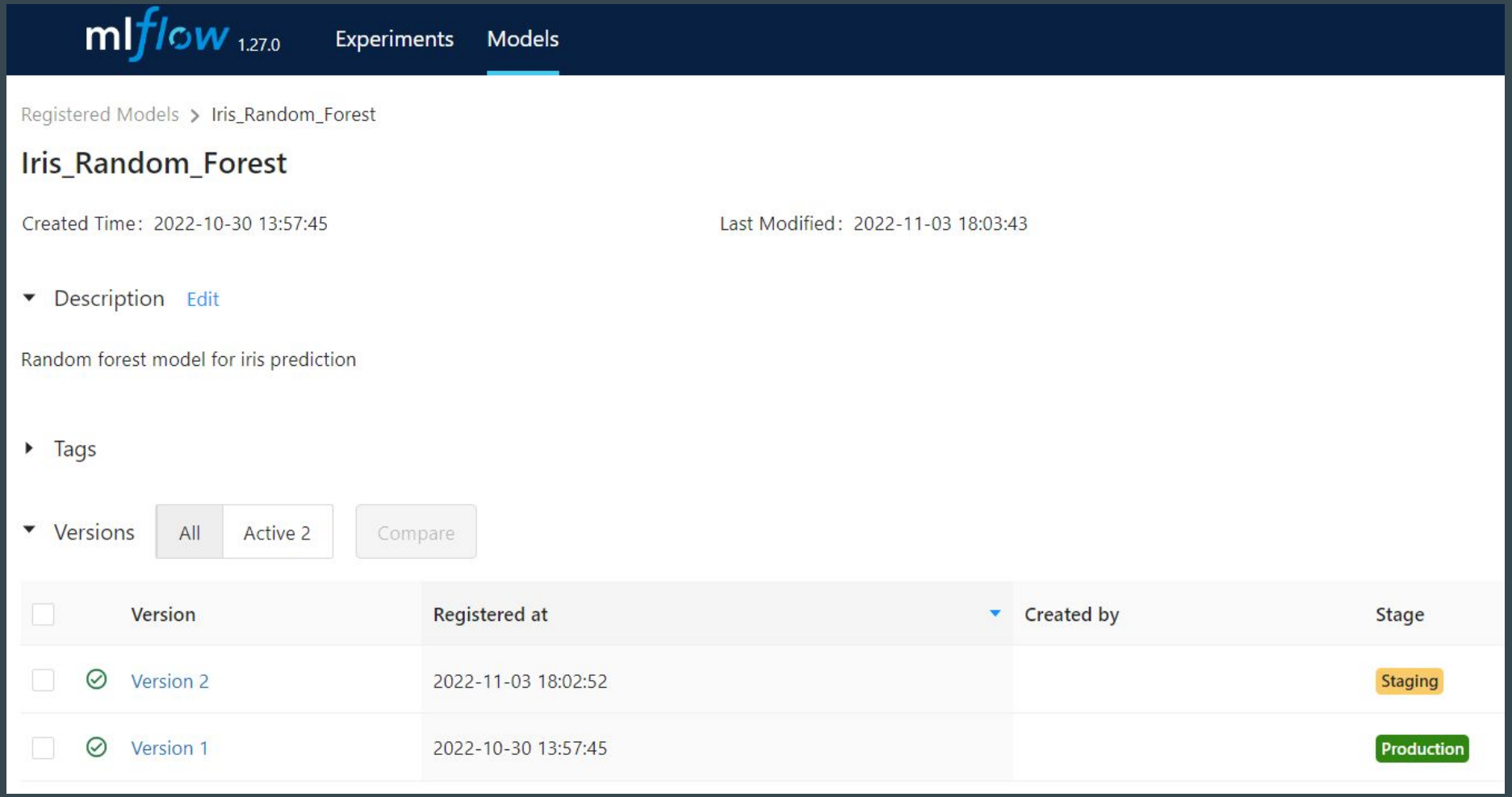
Disconnection between Data Scientist and Developer




We're using AI instead of biased humans
What did you train the AI on?
What did you train the AI on?
imgflip.com

Mlflow Model Registry + Kubernetes




mlflow 1.27.0
Experiments
Models
Registered Models > Iris_Random_Forest
Iris_Random_Forest
Created Time: 2022-10-30 13:57:45
Last Modified: 2022-11-03 18:03:43
Description
Edit
Random forest model for iris prediction
Tags
Versions
All
Active 2
Compare
Version
Registered at
Created by
Stage
Version 2
2022-11-03 18:02:52
Staging
Version 1
2022-10-30 13:57:45
Production

