

Introduction to Machine Learning Engineering and MLOps



1

About me, About Data Max

Bujar Bakiu, CEO Data Max

2

Hidden Technical Debt in Machine Learning

Hidden Technical Debt in ML, and why it is an issue

3

Modern ML Platforms, MLOps

Components of ML Platforms, their purpose, role of MLOps

4

ML Model for Email Classification

A ML Model for NLP Classification

5

Hands-on: Deploying ML Model in Kubernetes

Deploying and scaling an ML Model in k8s

ML IN PRODUCTION

Today's Agenda



B U J A R B A K I U

Bujar Bakiu

- ✓ CEO & Machine Learning Engineer, Data Max
- ✓ Graduated 2016, MSc, RWTH Aachen University
- ✓ Passionate about Machine Learning, Software Engineering and running
- ✓ Find me in Twitter: @bujarbakiu
LinkedIn: linkedin.com/in/bbakiu
Email: bujar@data-max.io



**Sadik
Bakiu**

PRINCIPAL ML ENGINEER



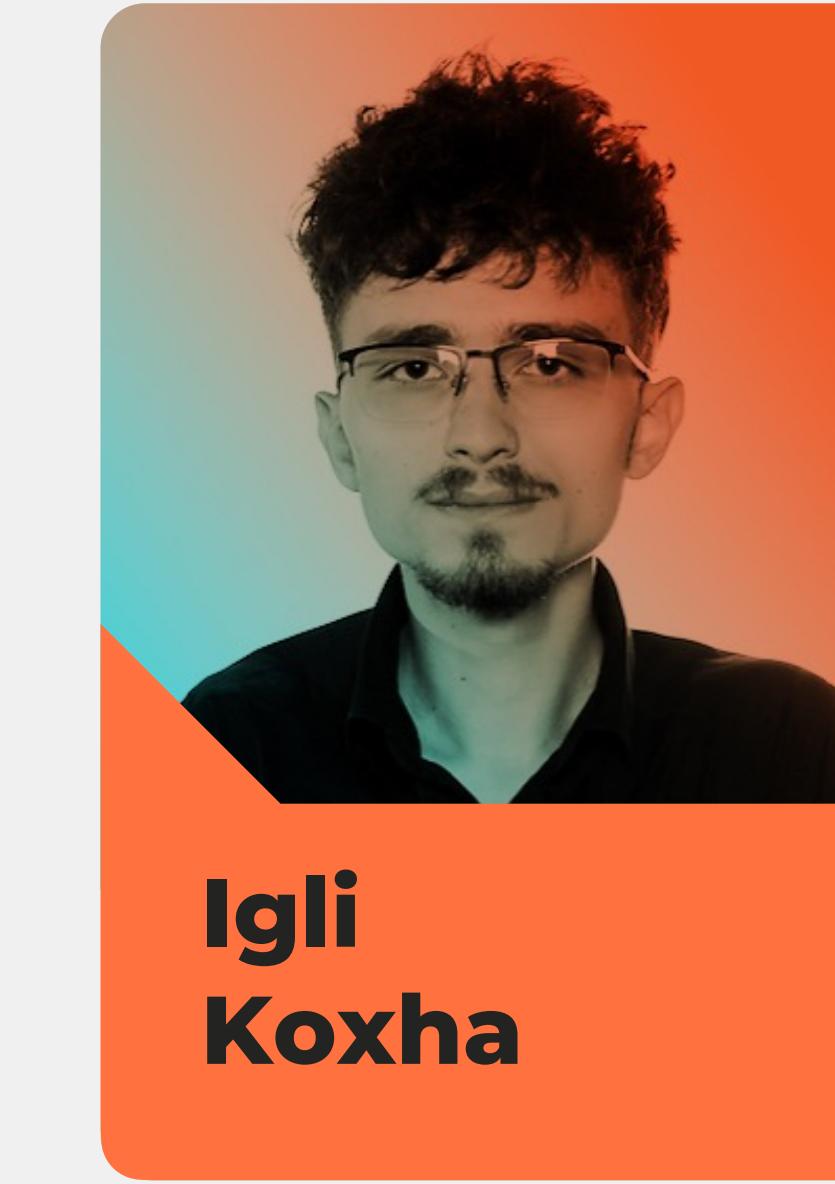
**Bujar
Bakiu**

ML ENGINEER



**Deni
Myftiu**

SYSTEMS ENGINEER



**Igli
Koxha**

ML ENGINEER

We're hiring! 🧑‍💻🧑‍💻🧑‍💻

Also, we offer paid internship! 🧑‍🎓🧑‍🎓🧑‍🎓

Find us at: data-max.io

linkedin.com/company/datamax-io

info@data-max.io / bujar@data-max.io



Hidden Technical Debt in Machine Learning

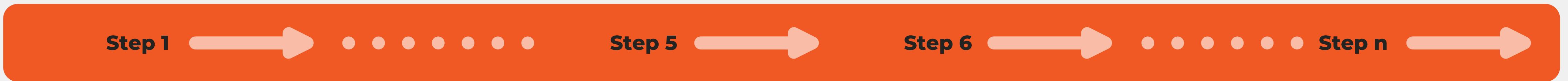
 **Technical Debt**

 **Why is it hidden?**

 **How to mitigate?**

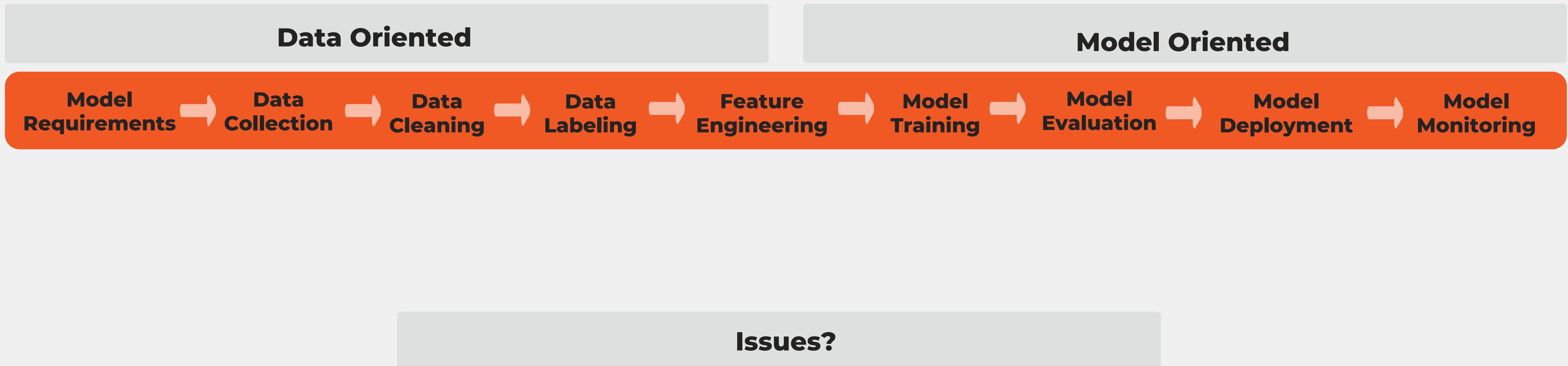
Machine Learning Workflow

How does a machine learning workflow look like?



Machine Learning Workflow

The nine stages of machine learning workflow.



