Project: Sentiment Analysis datascience pipeline

Module : Big Data Programming 2

Submitted by :

Prabhupad Pradhan

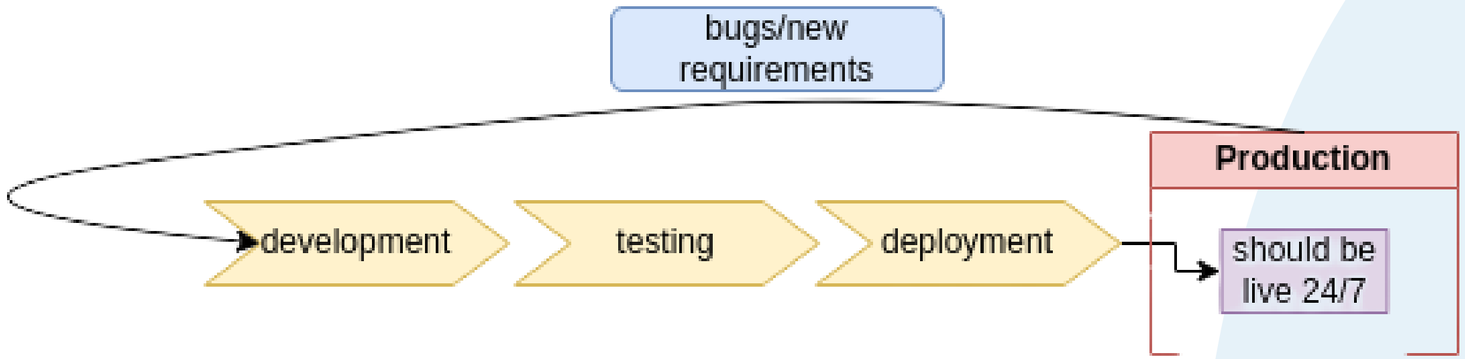
Ahmad Saud Azmi

Monica Suresh

Sharath Hagaranur

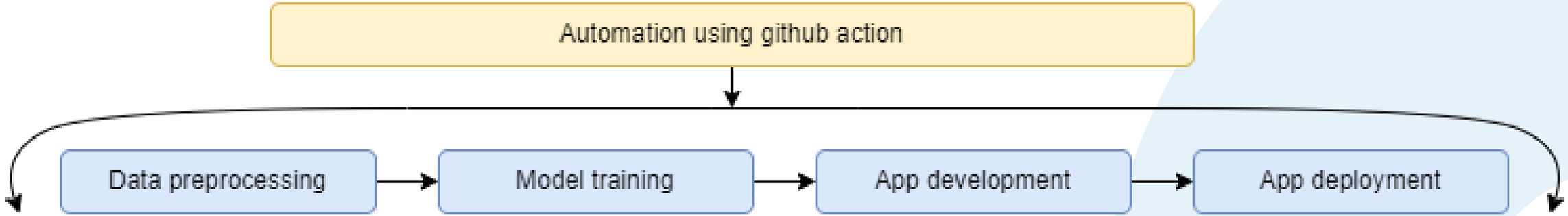
Problem statement

Company ABC. seeks to develop a real-time Twitter sentiment analysis system with a strong emphasis on scalability, high availability, and continuous machine learning model improvement. This system will process Twitter data in real-time, ensuring uninterrupted production even under high loads. Key objectives include the deployment of updated models without downtime, efficient data handling, monitoring, maintainability. The system will also offer a user-friendly interface.



Solution

Our solution leverages modern tools and technologies to build a robust real-time Twitter sentiment analysis system. We utilize GitHub Actions for streamlined code integration and deployment, harness the power of a Kubernetes cluster deployed on Google Kubernetes Engine for scalability, and employ Python as the primary coding language. The heart of our system is a fine-tuned BERT language model, renowned for its accuracy in classifying tweets into positive and negative sentiments. This combination of cutting-edge tools and a powerful machine learning model ensures seamless scalability, high availability, and continuous model improvement. Our solution not only meets the demands of real-time Twitter data processing but also provides valuable insights to users.



Framework components

sa_app
Model training code
Model inference code (flask app)

