iterative.ai

Model Registry with Open-Source tools: Git, GItHub and CI/CD

Co-Founder & CEO at Iterative.ai

Dmitry Petrov, Ph.D.

dmitry@iterative.ai

About me

Dmitry Petrov

@FullStackML

Before

Ex-Data Scientist at Microsoft

Now

Creator of **D**ata **V**ersion **C**ontrol

Co-founder, CEO at Iterative.ai
(San Francisco, CA)

I. Model registry & challenges

What is Model Registry (1)

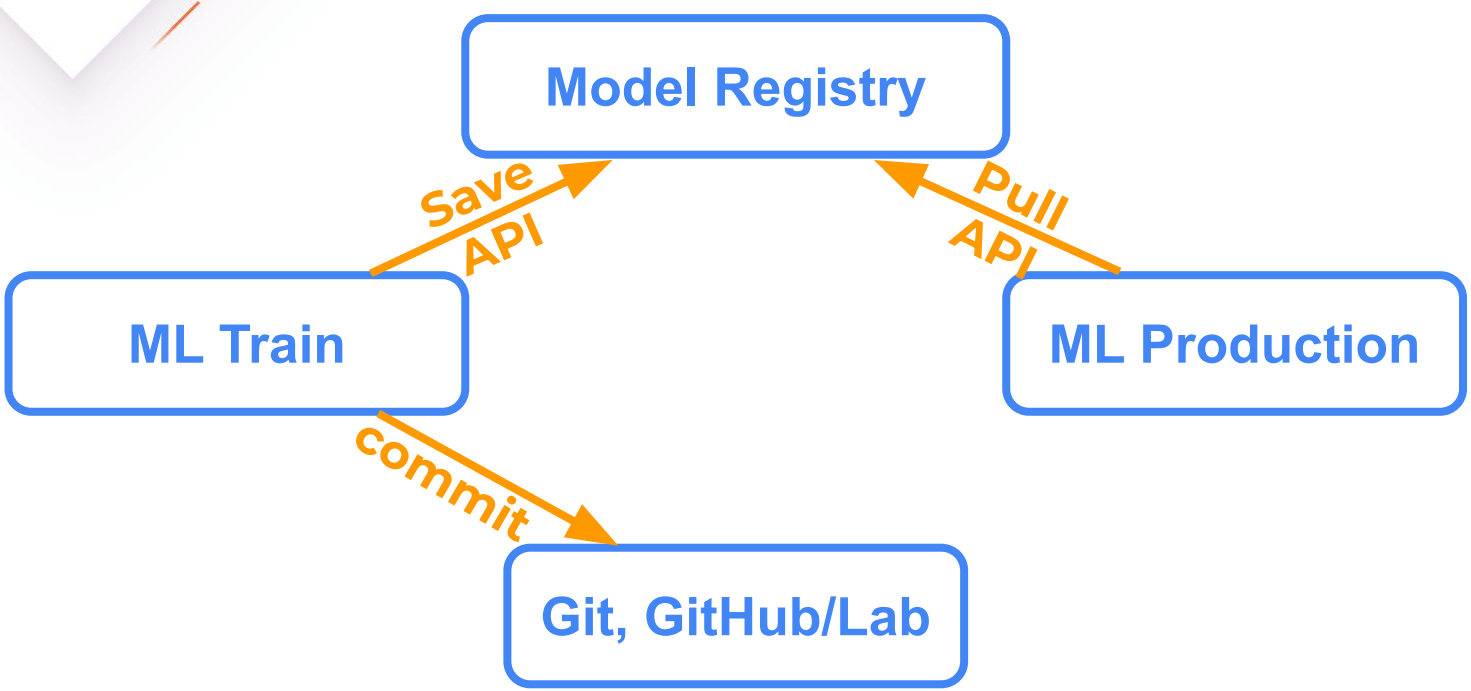
Centralized model store to collaboratively manage the full lifecycle of ML models.