TECHNICAL DEBT

Hidden Tech Debt

✓ Entanglement

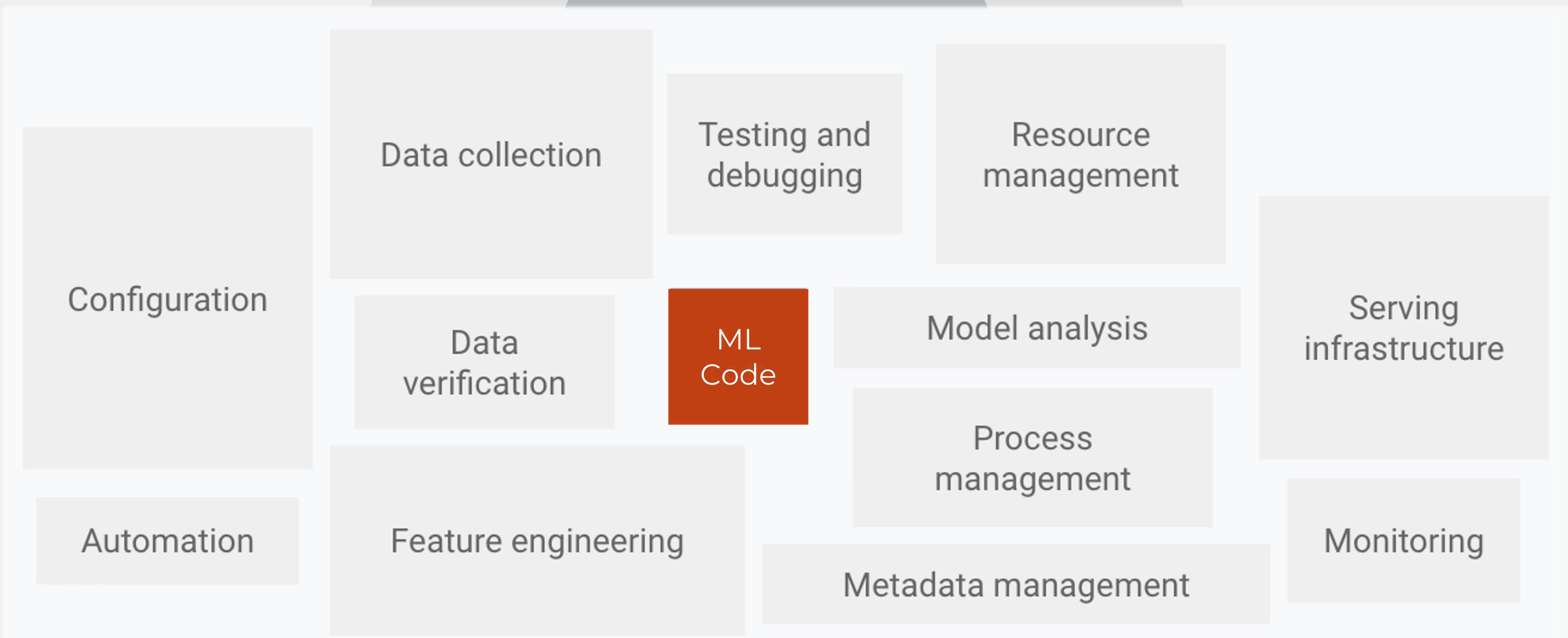
✓ Undeclared Consumer

✓ Unstable Data Dependency

✓ Underutilized Data Dependency

✓ Direct Feedback Loop

✓ Hidden Feedback Loop



**Software Engineering has
been dealing with these for a
while now.**

Questions so far?



