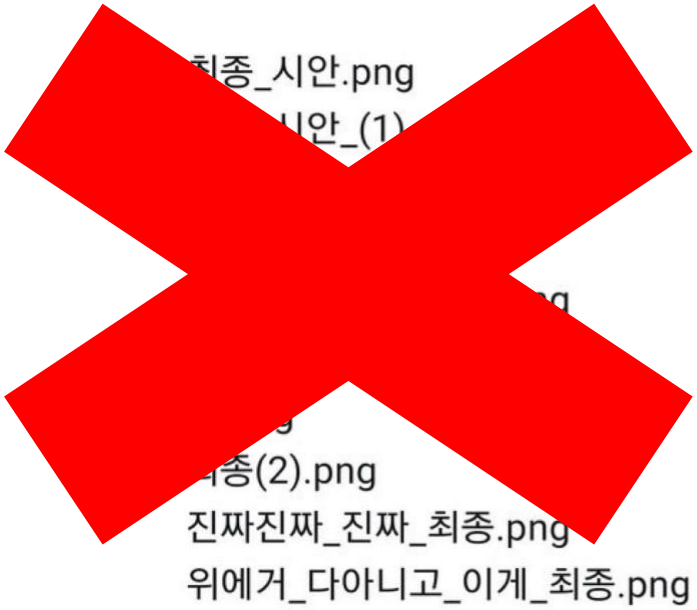
데이터 버전 관리.araboza

문성주 <arcuer.dev@gmail.com>

버전 관리?



아 이게 뭐야...
뭐 어쩌라는 거지 뭐가 최종이야.. 야발

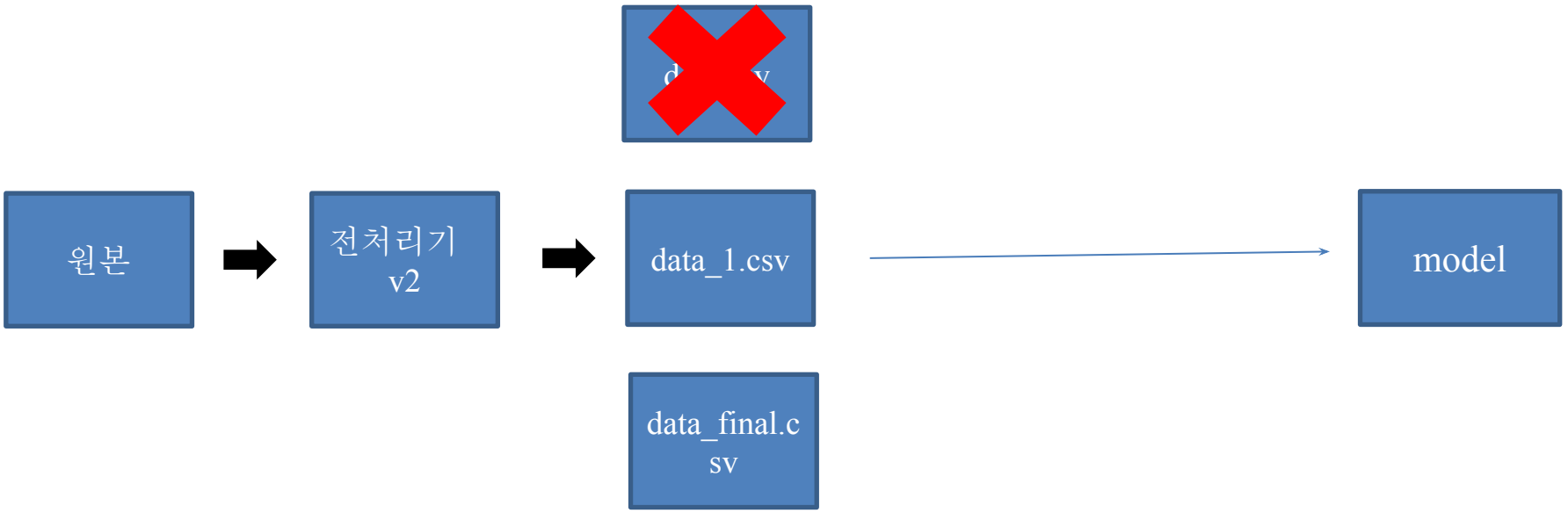
이런 상황을 피하기 위해 사용합니다..