- Model lineage & versioning
- Stage transitions: from staging to production
- Model Annotations & Discovery: timestamp, description, labels

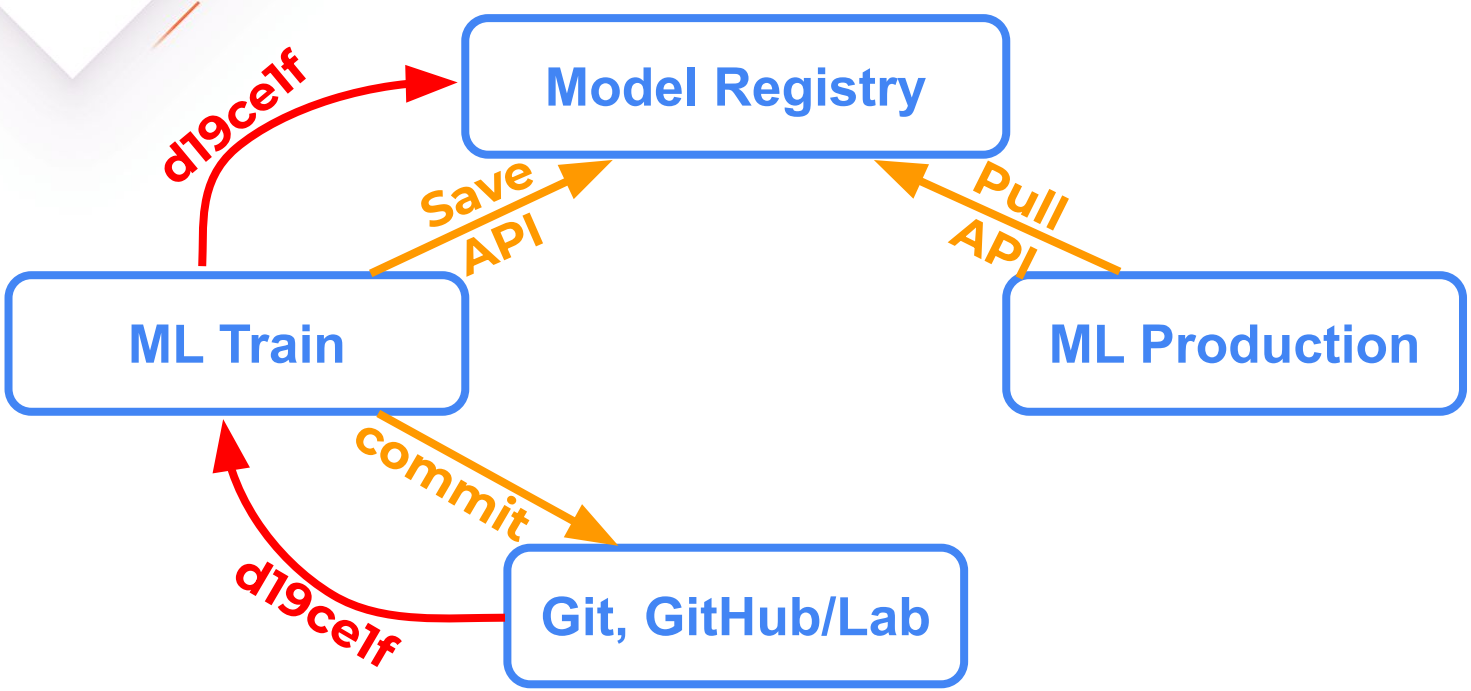
What is Model Registry (2)

Name	Version	Status	Code	Meta-info: labels
churn	3.1	prod	43ba862	customer, xgboost
segment	0.4	-	d19ce1f	cv, car
cv-class	1.13	staging	afd8e35	cv, car, doors, glass
…	…	…	…	…

What is Model Registry (3)