dasbhboard (streamlit)
call inference api
plotting
call mongo service

mongo service
/insert_tweet_data
/update_tweet_sentiment

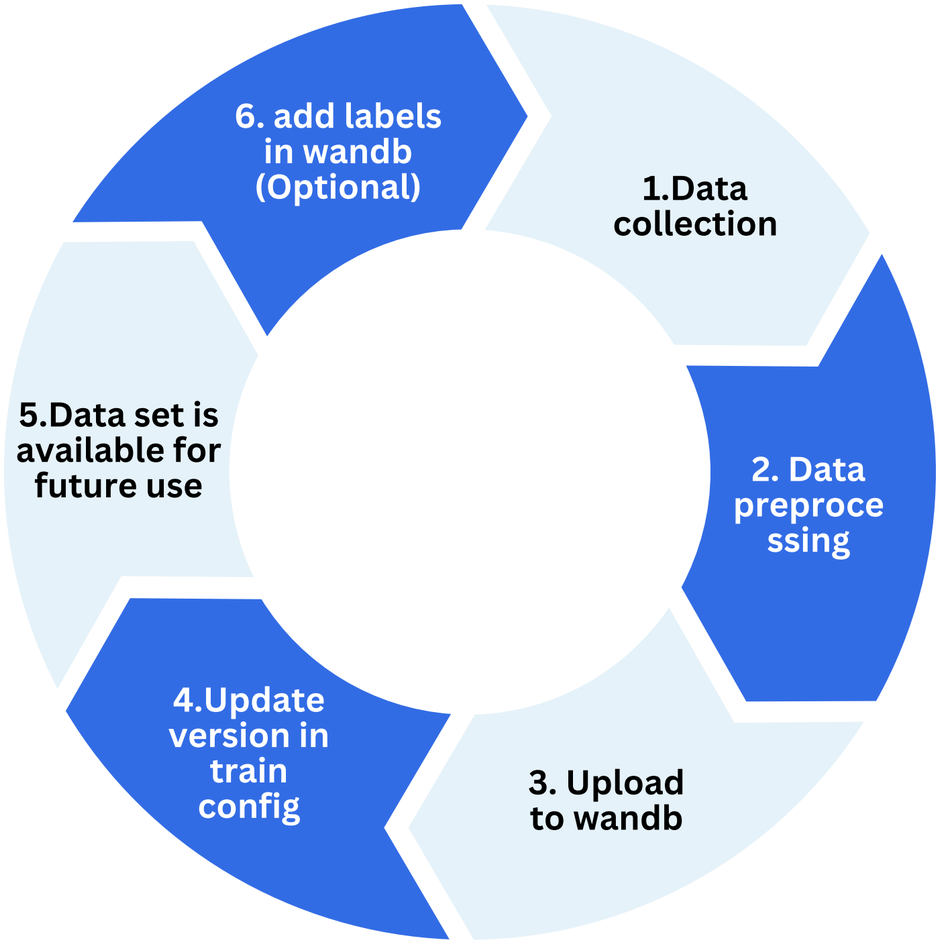
PyTorch Lightning

Hugging Face

Streamlit

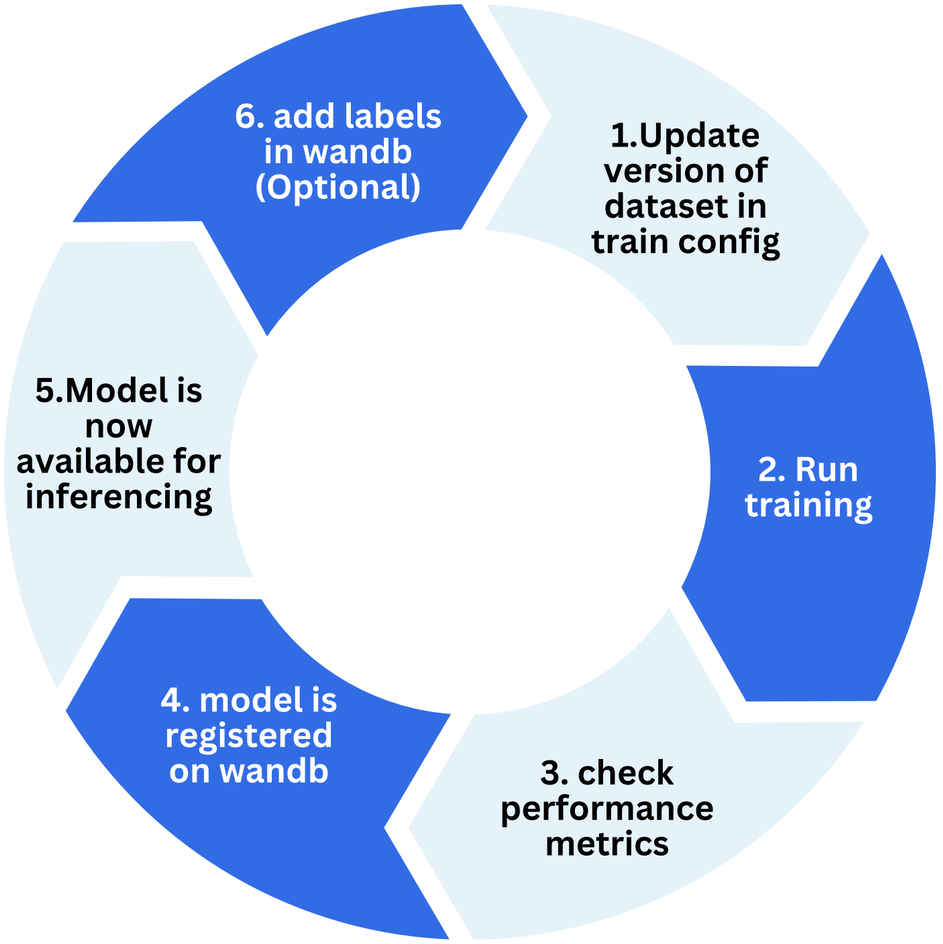
mongoDB

Dataset version control using wandb

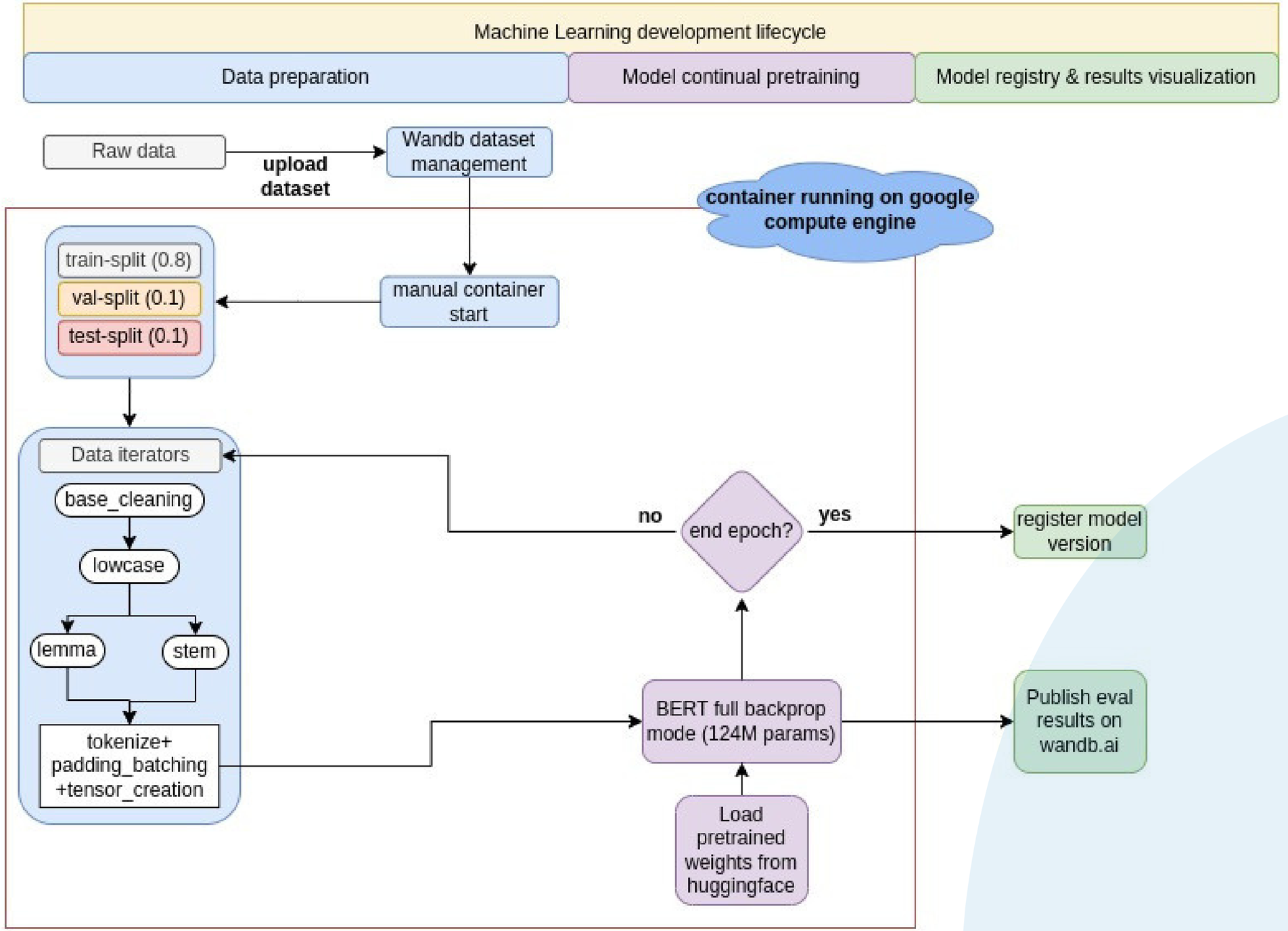


This is a manual process which is to be done by the developer who is trying to train the language model with their own dataset, the ad-hoc script will help in uploading the dataset to the wandb repo.

ML Models version control with wandb



After all the training epochs are finished running the training module takes care of saving the model to the local storage during the run and then finally uploads it to the wandb's model registry service which helps in creating the version of the model.


Machine Learning development lifecycle
Data preparation
Model continual pretraining
Model registry & results visualization
Raw data
upload dataset
Wandb dataset management
container running on google compute engine
train-split (0.8)
val-split (0.1)
test-split (0.1)
manual container start
Data iterators
base_cleaning
lowcase
lemma
stem
tokenize+ padding_batching +tensor_creation
no
end epoch?
yes
register model version
BERT full backprop mode (124M params)
Publish eval results on wandb.ai
Load pretrained weights from huggingface

Model Training & Inference process

Model training

Model inference

Loss function(Label, predicted)