데이터 버전 관리를 왜할까?

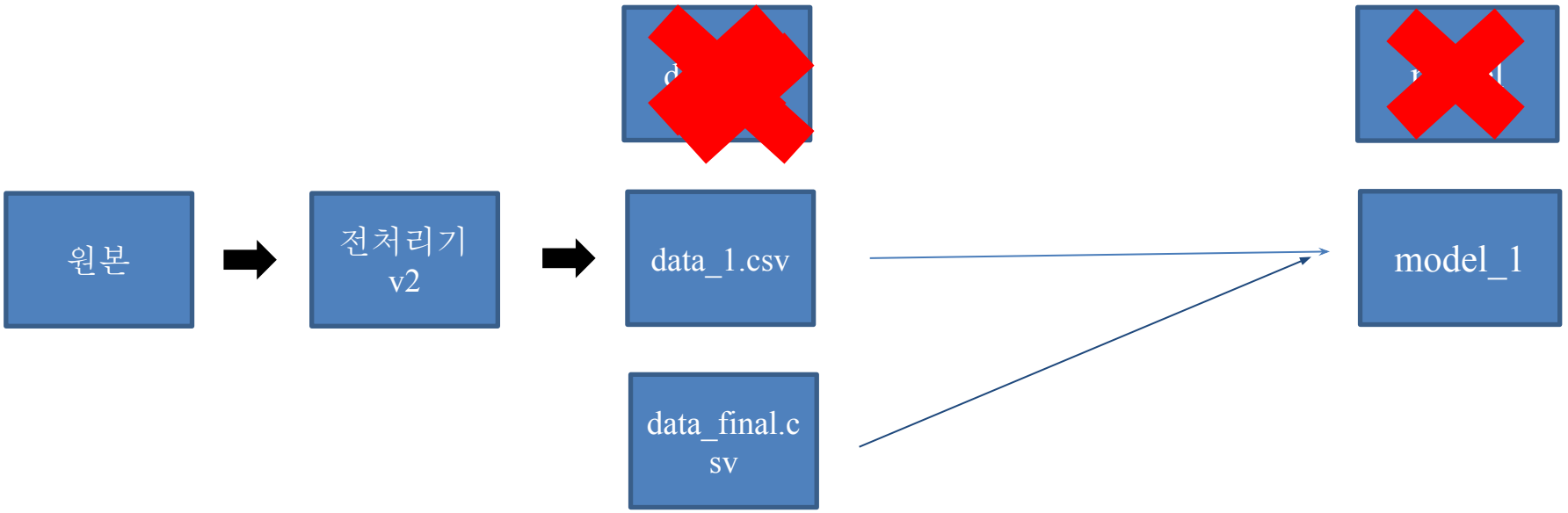
모델은 학습을 한다던데....

이런 flow로 학습을 하는데 만약..?

