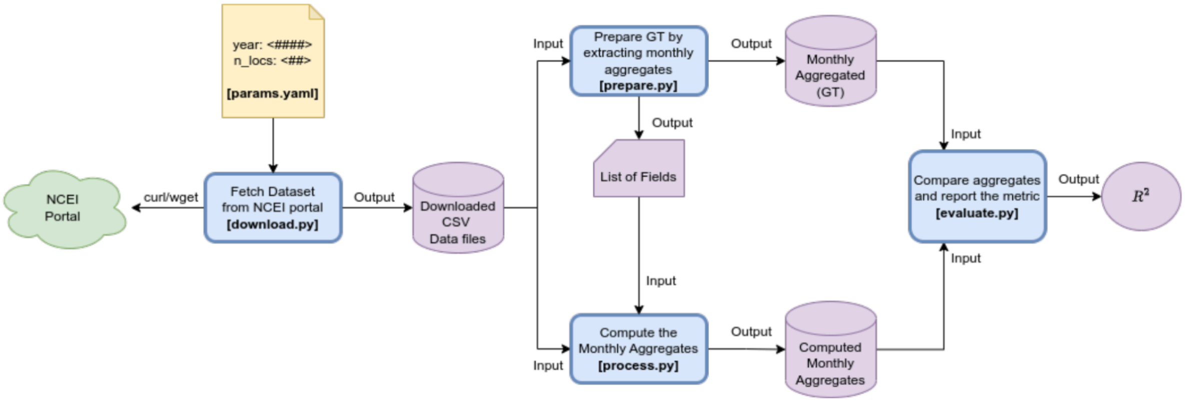
BDL: Assignment 4

Nikhil Anand BE20B022

Please go to this github repository to see the project:
https://github.com/nikhilanand03/CS5830_DVC/tree/main?tab=readme-ov-file

I have implemented the pipeline and each of the py files in the repository as well as connected it to dvc. Simply running dvc repro will produce the pipeline given in the question.

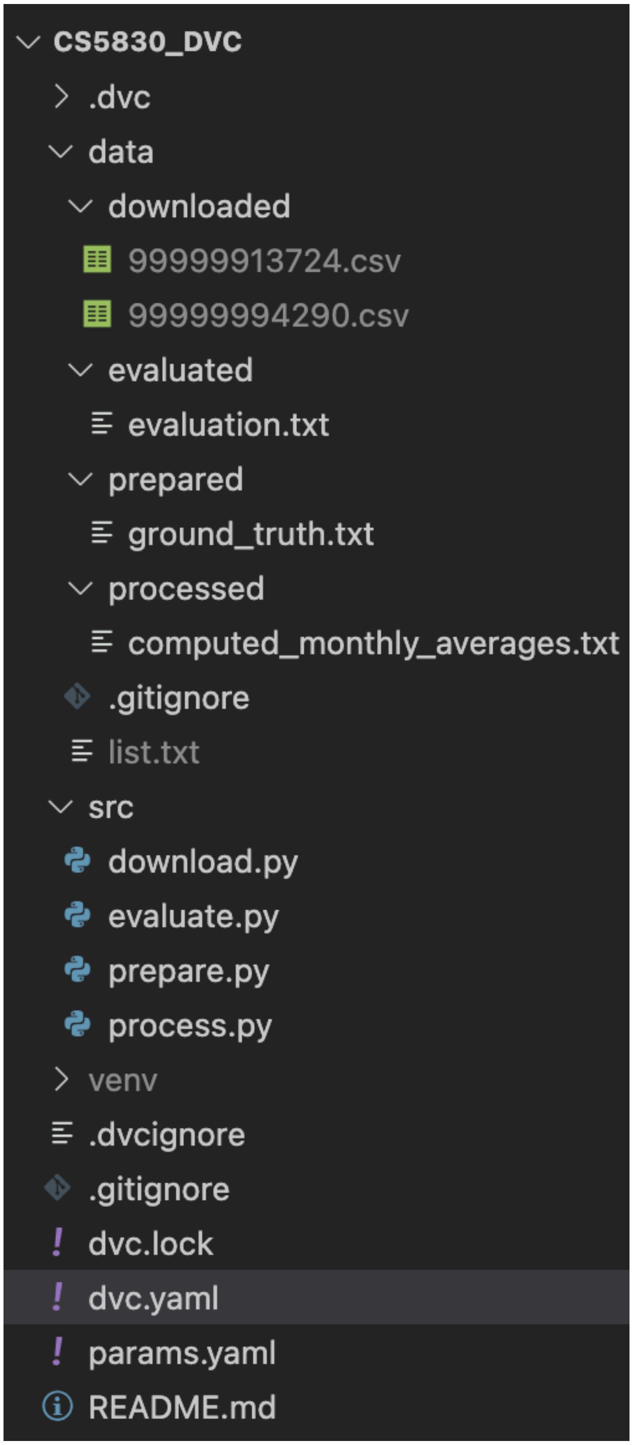
Running `dvc repro` will reproduce the pipeline below:



year: <####>
n_locs: <##>
[params.yaml]
NCEI Portal
curl/wget
Fetch Dataset from NCEI portal [download.py]
Output
Downloaded CSV Data files
Input
Prepare GT by extracting monthly aggregates [prepare.py]
Output
Monthly Aggregated (GT)
Output
List of Fields
Input
Input
Compute the Monthly Aggregates [process.py]
Output
Computed Monthly Aggregates
Input
Compare aggregates and report the metric [evaluate.py]
Input
Output
R^2

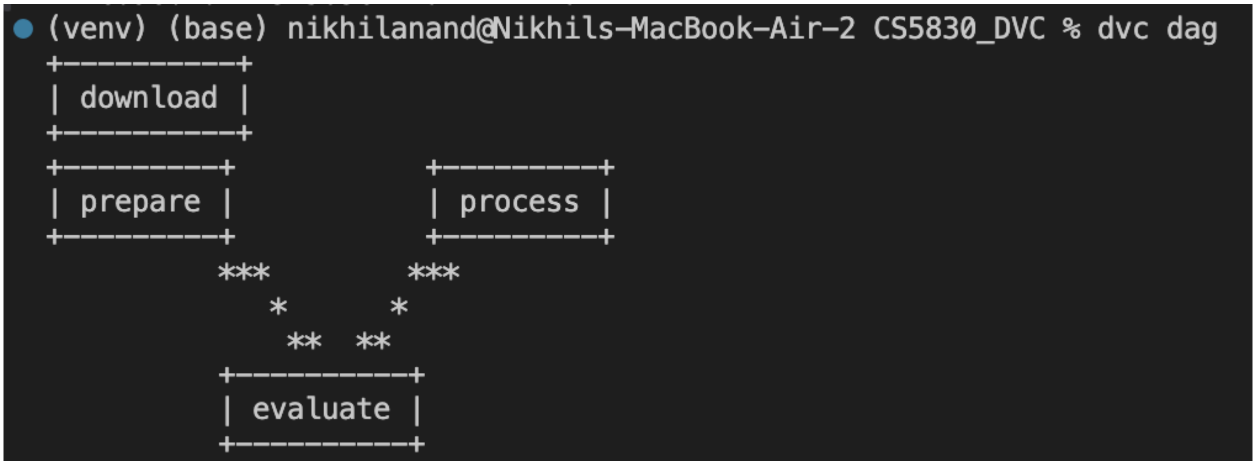
1. I have gone through the slides and understood how to create a pipeline on dvc. Some basic steps are:
    a. ‘Dvc get’ can be used to fetch data, and dvc add tracks the data file on dvc by ensuring it is in .gitignore but a pointer is created in a .dvc file that helps refer to the file.
    b. A virtual environment venv must be created in the directory.
    c. We first write out the code files that represent the tasks performed in the pipeline.
    d. Then, we create a pipeline of what is run in what order, using dvc stage add commands. This updates a dvc.yaml file showing the pipeline.
    e. The dvc repro command then runs the stages.