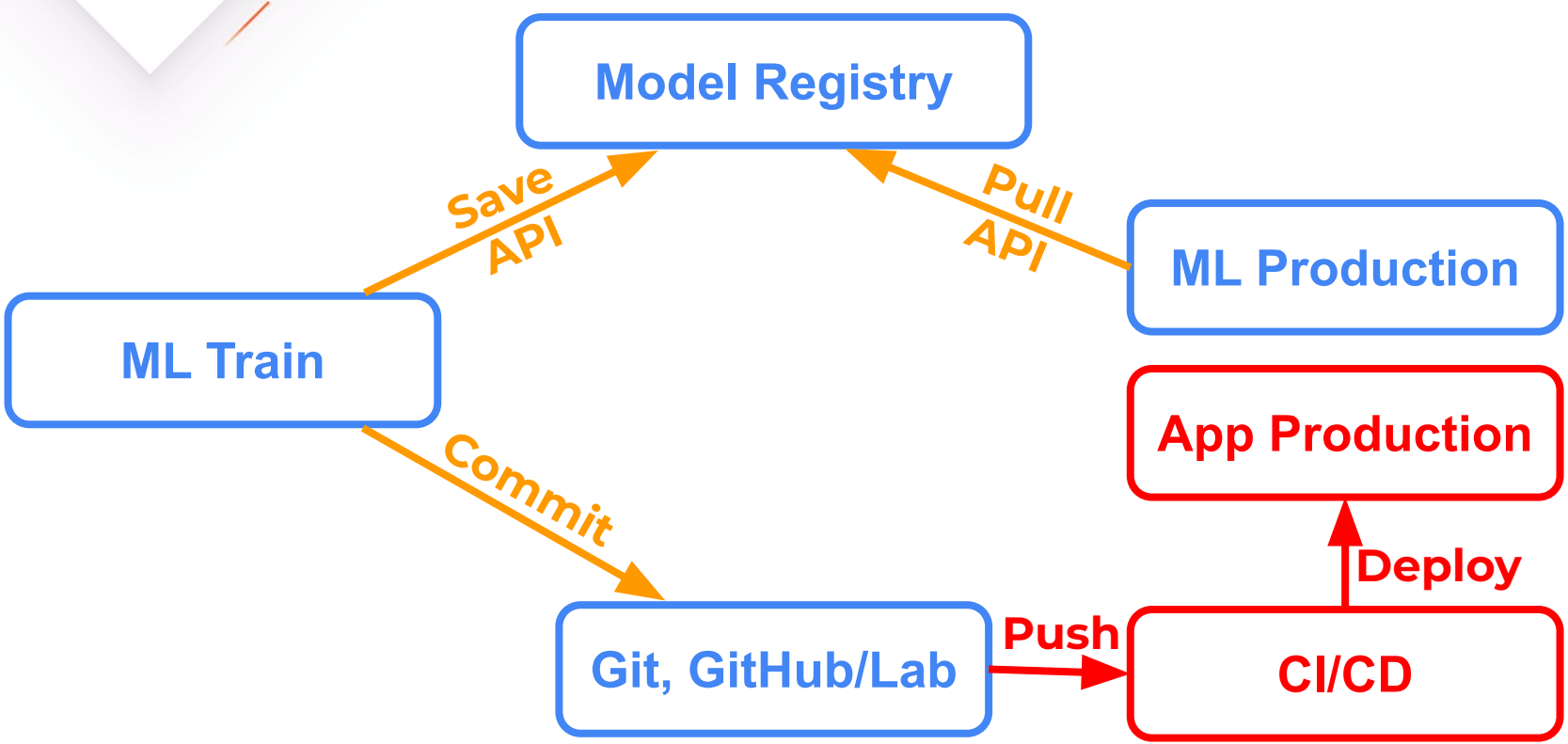
Challenge 1: model & code lineage (1)