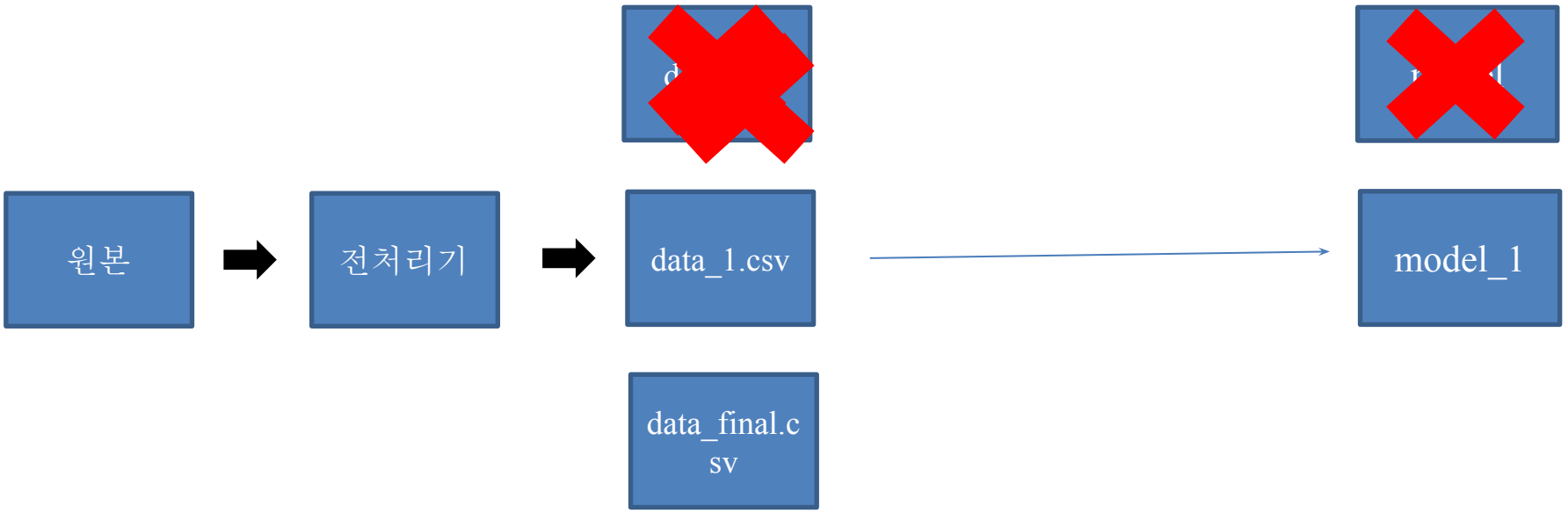
데이터 처리하는 방식이 바뀌면?



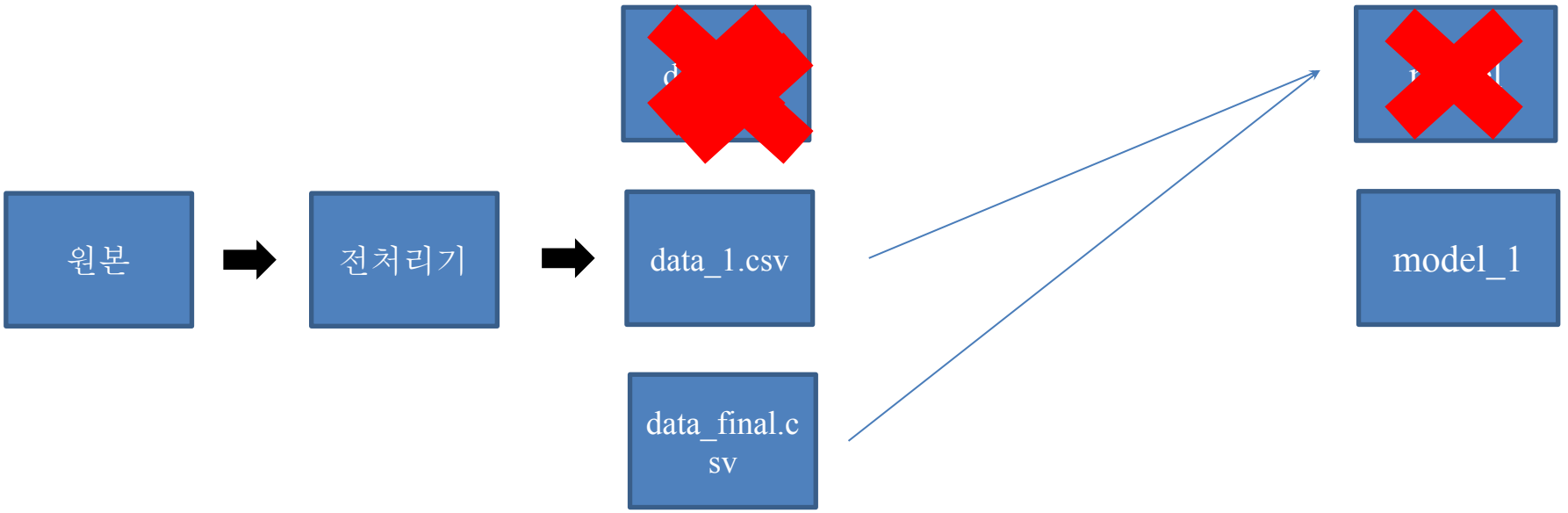
근데 모델이 바뀌면 어쩌죠?