Modern Machine Learning Platforms



Components



Deployment/Scaling



Monitoring

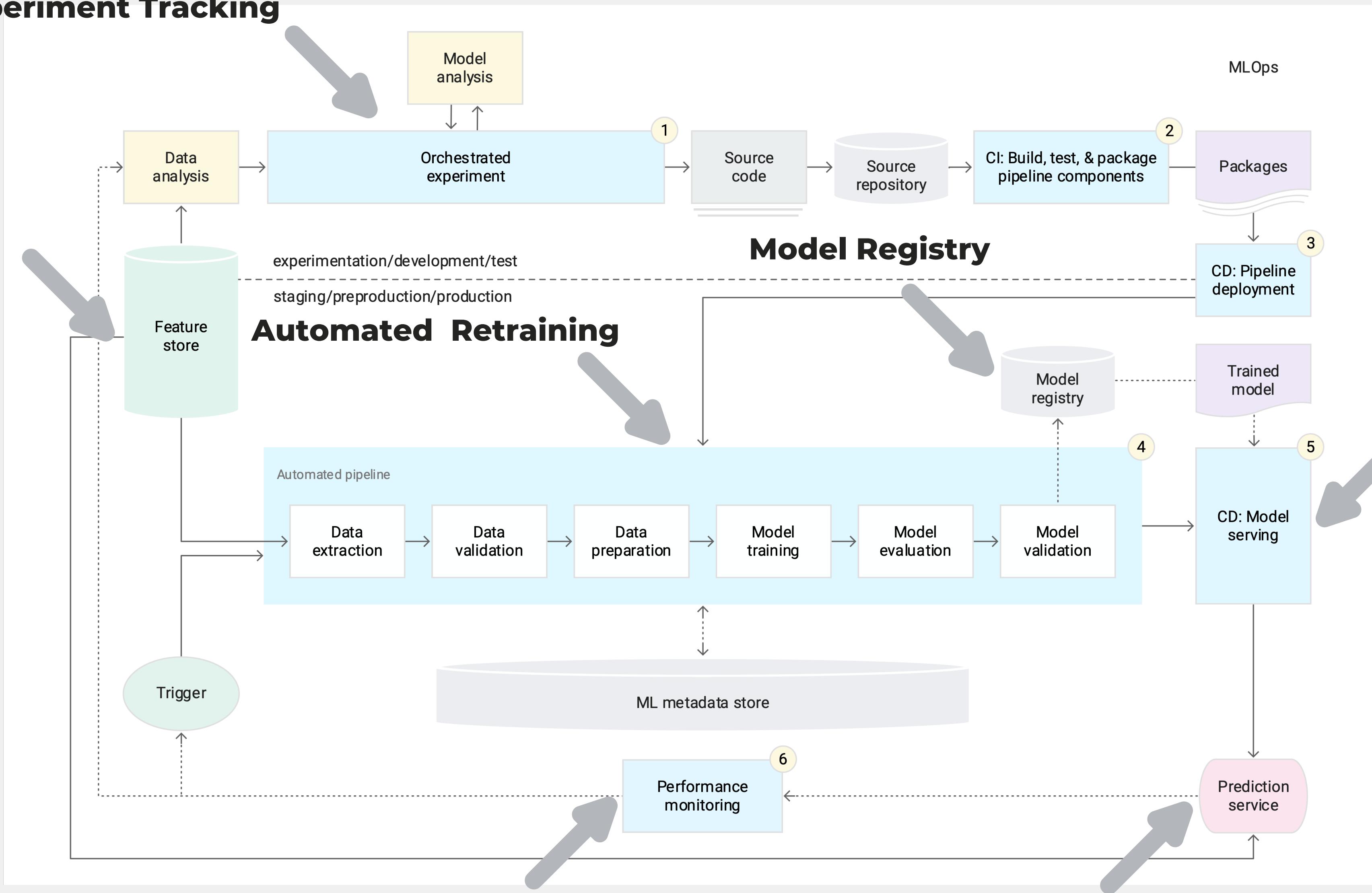
Experiment Tracking**Feature Store****Model Registry****Serving****Automated Retraining**