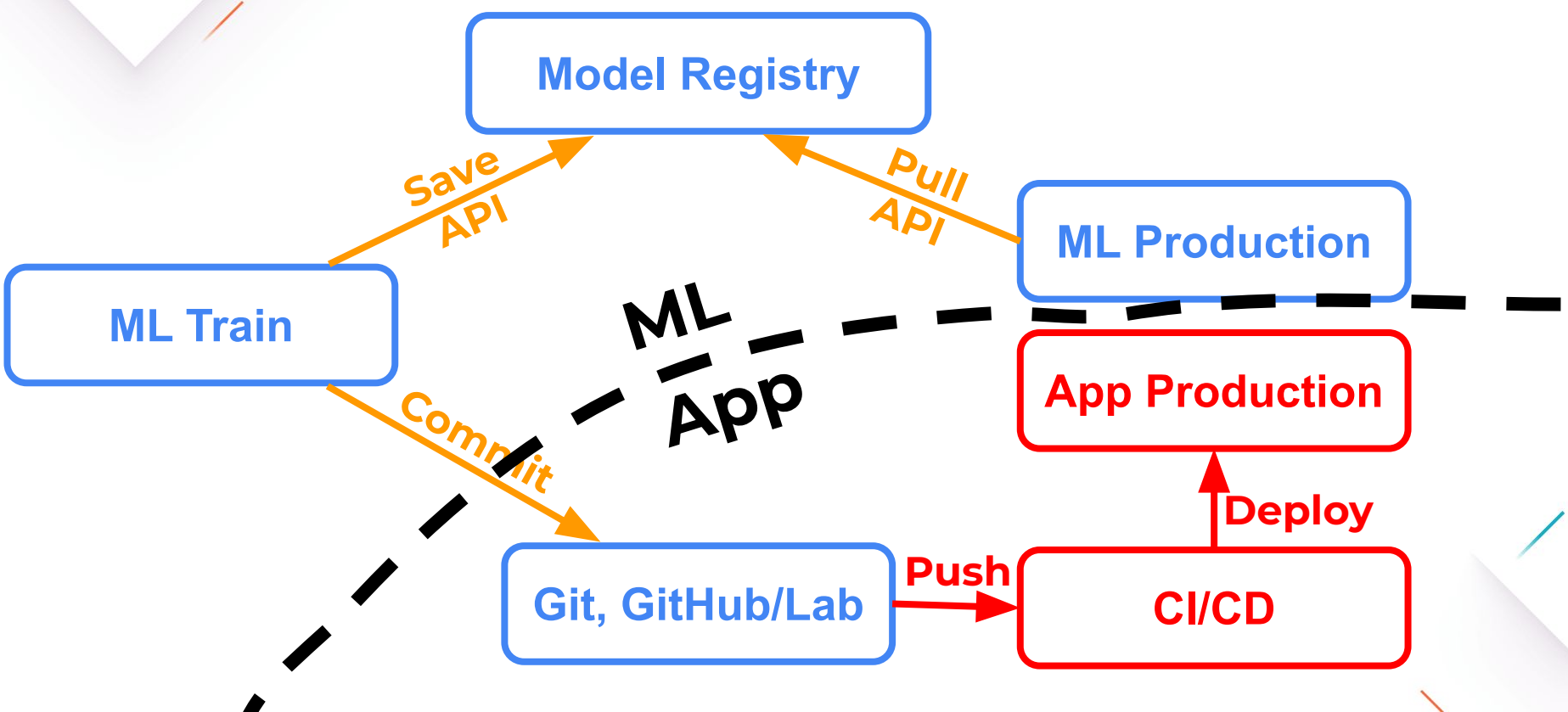
Challenge 1: model & code lineage (2)

Name	Version	Status	Code	Meta-info: labels
churn	3.1	prod	~~43ba862~~	customer, xgboost
segment	0.4	-	~~d19ce1f~~	cv, car
cv-class	1.13	staging	afd8e35	cv, car, doors, glass
…	…	…	…	…

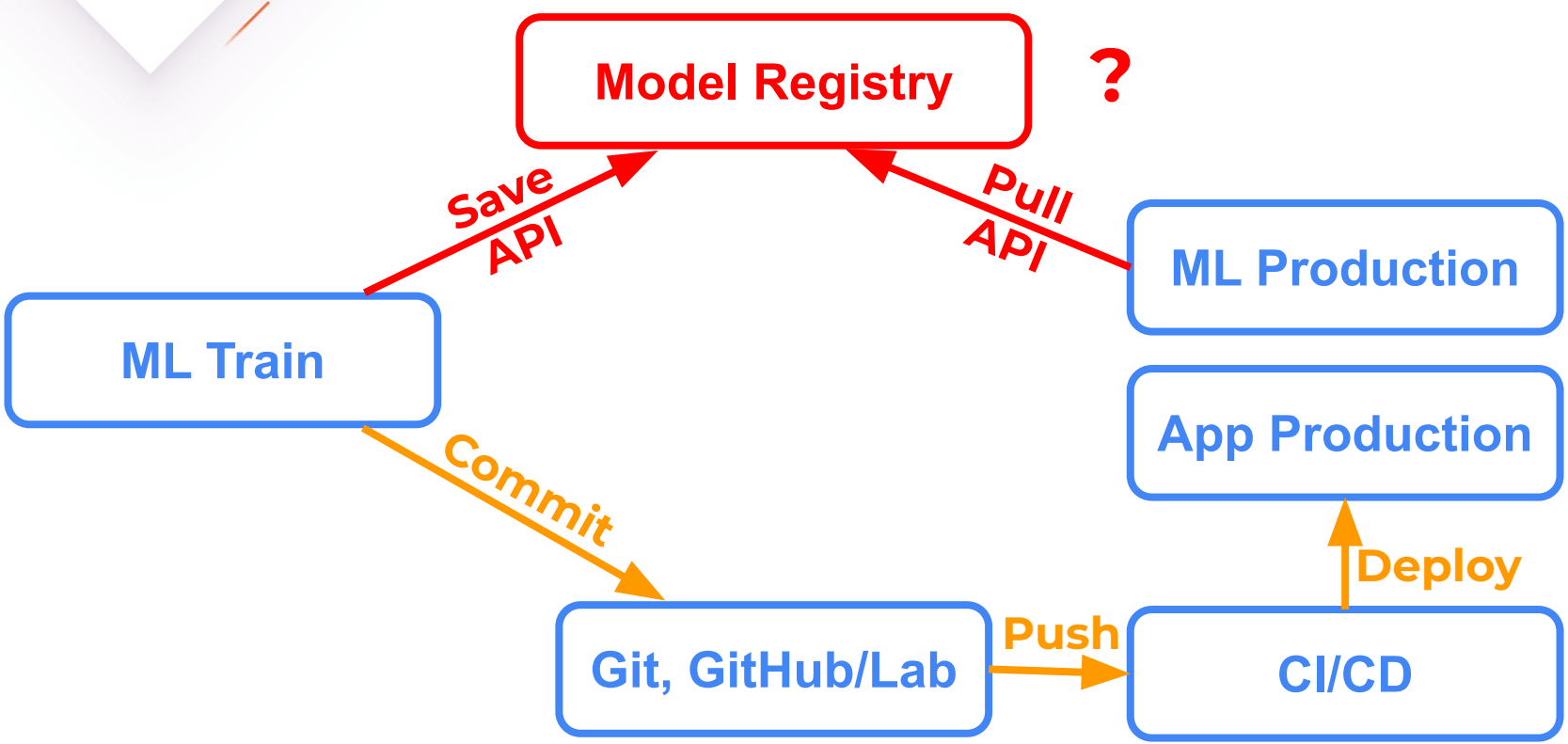
Challenge 2: app & model deployment diff (1)