만약 이전 모델의 문제가 생기면 어쩌죠?

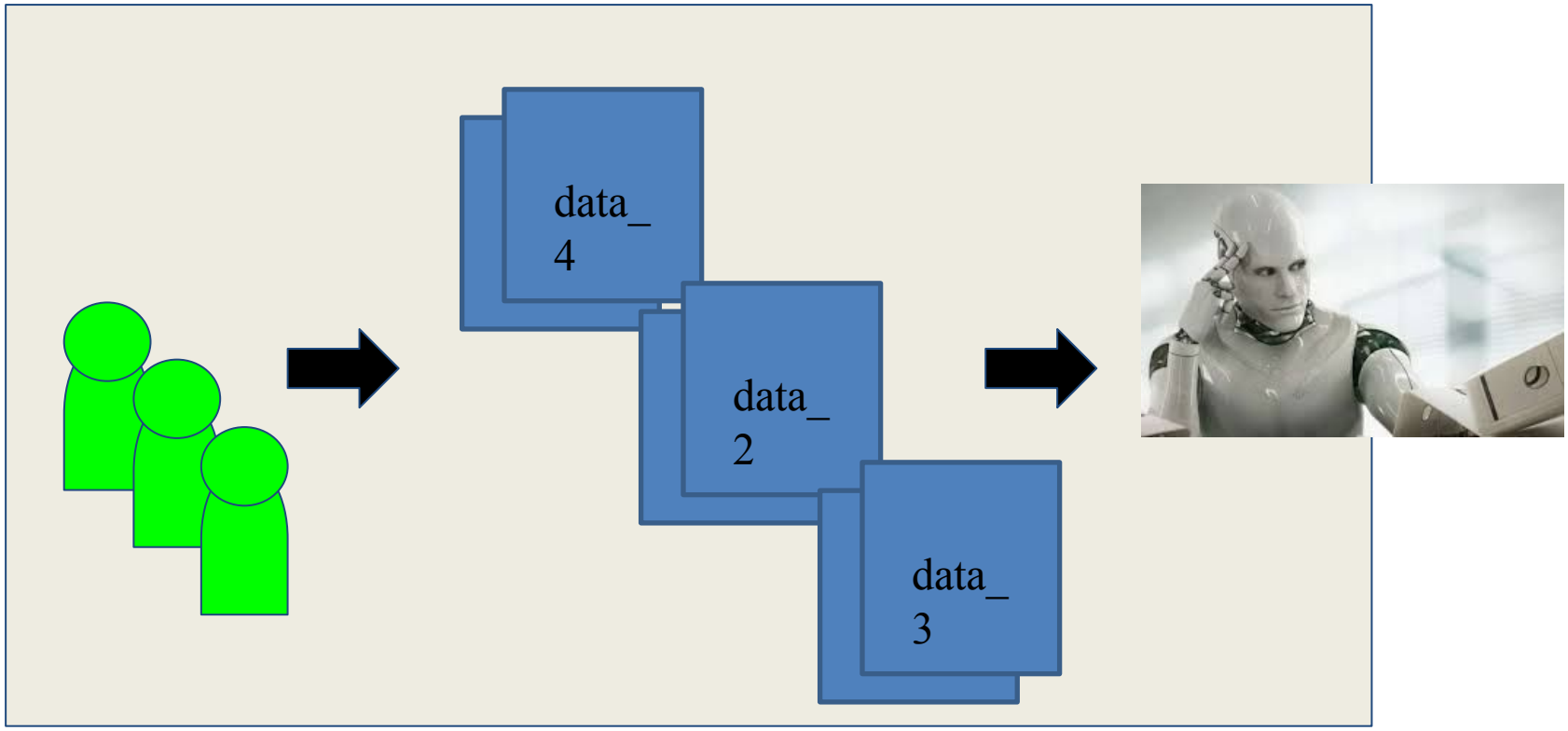


이전 모델의 문제가 생기면 어쩌죠?