Classification head

Result

Input text

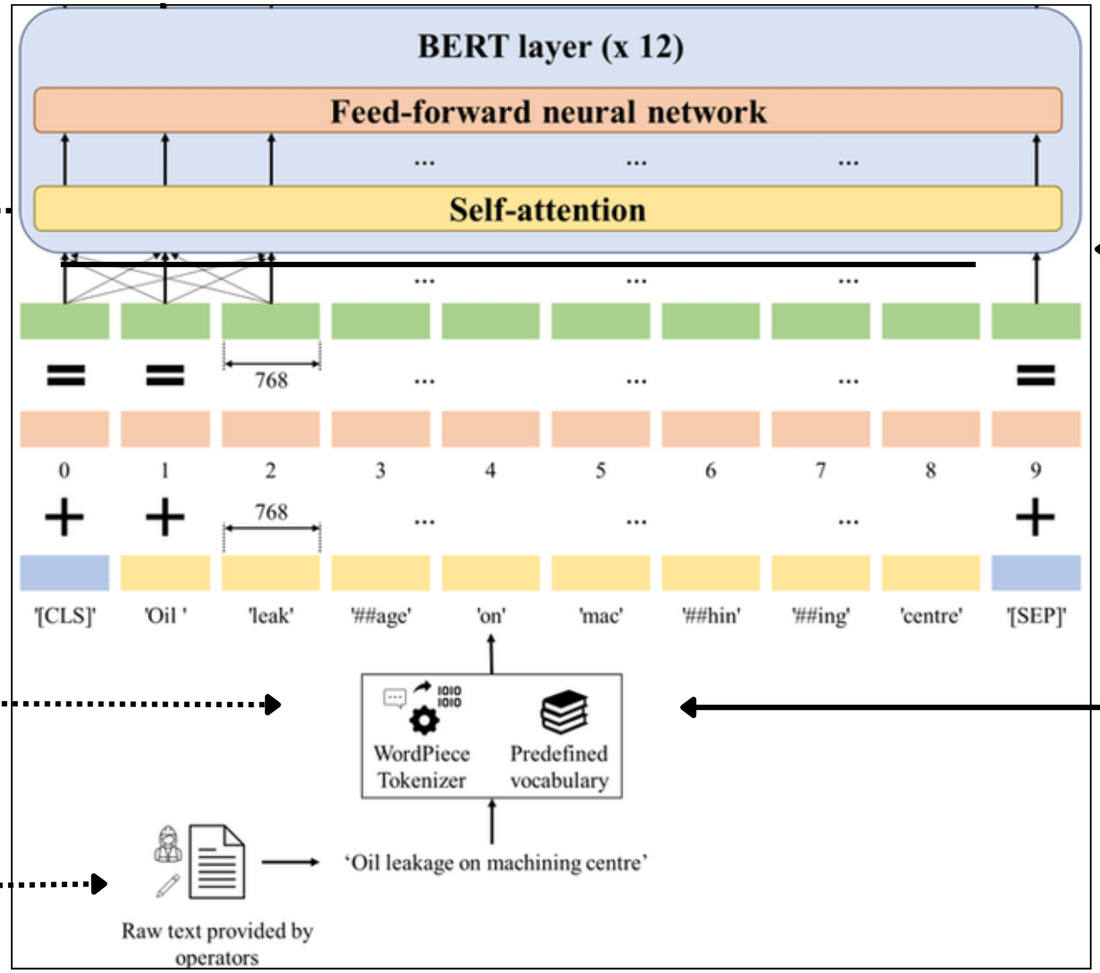
User gives input

Model version uploaded to wandb

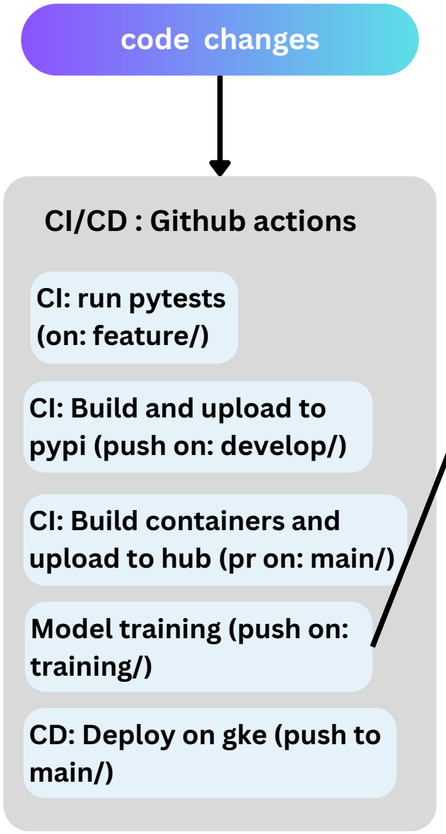
Backprop

Model version downloaded from wandb

Dataset version downloaded from wandb

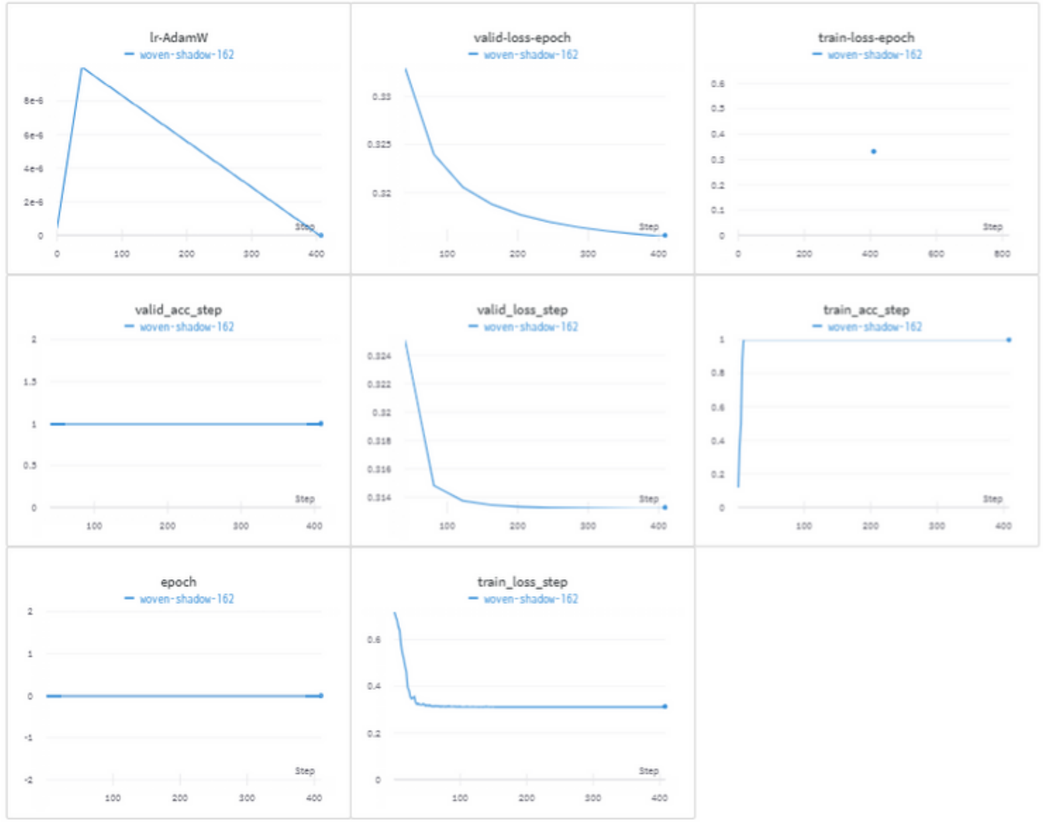