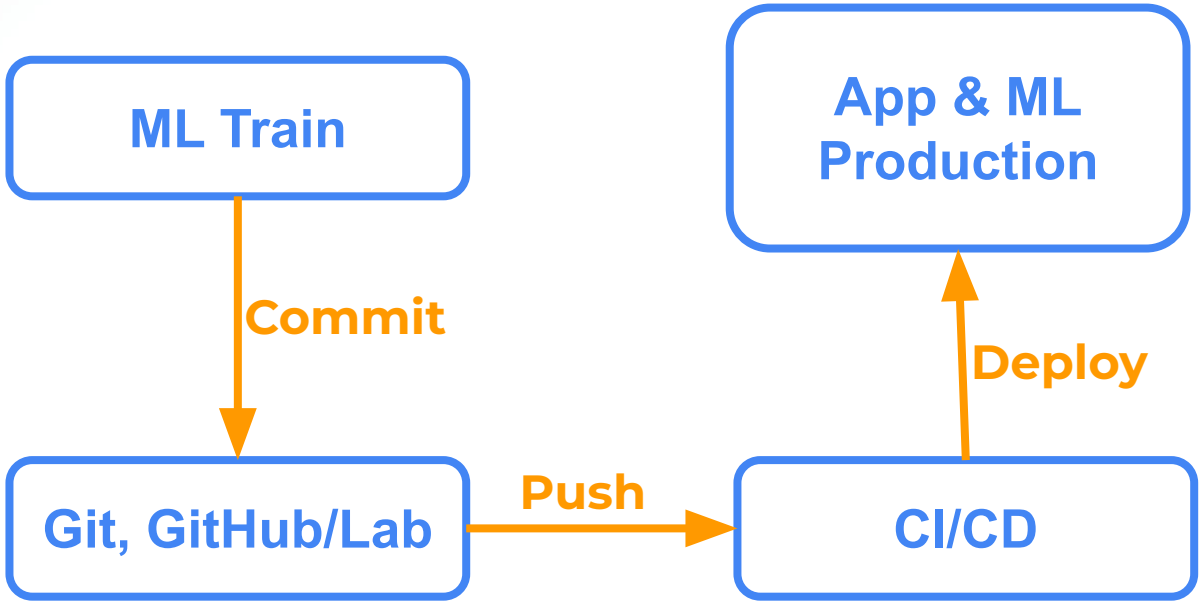
Challenge 2: app & model deployment diff (2)