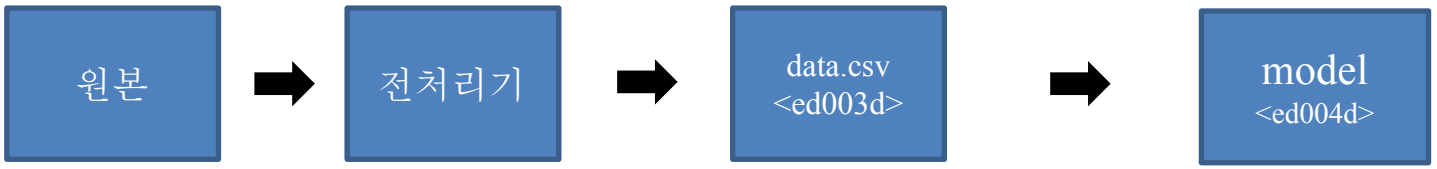
데이터셋도 바뀌었고 모델도 바뀌었는데 어떻게 재현을 하죠?..

사실 문제 재현만이 아니라..



모델을 실험을 하기 위해서 다양한 데이터가 한데..
만약 전처리기를 바꾸게 되면 어떨까?
바꾼다고 해도 혼자면 괜찮은데 만약 여러명들의 사람과 동시에 작업한다면.. 과연 어떻게 될까?