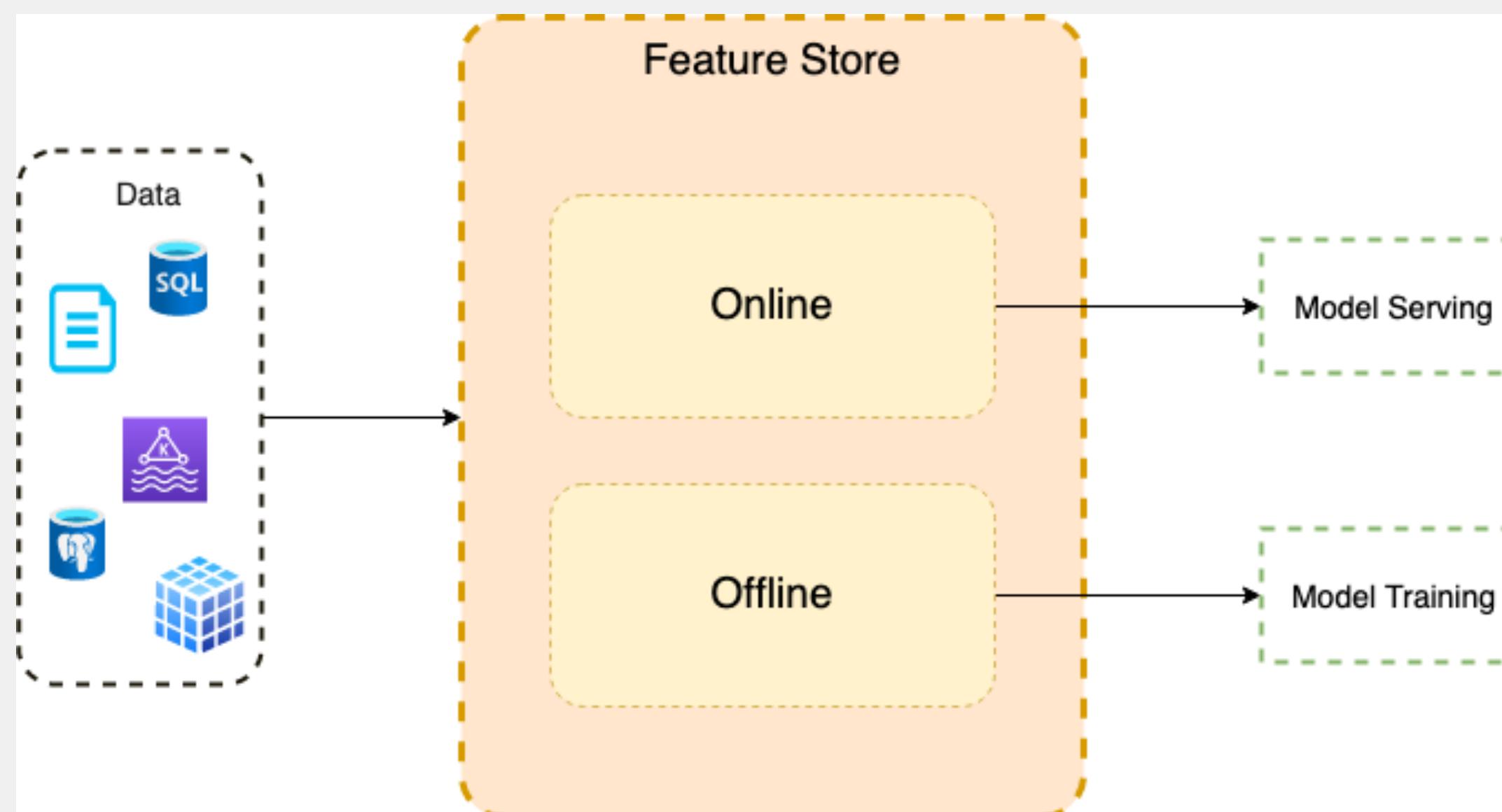
Automated pipeline

Trigger

ML metadata store

Performance monitoring

Inference**Monitoring**



Feature Store

- ✓ Stores transformed data
- ✓ Offline, stored in big warehouse/database.
Historic data used for batch processing and training
- ✓ Online, fast, responsive and usually only a small subset of all data. Used for data augmentation
- ✓ Data can be of different formats, or sources.

Example Feature

```
1 @new_feature_name(  
2     inputs={'name': 'input_table_name'},  
3     online=True,  
4     offline=True,  
5     feature_start_time=datetime(2021, 5, 20),  
6     batch_schedule='1d',  
7     ttl='30days',  
8     owner="name@email.ai",  
9     family='feature_family',  
10    description='A helpful description for the feature'  
11 )  
12 def new_feature_name_sql(inputs):  
13     return f'''  
14         SELECT  
15             colA,  
16             colB as col_B,  
17             colC  
18         FROM  
19             {inputs.name}  
20         '''
```