Application Deployment lifecycle

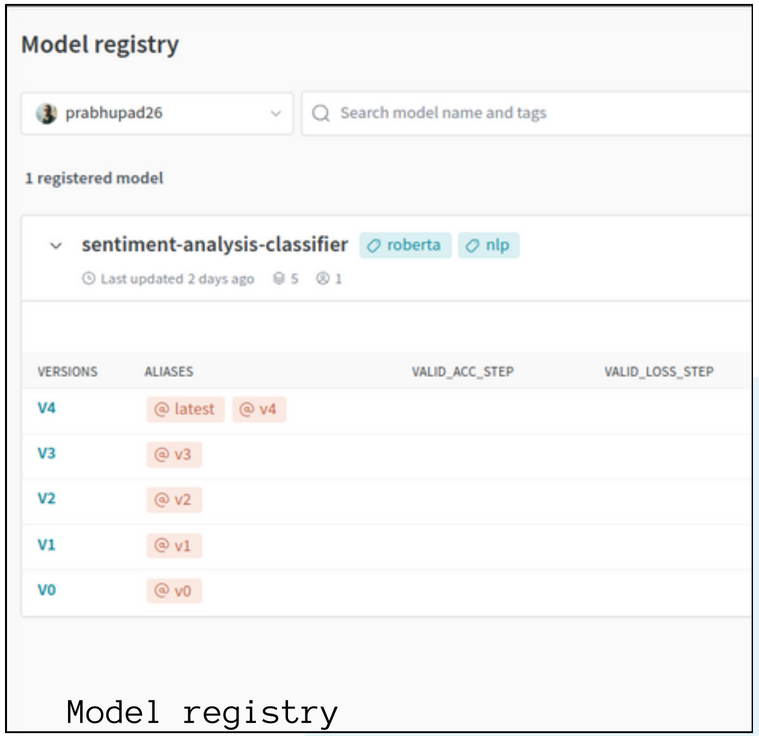


When the code is pushed to **training/** branch the this action will to run to create a vm machine using the google compute engine service and run the training module and then once completed it will stop and delete the vm machine.