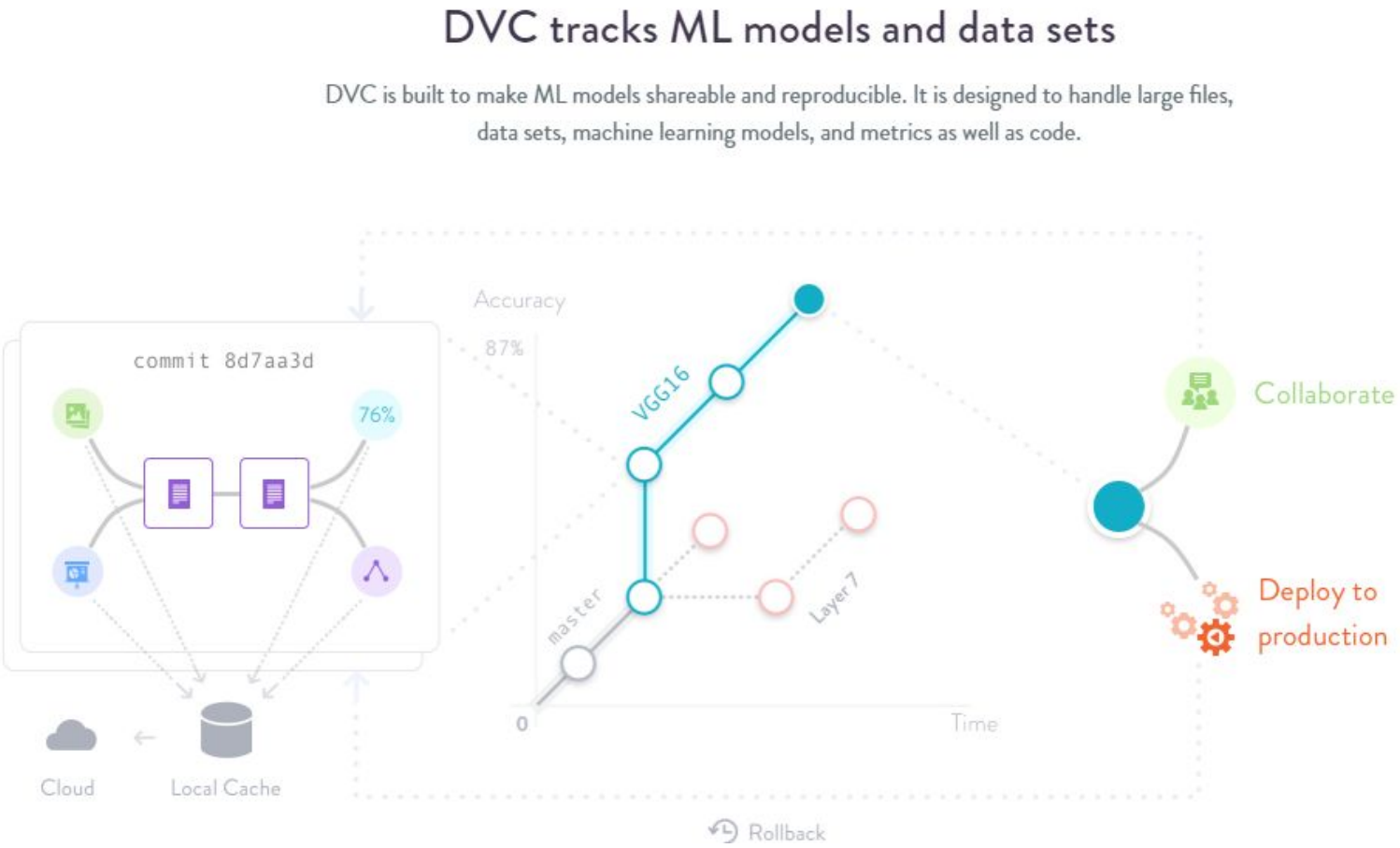
그러면 데이터랑 모델을…



버전관리를 해버리자?!

그런 솔루션이 필요한 여러분을 위해…

https://dvc.org



자 dvc를 한번 해볼까요?

자세한 정보는 https://dvc.org/doc/get-started/install에서 확인할 수 있습니다.

pip

pip install dvc

ubuntu deb

```
$ sudo wget https://dvc.org/deb/dvc.list -O /etc/apt/sources.list.d/dvc.list
$ sudo apt-get update
$ sudo apt-get install dvc
```

centos rpm

```
$ sudo wget https://dvc.org/rpm/dvc.repo -O /etc/yum.repos.d/dvc.repo
$ sudo yum update
$ sudo yum install dvc
```

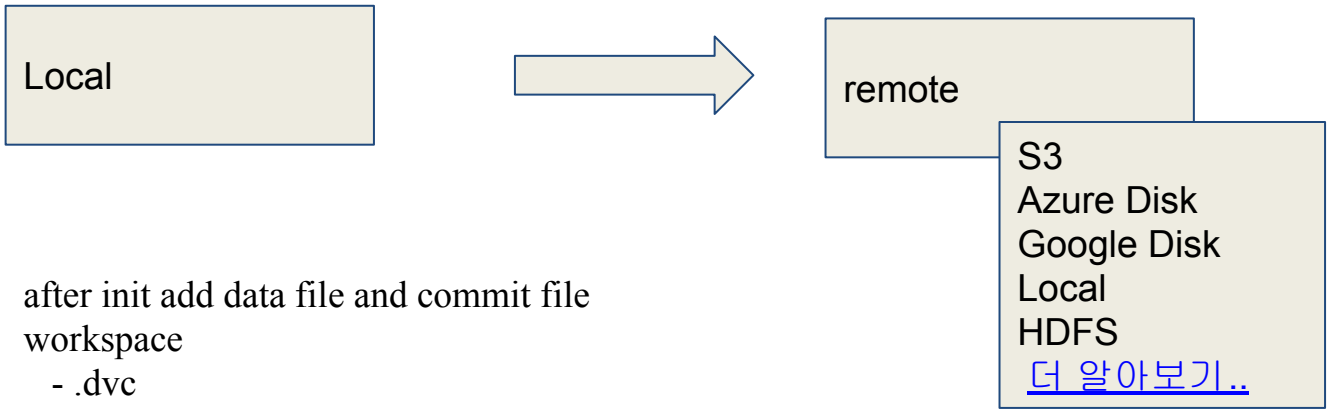
macOS

```
brew install iterative/homebrew-dvc/dvc
```

그러면 한번 해봅시다?

