Stage 1: Data preparation (Day 1: Initial setup)

Main repository link:

- On top right corner next to the search bar link for the main repo is given or click here to see the main repo

> **Note**
>
> The main repo contains various branch so make sure you select a branch which has the latest commit to see the recent updates.

Steps and important commands to begin-

> **Note**
>
> Replace the text mentioned as `<some_txt>` with your preferred choice.

STEP 1 Create a new conda environment

- Open an anaconda prompt and create an environment -

```
conda create -n <your_env_name> python=3.7 -y
```

wafer2 Don't specify python version

- Activate the environment -

```
conda activate <your_env_name>
```

wafer2

STEP 2 Create a default structure

- Install cookiecutter template

```
pip install cookiecutter
```

- Start a new project

```
cookiecutter https://github.com/drivendata/cookiecutter-data-science
```

- After above step you'll be given options in the command line.

> a. project_name: mlops
>
> b. repo_name: mlops2
>
> c. author_name: zymxiaotie
>
> d. description: replicate mlops_main to solve app issues
>
> e. Select open_source_license: MIT
>
> f. s3_bucket `[Optional]`:
>
> g. Select python_interpreter: python3

Once you are done with above step you'll see a following directory structure inside a directory by your given *project_name*

Click to see Project Organization

```
├── LICENSE
├── Makefile           <- Makefile with commands like `make data` or `make train`
├── README.md          <- The top-level README for developers using this project.
├── data
│   ├── external       <- Data from third party sources.
│   ├── interim        <- Intermediate data that has been transformed.
│   ├── processed      <- The final, canonical data sets for modeling.
│   └── raw            <- The original, immutable data dump.
│
├── docs               <- A default Sphinx project; see sphinx-doc.org for details
│
├── models             <- Trained and serialized models, model predictions, or
model summaries
│
├── notebooks          <- Jupyter notebooks. Naming convention is a number (for
ordering),
│                         the creator's initials, and a short `-` delimited
description, e.g.
│                         `1.0-jqp-initial-data-exploration`.
│
├── references         <- Data dictionaries, manuals, and all other explanatory
materials.
│
├── reports            <- Generated analysis as HTML, PDF, LaTeX, etc.
│   └── figures        <- Generated graphics and figures to be used in reporting
│
├── requirements.txt   <- The requirements file for reproducing the analysis
environment, e.g.
│                         generated with `pip freeze > requirements.txt`
│
├── setup.py           <- makes project pip installable (pip install -e .) so src
can be imported
├── src                <- Source code for use in this project.
│   ├── __init__.py    <- Makes src a Python module
│   │
│   ├── data           <- Scripts to download or generate data
│   │   └── make_dataset.py
│   │
│   ├── features       <- Scripts to turn raw data into features for modeling
│   │   └── build_features.py
│   │
│   ├── models         <- Scripts to train models and then use trained models to
make
│   │   │                 predictions
│   │   ├── predict_model.py
│   │   └── train_model.py
│   │
│   └── visualization  <- Scripts to create exploratory and results oriented
visualizations
│       └── visualize.py
│
```

```
└── tox.ini            <- tox file with settings for running tox; see
tox.readthedocs.io
```

Now open the project in your favorite code editor.

STEP 3 Get the dataset

- Clone it from the dataset repository or directly-

> **Download Dataset**

- Now extract the *Prediction_Batch_files*, *Training_Batch_Files* directory in the root directory of the project

STEP 4 Initialize git in Current working directory in your terminal, command prompt or git bash.

$cd C:\Users\zymxi\mlops2

```
git init
```

> **Note**
>
> If git is not installed in your system then download it from GIT-SCM site

STEP 5 Install DVC and its gdrive extension

```
pip install dvc
pip install dvc[gdrive]
```

STEP 6 Initialize DVC

```
dvc init
```

STEP 7 Add data into dvc for tracking

```
dvc add Training_Batch_Files/*.csv Prediction_Batch_files/*.csv
```

> **Warning**
>
> Above command will not work for windows users so they can create and run the following file in the root of their project

```
# NOTE: For windows user-
# This file must be created in the root of the project
# where Training and Prediction batch file as are present

import os
from glob import glob


data_dirs = ["Training_Batch_Files","Prediction_Batch_files"]

for data_dir in data_dirs:
    files = glob(data_dir + r"/*.csv")
    for filePath in files:
        # print(f"dvc add {filePath}")
        os.system(f"dvc add {filePath}")

print("\n #### all files added to dvc ####")
```

STEP 8 Do the first commit and push to the remote repository

run below commands on by one -

```
git add . && git commit -m "first commit and added raw data"
```

```
git branch -M main
```

```
git remote add origin https://github.com/<USERNAME>/<REPONAME>.git
```

zymxiaotie/mlops_main.git

```
git push -u origin main
```

> **Note**
>
> replace `<USERNAME>` and `<REPONAME>` as per you.

STEP 9 Create and checkout a development branch for our development

```
git checkout -b dev
```

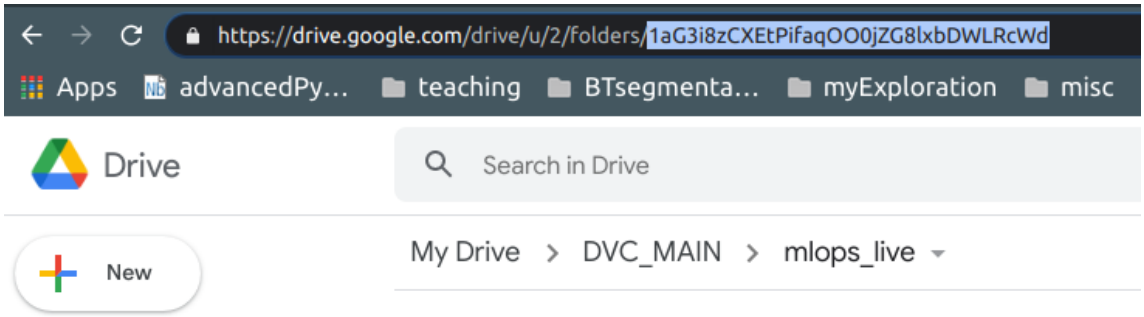
STEP 10 Add remote storage

```
dvc remote add -d storage gdrive://<DRIVE ID>

git add .dvc/config && git commit -m "Configure remote storage"
```

Note

Get the as shown in the highlighted part in the below screenshot-



STEP 11 Push the data to the remote storage-

```
dvc push
```

- This step will ask you to authenticate yourself by clicking on the link which will appear in the terminal.
- Once you allow dvc to read and write on gdrive it'll give an access token which you'll paste in the terminal.
- Now the copy of your data will be pushed to the gdrive
- Above step will create a gdrive credential file (Now check next step).

STEP 12 Add Gdrive credential secrets in github repo secrets.

- Find this credentials in the given path -
  I didn't find this file, skip step 12
  `.dvc >> temp >> gdrive-user-credentials.json`
- Now to add the secrets in your github repo -
  - Go to settings
  - secrets
  - Click on add secrets
  - Give name of secretes
  - Paste the json file content from `gdrive-user-credentials.json`

> **To retrieve data anytime**
>
> ```
> dvc pull
> ```
>
> refer dvc-data-versioning to know more

STEP 13 Install full requirements.txt as given in the repository

```
pip install -r requirements.txt
```

> **One line readme update and push command to dev branch-**
>
> ```
> git add README.md && git commit -m "update readme" && git push origin dev
> ```

STEP 14 Now you can follow along after this point as shown in the following video -