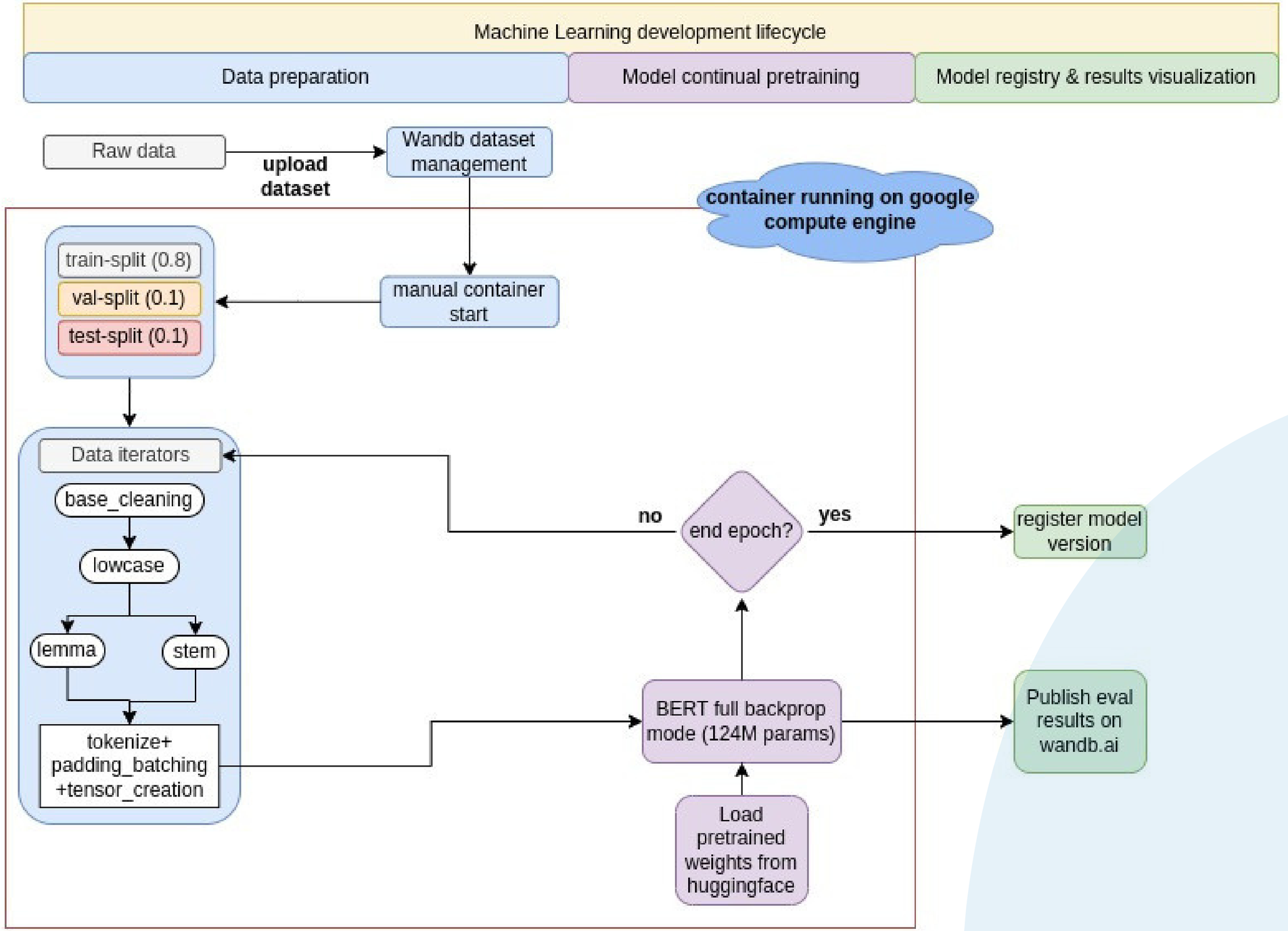
Model results



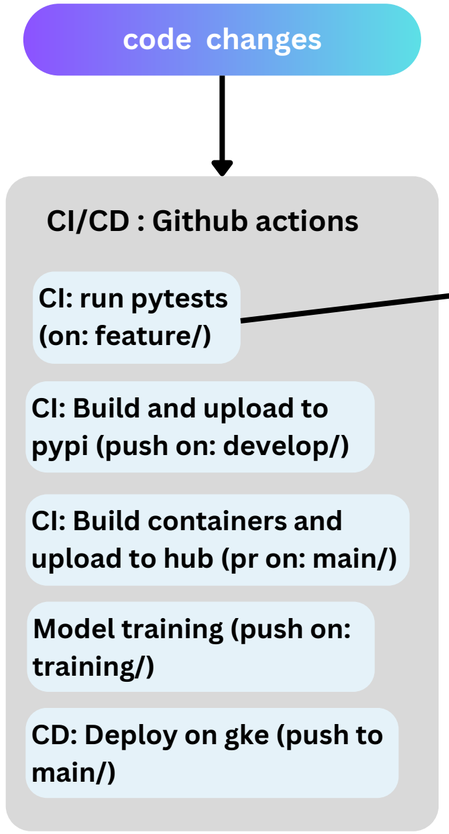
Wandb run details



Model registry

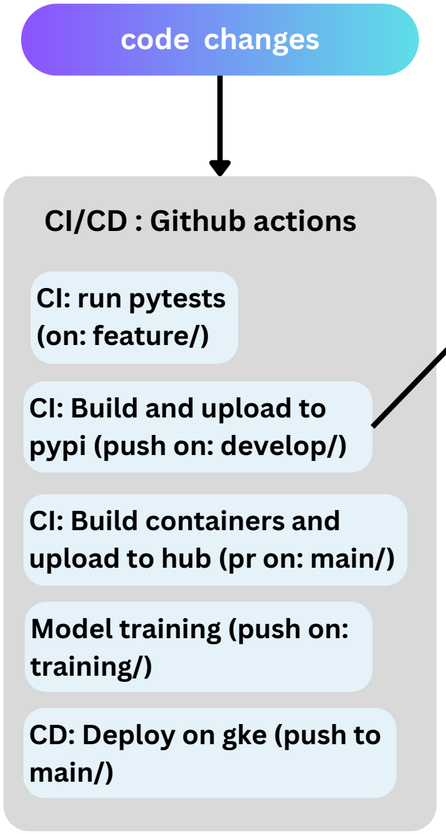

Machine Learning development lifecycle
Data preparation
Model continual pretraining
Model registry & results visualization
Raw data
upload dataset
Wandb dataset management
container running on google compute engine
train-split (0.8)
val-split (0.1)
test-split (0.1)
manual container start
Data iterators
base_cleaning
lowcase
lemma
stem
tokenize+ padding_batching +tensor_creation
no
end epoch?
yes
register model version
BERT full backprop mode (124M params)
Publish eval results on wandb.ai
Load pretrained weights from huggingface