https://dvc.org/doc/get-started/example-versioning

dvc init
create .dvc 디렉토리

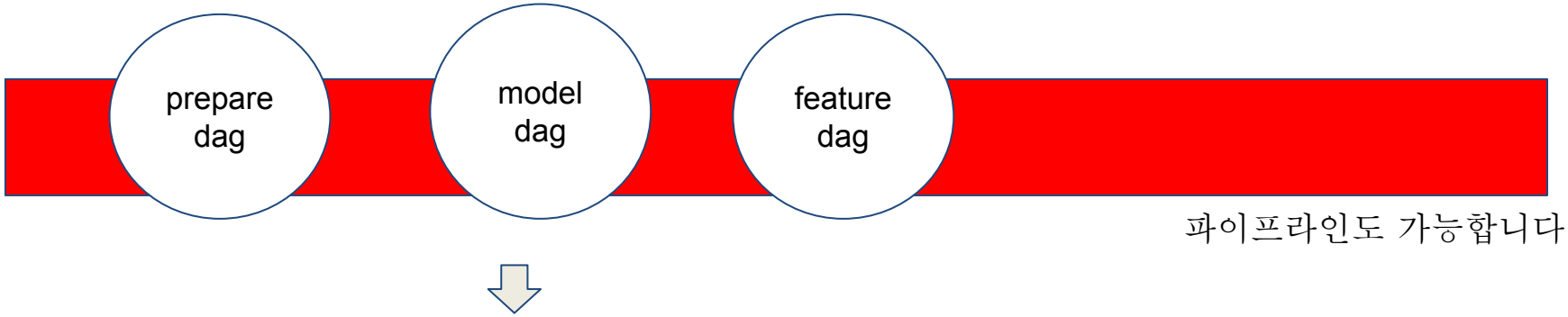


dvc push
add metadata
- data
  - data.xml.dvc

after init add data file and commit file
workspace
- .dvc
- data
  - data.xml

지원하는 리모트 저장소
dvc remote add -d local /tmp/data/local
dvc remote add -d local s3://backup/data

그러면 그렇게 데이터 버전 관리만 하나요?

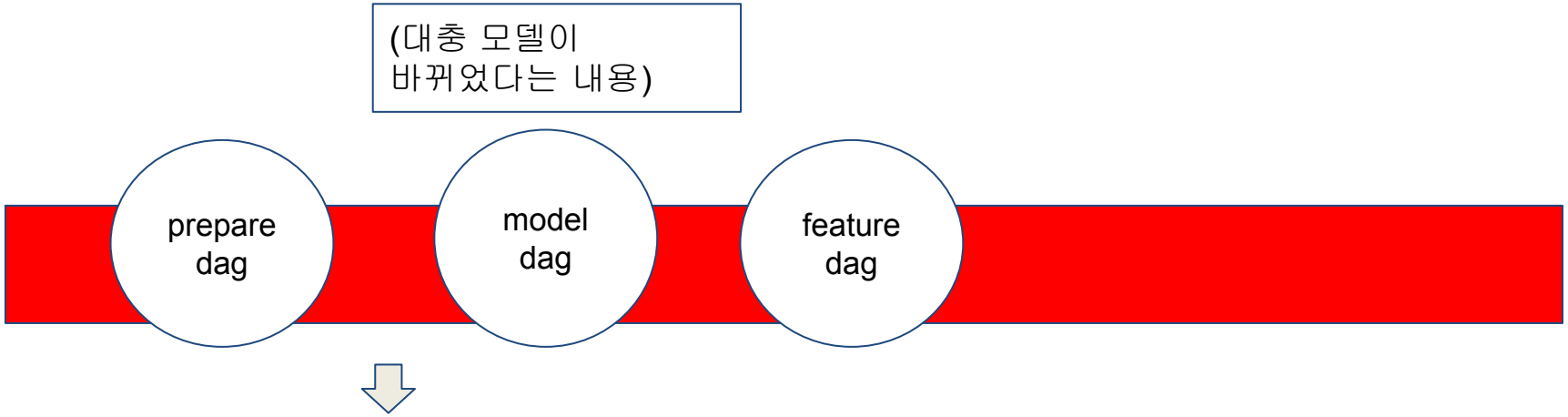


파이프라인도 가능합니다!

```
(classify-test) seongju@seongJu-notebook:~/workspace/classify$ dvc run -d code/evaluate
.py -d data/model.pkl -d data/matrix-test.pkl          -M metrics/auc.metric
 -f evaluate.dvc          python code/evaluate.py
Running command:
        python code/evaluate.py
Output 'metrics/auc.metric' doesn't use cache. Skipping saving.
```