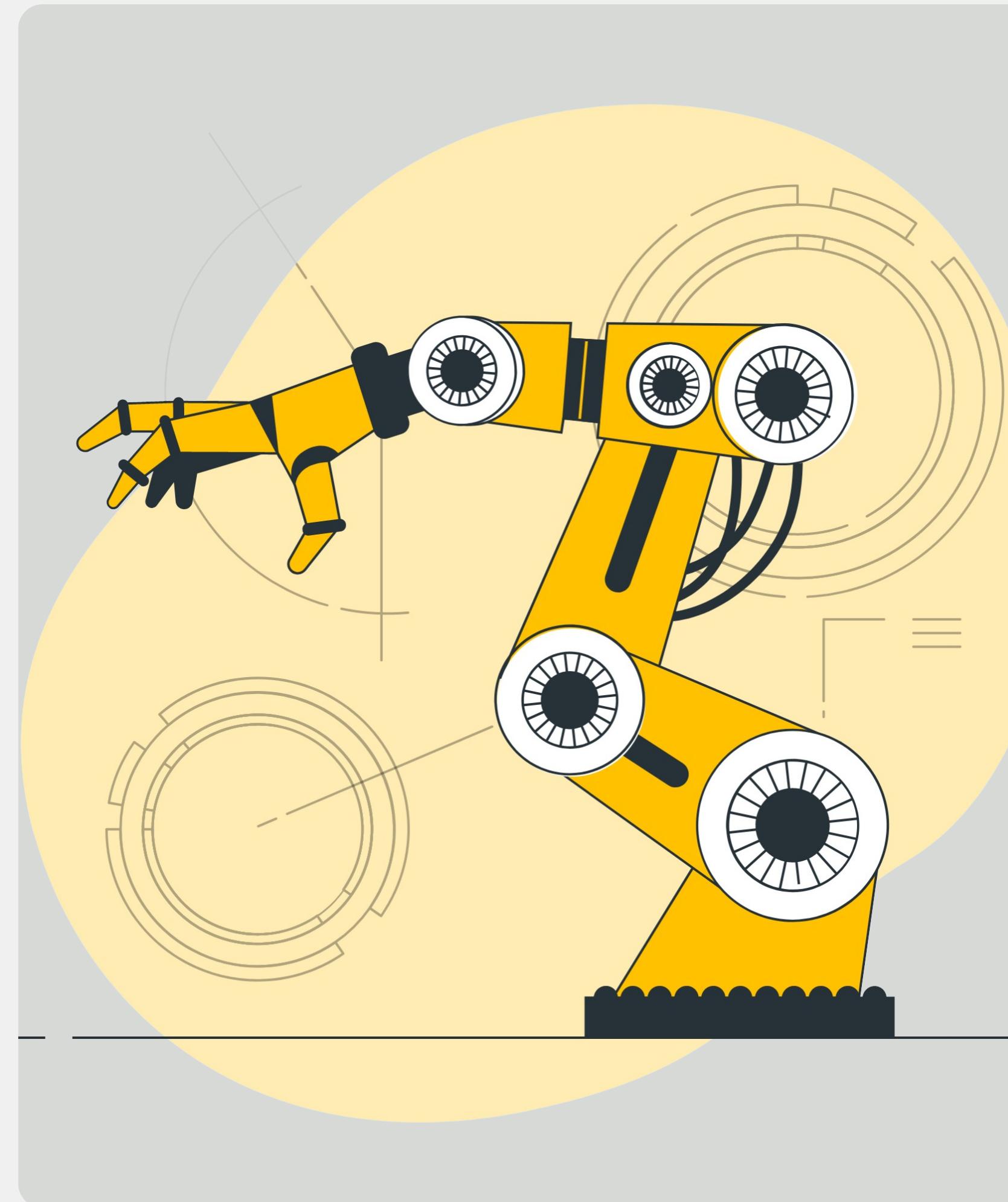
ML WORKFLOW

Experiment Tracking

- ✓ Keep track of models and hyperparameters used
- ✓ Keep track of training and evaluation data used
- ✓ Environment configuration files
- ✓ Model weights, evaluation metrics

Automated Retraining

- ✓ Train and retrain models on trigger
- ✓ Metadata stored for each step in the process
- ✓ Artifacts of each step documented
- ✓ Automated checks before going in production/next stage



ML WORKFLOW

Model Registry

- ✓ Trained model artifacts are stored
- ✓ Centralized storage
- ✓ Metadata of model
- ✓ Source for production model deployment





ML WORKFLOW

Model Serving

- ✓ Serve model as API
- ✓ Take the latest version without downtime
- ✓ Automated scaling

Model Monitoring

- ✓ Check for Drift
 - Training data not representing reality
 - Relationship between input and prediction changes
- ✓ Check for performance issues
- ✓ Monitor for bugs
- ✓ Monitor for fairness



Questions so far?