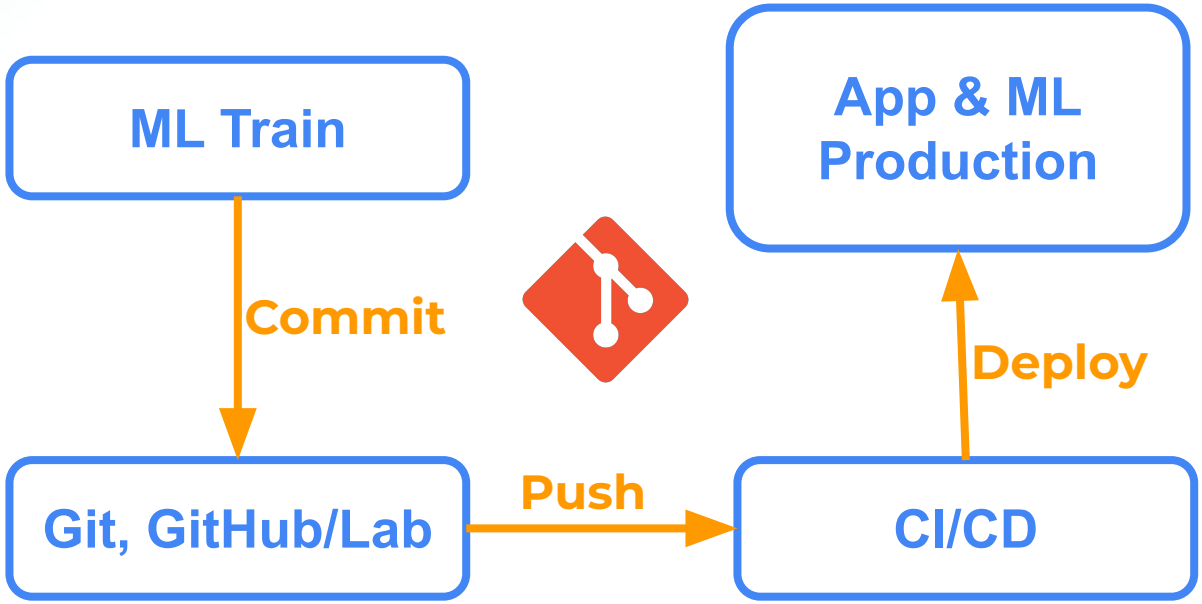
Challenge 3: additional service



Git as the source of truth



Git as the source of truth: Git only API



II. What is needed from Git?

How to save the table in Git?

Name	Version	Status	Code	Meta-info: labels
churn	3.1	prod	43ba862	customer, xgboost
segment	0.4	-	d19ce1f	cv, car
cv-class	1.13	staging	afd8e35	cv, car, doors, glass
…	…	…	…	…

Model registry functionality (1)

Basic:

- Human-friendly versioning: d19ce → v1.0.1
- Status: prod, staging, dev

Integration to production:

- Push to production

Model registry functionality (2)

Basic:

- Human-friendly versioning: d19ce → v1.0.1
- Status: prod, staging, dev

Integration to production:

- Push to production

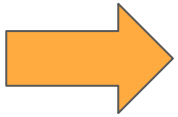
Advanced:

- Large ML models
- Mono-repo or multiple models
- Annotations

Model registry functionality (3)

Basic:

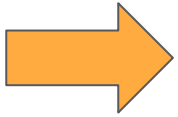
- Human-friendly versioning
- Status



Git tags

Integration to production:

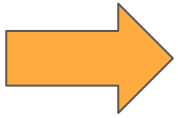
- Push to production



CI/CD

Advanced:

- Large ML models
- Mono-repo or multiple models
- Annotations



Meta-file
artifacts.yaml

Solution: Git as the source of truth

Infrastructure as Code (IaC)

We are creating **Model Registry as Code (MRaC)**

Advanced:

- Large ML models
- Mono-repo or multiple models
- Annotations

III. Git as model registry (Basic)

Git: Human-readable Versioning

```
$ git tag -a model-v1.0.0 -m 'my model'

$ git push origin model-v1.0.0 # or --tags

$ git tag -l 'model*'
model-v1.0.0
```

Git: model status - from Stage to Prod (1)

```
$ git tag -a model-staging -m 'move my model to staging'

$ git push origin --tags
```

Git: model status - from Stage to Prod (2)

```
$ git tag -d model-staging
$ git tag -a model-prod -m 'promote to production'

$ git push origin --delete model-staging
$ git push origin --tags
```

Git: Model deployment (GitHub Action)

```yaml
name: ML-Production-Deployment
on:
  push:
    tags:
      - model-prod
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
```