Application Deployment lifecycle



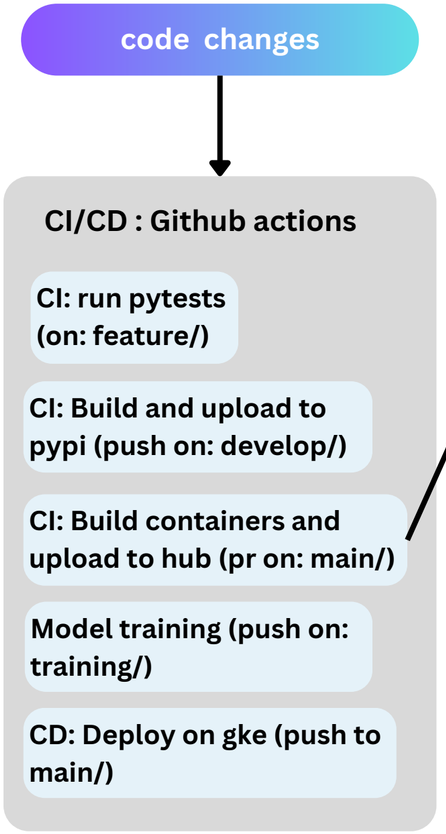
Once the code is checked in and pushed to the **feature/** branch the linter (pre-commit) will run and then the pytest will execute the unittests for various modules of the training and inference steps in the framework

Application Deployment lifecycle



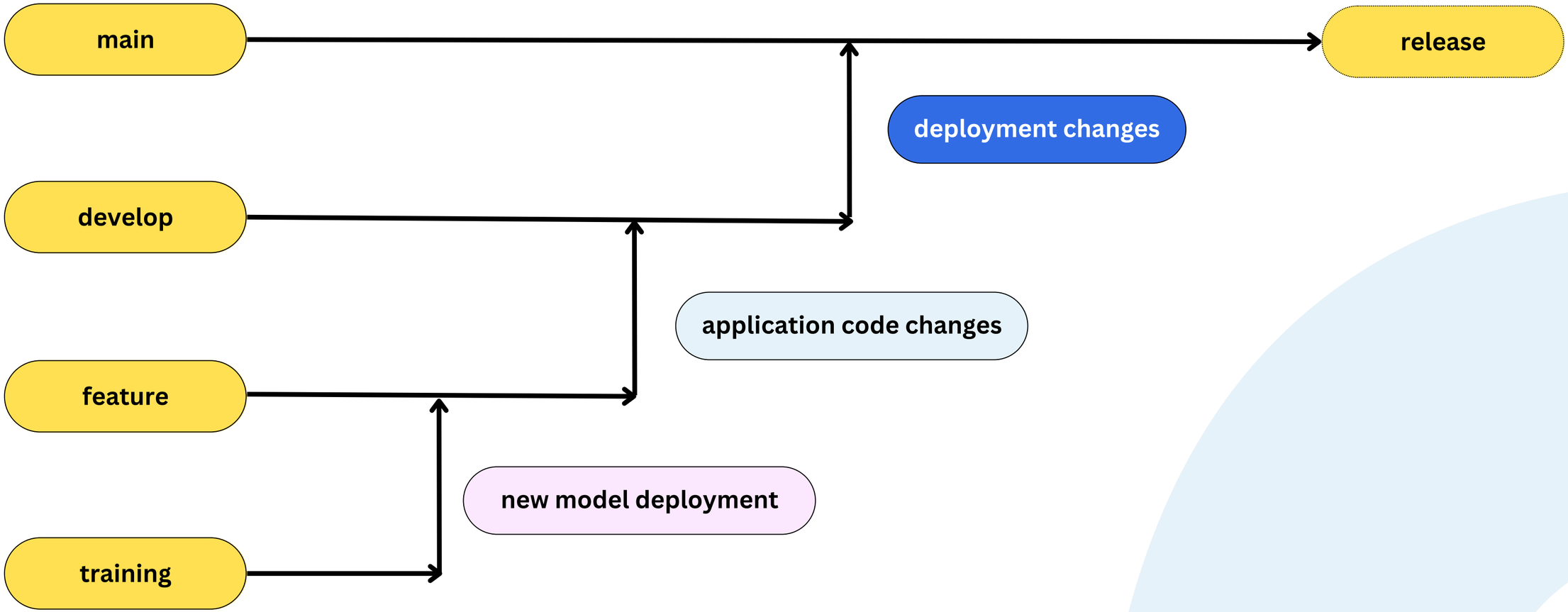
When the pull request is approved and merged to **develop/** branch the this action will run and then build the code using poetry and finally upload the package to pypi