What was this all about?

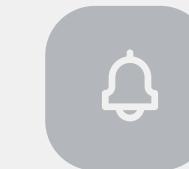
Let's go back to the diagram!



Use Case – Email Classification



Enron Data Set



Supervised Learning



Naïve Bayes

Email Classifier

Based on the content of an e-mail define on which department to forward it.

- ✓ CountVectorizer & TfIdf Transformation
- ✓ Manually labeled data from Enron dataset

- ✓ Naïve Bayes Estimator
- ✓ Not great accuracy but that is not the point

```
1 pipeline = Pipeline([
2     ("vect", CountVectorizer()),
3     ("tfidf", TfidfTransformer()),
4     ("clf", MultinomialNB())
5 ])
6
7 model = pipeline.fit(X_train, y_train)
```

Email Classifier



Serve model in a service with API

```
1 model = pickle.load(open('model.pkl', 'rb'))  
2  
3 @app.route('/predict', methods=['POST'])  
4 def predict():  
5     message = req.get("message")  
6     logging.info(f"Received message: {message}")  
7  
8     prediction = model.predict(message)  
9     logging.info(f"prediction: {prediction}")  
10  
11    return {"label": prediction}
```

Email Classifier



Containerize the service

```
1 FROM python:3.8.10
2
3 WORKDIR /app
4
5 COPY requirements.txt requirements.txt
6 RUN pip install -r requirements.txt
7
8 COPY app.py app.py
9 COPY model.pkl model.pkl
10
11 EXPOSE 5000
12
13 ENV FLASK_APP=app.py
14 ENV FLASK_RUN_HOST=0.0.0.0
15
16 CMD ["flask", "run"]
```

Email Classifier

- ✓ Build image
- ✓ Add to cluster
- ✓ Deploy and scale

```
1 # Build Docker image
2 docker build -t email_classifier_image .
3 # Add image to cluster
4 minikube image load email_classifier_image
5
6 # create kubernetes resources
7 kubectl apply -f deployment.yaml
8
9 # scale deployment
10 kubectl scale deployment email-classifier-deployment
11 -n email-classifier-namespace --replicas=3
```

Email Classifier



Kubernetes resources

```
1 # Build Docker image
2 docker build -t email_classifier_image .
3 # Add image to cluster
4 minikube image load email_classifier_image
5
6 # create kubernetes resources
7 kubectl apply -f deployment.yaml
8
9 # scale deployment
10 kubectl scale deployment email-classifier-deployment
11 --namespace email-classifier-namespace --replicas=3
```

Email Classifier



Kubernetes resources

```
1 kind: Namespace
2 apiVersion: v1
3 metadata:
4   name: email-classifier-namespace
5   labels:
6     name: email-classifier
7   -
8   apiVersion: v1
9   kind: Service
10  metadata:
11    name: flask-service
12    namespace: email-classifier-namespace
13  spec:
14    selector:
15      app: email-classifier
16    ports:
17      - protocol: "TCP"
18        port: 5000
19        targetPort: 5000
20    type: ClusterIP
21
```

```
1   -
2   apiVersion: apps/v1
3   kind: Deployment
4   metadata:
5     name: email-classifier-deployment
6     namespace: email-classifier-namespace
7   spec:
8     selector:
9       matchLabels:
10         app: email-classifier
11     replicas: 1
12     template:
13       metadata:
14         labels:
15           app: email-classifier
16     spec:
17       containers:
18         - name: email-classifier
19           image: email-classifier:latest
20           imagePullPolicy: Never
21         ports:
22           - containerPort: 5000
```

THANK YOU!

- data-max.io
- info@data-max.io

- [Linkedin.com/company/datamax-io](https://www.linkedin.com/company/datamax-io)
- bujar@data-max.io