GTO - Git Tag Ops

https://github.com/iterative/gto

Turn your Git Repo into Artifact Registry:

- Register new versions of artifacts marking significant changes to them
- Promote versions to signal downstream systems to act
- Attach additional info about your artifact with Enrichments
- Act on new versions and promotions in CI

GTO: Human-readable Versioning

```
$ gto register model
Created git tag 'model@v0.0.1'

$ gto register model --bump-patch
Created git tag 'model@v0.0.2'

$ gto register model --bump-major
Created git tag 'model@v1.0.0'
```

GTO: model status - from Stage to Prod

```
$ gto promote model prod --simple
Created git tag 'model#prod' that changes status

$ gto promote model my-stage2 --simple
Created git tag 'model#my-stage2' that changes status
```

GTO: reliable / numeric model statuses

```
$ gto promote model prod
Created git tag 'model#prod-1' that changes status

$ gto promote model prod
Created git tag 'model#prod-2' that changes status
```

GTO: Model Registry in command line

```
$ gto show
```

name	latest	#prod	#staging	#dev
churn	v3.1.0	v3.0.0	v3.1.0	-
segment	v0.4.1	-	-	v0.4.1
cv-class	v0.1.13	-	-	-

GTO: important!

1. Do not forget to push tags:

```
$ git push origin --tags
```

2. CI/CD tags pattern:

```
on:
  push:
    tags:
      - model#prod#*  # Numeric tags
```

Git: summary

Versioning	**Git tag `model-v1.0.0`**
Status	**Git tag `model-prod`**
Pull & Push to production	**CI/CD**

VI. Git as model registry (Advanced)

Git: Large ML model files - artifacts.yaml

```
$ cat artifacts.yaml
model:
  path: s3://mycorp/proj-ml/model-2022-04-15.pt

$ yq -r .model.path artifacts.yaml
s3://mycorp/proj-ml/model-2022-04-15.pt
```

Git: multiple models in a single repo (1)

```
$ git tag -a segm-nn-v1.0.0 -m 'NN model init'

$ git tag -a segm-nn-prod -m 'NN to production'
```

Git: multiple models in a single repo (2)

artifacts.yaml
(with sections)

```yaml
classif:
  path: s3://mycorp/proj-ml/classif-v2.pt

segm-nn:
  path: s3://mycorp/proj-ml/segm-model-2022-04-15.pt
```

Git: ML model annotations

```
classif:
  path: s3://mycorp/proj-ml/classif-v2.pt

segm-nn:
  description: 'This is my CV model'
  tags: ['cv', 'segmentation', 'pedestrian']
  path: s3://mycorp/proj-ml/segm-model-2022-04-15.pt
```

GTO: Large ML model files

```
$ gto annotate classif --path s3://mycorp/class-ml/mod.pt

$ cat artifacts.yaml
classif:
  path: s3://mycorp/proj-ml/classif-v2.pt
```

```
$ gto describe classif --path
s3://mycorp/proj-ml/classif-v2.pt
```

GTO: ML model annotations

```
$ gto annotate segm-nn \
    --path s3://mycorp/proj-ml/segm-model-2022-04-15.pt \
    --description 'This is my CV model' \
    --type model --label CV --label segmentation
```

artifacts.yaml

```
segm-nn:
  description: This is my CV model
  labels: ["CV", "segmentation"]
  path: s3://mycorp/proj-ml/segm-model-2022-04-15.pt
  type: model
```

GTO: multiple models in a single repo

```yaml
classif:
  path: s3://mycorp/proj-ml/classif-v2.pt
segm-nn:
  description: This is my CV model
  labels: ["CV", "segmentation"]
  path: s3://mycorp/proj-ml/segm-model-2022-04-15.pt
  type: model
```

Git: summary

Versioning	Git tag `model-v1.0.0`
Status	Git tag `model-prod`
Pull & Push to production	CI/CD
Large ML models	URL in `artifacts.yaml`
Multiple models	Sections in `artifacts.yaml`
Annotations	Labels in `artifacts.yaml`

V. Limitations of Git

Model visibility

Git good for “inside repository” model registry, not cross-repository.

Summary: Model Registry with Git

- Git as the source of truth
- No need in external services: only Git, GitHub/Lab
- Limitation - only in repository level

iterative.ai

Thank you!

Dmitry Petrov

@FullStackML

Email dmitry@iterative.ai