Register Model in Mlflow

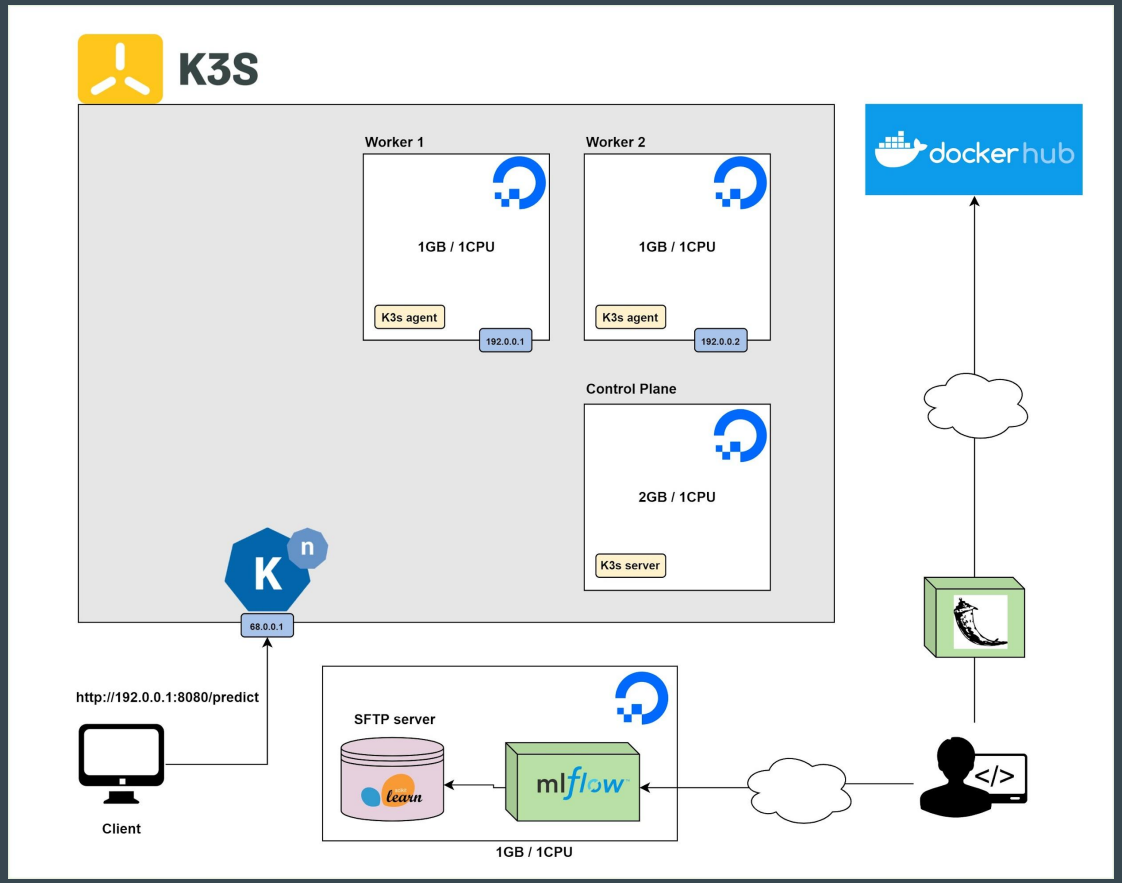
```
17  # train rf classifier
18  logger.info("Training random forest model..")
19  iris = datasets.load_iris()
20  clf = rf()
21  clf.fit(iris.data, iris.target)
22  # save model in mlflow
23  logger.info("Saving trained model in mlflow..")
24  name = "iris-rf"
25  mlflow_uri = os.getenv("MLFLOW_URI")
26  artifact_location = os.getenv("ARTIFACT_LOCATION")
27  mlflow.set_tracking_uri(mlflow_uri)
28  # create experiment if not exist
29  if mlflow.get_experiment_by_name(name) is None:
30      mlflow.create_experiment(
31          name, artifact_location=artifact_location
32      )
33  mlflow.set_experiment(name)
34  mlflow.sklearn.log_model(clf, artifact_path="sk_models")
35  logger.info("All done!")
36
```

Load Model for inference

```
44  def create_app():
45      import mlflow
46      import os
47
48      # get model name and version
49      model_name = os.getenv("MLFLOW_MODEL_NAME")
50      model_version = os.getenv("MLFLOW_MODEL_VERSION")
51      mlflow_uri = os.getenv("MLFLOW_URI")
52      # load model
53      logger.info(f"Loading {model_name=} | {model_version=}")
54      mlflow.set_tracking_uri(mlflow_uri)
55      model = mlflow.sklearn.load_model(
56          f"models:/{model_name}/{model_version}"
57      )
58      # init flask
59      app = Flask(__name__)
60      api = Api(app)
61      api.add_resource(
62          IrisRandomForestPredictor,
63          "/predict",
64          resource_class_kwargs={"model": model},
65      )
66      logger.info("Service ready!")
67      return app
```

```yaml
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
  name: iris-rf
  namespace: knative-demo
spec:
  template:
    spec:
      containers:
        - name: iris-rf
          image: boxkey/iris-rf:beta
          ports:
            - containerPort: 5000
          env:
            - name: MLFLOW_MODEL_NAME
              value: "Iris_Random_Forest"
            - name: MLFLOW_MODEL_VERSION
              value: "1"
            - name: MLFLOW_URI
              valueFrom:
                secretKeyRef:
                  name: mlflow
                  key: mlflowuri
                  optional: false
```

Kubernetes Setup - $30/month



Thank you!