dvc run 명령을 이용하여 DAG라는 작업단위를 만들고 이를 작업이름.dvc로 저장합니다
이렇게 만들어진 아웃풋을 다시 dvc 명령으로 묶어서 파이프라인을 만듭니다.

만약에 파이프라인에서 처리하는 커맨드가 바뀌면?




```
(classify-test) seongju@seongJu-notebook:~/workspace/classify$ dvc repro evaluate.dvc
Stage 'extract.dvc' didn't change.
Stage 'prepare.dvc' didn't change.
Stage 'split.dvc' didn't change.
WARNING: Dependency 'code/featurization.py' of 'featurize.dvc' changed because it is 'm
odified'.
WARNING: Stage 'featurize.dvc' changed.
Running command:
        python code/featurization.py data/Posts-train.tsv data/Posts-test.tsv data/matr
ix-train.pkl data/matrix-test.pkl
The input data frame data/Posts-train.tsv size is (19999, 3)
The output matrix data/matrix-train.pkl size is (19999, 5002) and data type is float64
The input data frame data/Posts-test.tsv size is (5001, 3)
The output matrix data/matrix-test.pkl size is (5001, 5002) and data type is float64
Saving information to 'featurize.dvc'.
WARNING: Dependency 'data/matrix-train.pkl' of 'train.dvc' changed because it is 'modif
ied'.
WARNING: Stage 'train.dvc' changed.
Running command:
        python code/train_model.py 20170426
Input matrix size (19999, 5002)
X matrix size (19999, 5000)
Y matrix size (19999,)
Saving information to 'train.dvc'.
WARNING: Dependency 'data/model.pkl' of 'evaluate.dvc' changed because it is 'modified'
.
WARNING: Stage 'evaluate.dvc' changed.
Running command:
        python code/evaluate.py
Output 'data/auc.metric' doesn't use cache. Skipping saving.
Saving information to 'evaluate.dvc'.

To track the changes with git, run:

        git add evaluate.dvc train.dvc featurize.dvc
```

의존 데이터가 바뀌거나, 관련된 커맨드의 코드가 바뀐경우에는 **dvc repro [해당 DAG].dvc** 명령을 사용합니다. 명령이 실행되면 dvc가 변경된 내용을 찾아서 다시 파이프라인를 정의할때 등록했던 작업들을 수행합니다. 이 때 변경된 내용이 DAG에 대해서는 별도의 작업을 다시 수행하지 않습니다.

완성된 파이프라인을 시각화 해보죠!

dvc pipeline [DAG].dvc

Hoxy.. 여기서 끝인가요?

```
(classify-test) seongju@seongJu-notebook:~/workspace/classify$ dvc run -d code/evaluate
.py -d data/model.pkl -d data/matrix-test.pkl          -M metrics/auc.metric
 -f evaluate.dvc          python code/evaluate.py
Running command:
      python code/evaluate.py
Output 'metrics/auc.metric' doesn't use cache. Skipping saving.
```

dvc run에서 -M 옵션을 준 항목에 대해서 메트릭 정보를 확인할 수 있습니다.
다만 코드 상에서 메트릭 정보를 해당 경로의 파일로 저장해주어야 합니다.

dvc metrics show 라는 명령으로 해당 작업의 매트릭을 확인할 수 있습니다.

```
(classify-test) seongju@seongJu-notebook:~/workspace/classify$ dvc metrics show -a
working tree:
      data/auc.metric: AUC: 0.587951
master:
      data/auc.metric: AUC: 0.587951
```

이제 정말 끝이죠?

네 끝입니다.

봐주셔서 감사합니다.