2. Git and dvc have been installed
3. Project has been created here.
4. DVC has been initialised.
5. Stages have been created as can be seen in the dvc.yaml file.



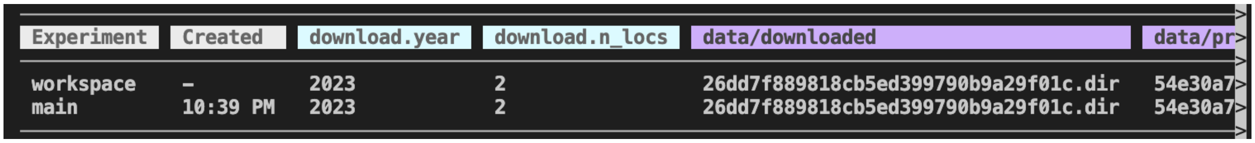
```
CS5830_DVC
  .dvc
  data
    downloaded
      99999913724.csv
      99999994290.csv
    evaluated
      evaluation.txt
    prepared
      ground_truth.txt
    processed
      computed_monthly_averages.txt
    .gitignore
    list.txt
  src
    download.py
    evaluate.py
    prepare.py
    process.py
  venv
  .dvcignore
  .gitignore
  dvc.lock
  dvc.yaml
  params.yaml
  README.md
```

6. We run dvc dag to visualise the dag.



```
(venv) (base) nikhilanand@Nikhils-MacBook-Air-2 CS5830_DVC % dvc dag
+----------+
| download |
+----------+
+---------+           +---------+
| prepare |           | process |
+---------+           +---------+
          ***        ***
             *      *
              **  **
          +----------+
          | evaluate |
          +----------+
```

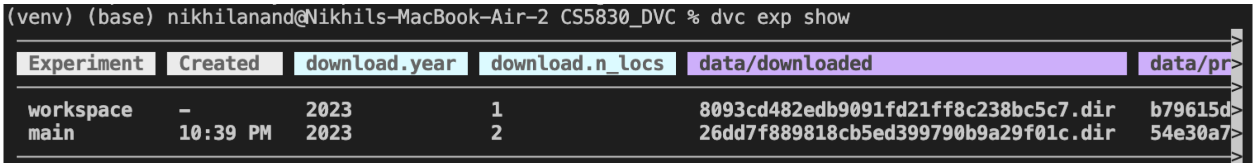
7. The dvc repro command helps us run the pipeline. We try this for cases n_locs = 1 and 2. Thus the parameters have been changed.
   Before changing parameters, the dvc exp show commands results in:



```
 Experiment   Created    download.year   download.n_locs   data/downloaded                         data/pr>

 workspace    -          2023            2                 26dd7f889818cb5ed399790b9a29f01c.dir    54e30a7>
 main         10:39 PM   2023            2                 26dd7f889818cb5ed399790b9a29f01c.dir    54e30a7>
```

8. After changing params, we see dvc exp show as:

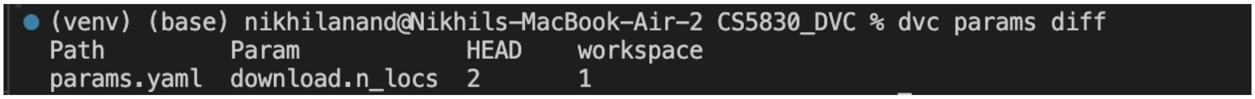


```
(venv) (base) nikhilanand@Nikhils-MacBook-Air-2 CS5830_DVC % dvc exp show

 Experiment   Created    download.year   download.n_locs   data/downloaded                         data/pr>

 workspace    -          2023            1                 8093cd482edb9091fd21ff8c238bc5c7.dir    b79615d>
 main         10:39 PM   2023            2                 26dd7f889818cb5ed399790b9a29f01c.dir    54e30a7>
```

9. We can use dvc params diff to see the differences.



```
(venv) (base) nikhilanand@Nikhils-MacBook-Air-2 CS5830_DVC % dvc params diff
Path         Param            HEAD    workspace
params.yaml  download.n_locs  2       1
```

The code files have been properly commented to help effectively understand them.