Application Deployment lifecycle

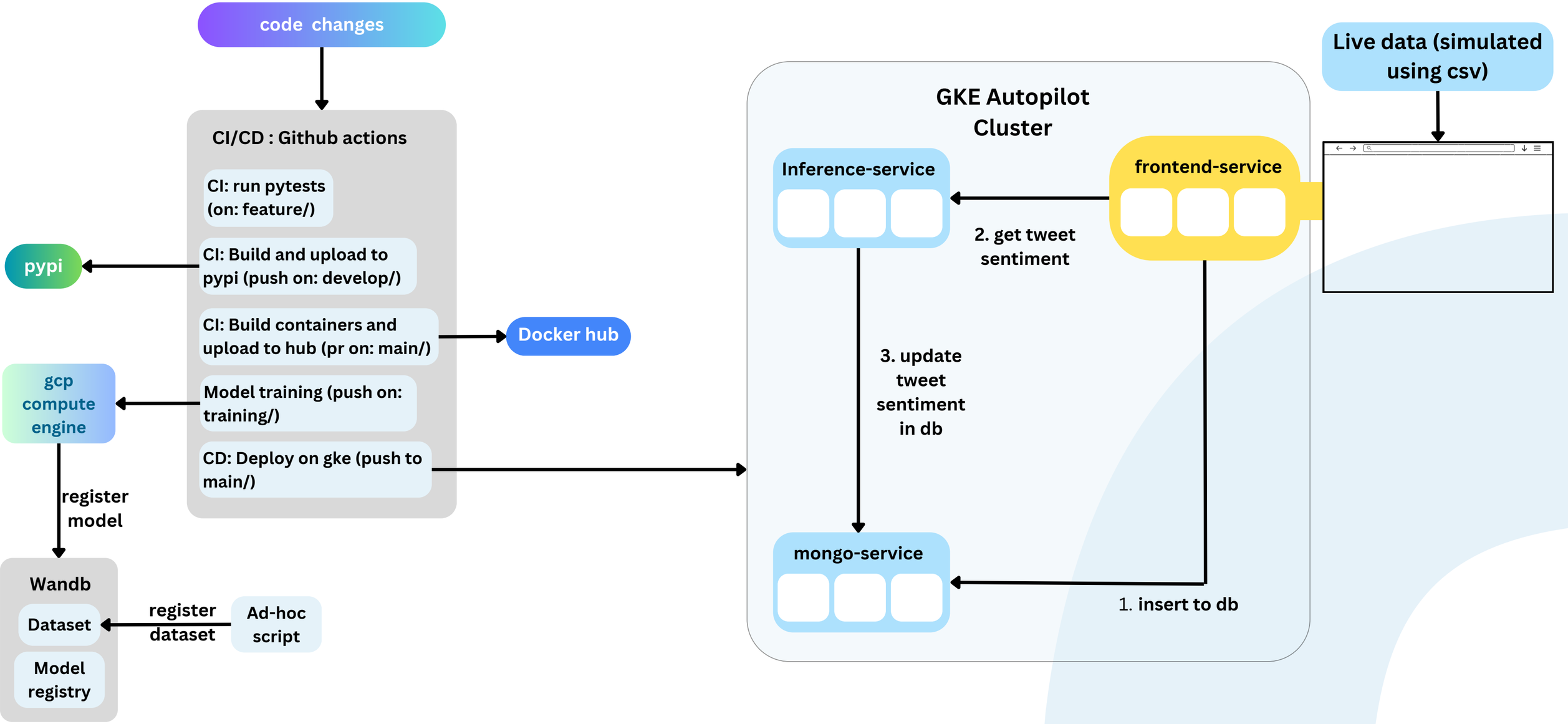


When the pull request raised on **main/** branch the this action will run and then docker image will be build and uploaded to **Docker hub** (for frontend, training, inference and mongo-service modules)

from development till deployment



Application Deployment lifecycle



Learnings

- Build a python scalable, maintainable and deployable application
- Using google cloud services such as : compute engine, kubernetes engine
- Using ML Ops tools:wandb dataset versioning service, model registry service, pytorch lightning support to publish results to wandb model tracking service.
- ML training code abstraction using pytorch-lightning framework.
- Using huggingface pretrained model for training the sentiment analysis language model.
- To ensure best python coding practices we used a linter (pre-commit) as well.

DEMO

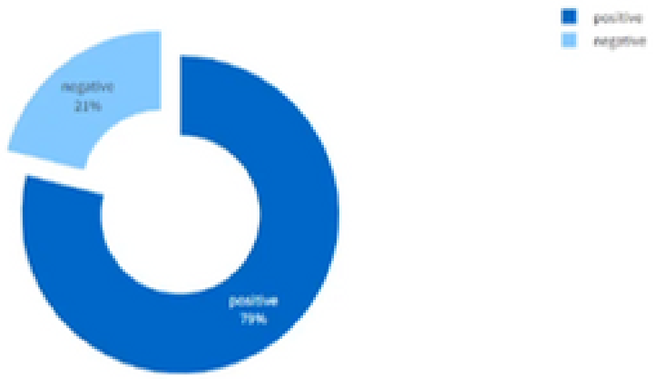
Sentiment Analysis Dashboard

Number of tweets to be analysed

100

Start collecting

Total tweets	Positive Count	Negative Count
100	79	21
↑ 100		



	0	1
0	Eating popcorn in the backyard w/ my family and trying to making it sound interesting (wished i went to dairy queen)	negative
1	So apparently my tan is more or less a sunburn.. It Itches lol Burnt my poor boobies real good too .. DANG!	negative
2	@qatv Ha! Loves it. Ur lady friend don't like me tho.	negative
3	I'll Take Yu Home If Yu Dont Leave Me At The Front Door!!! Shake It!! WooHoo!! @xOiLoveYouuOx	positive

Resources

- Model used : https://huggingface.co/cardiffnlp
- https://docs.github.com/en/actions/learn-github-actions
- https://kubernetes.io/docs/tasks/administer-cluster/
- https://kubernetes.io/docs/concepts/
- https://towardsdatascience.com/deploy-machine-learning-model-on-google-kubernetes-engine-94daac85108b
- https://cloud.google.com/deep-learning-containers/docs/kubernetes-container
- https://www.researchgate.net/figure/Example-of-a-trained-BERT-for-text-classification_fig1_353419108
- https://chat.openai.com/chat