

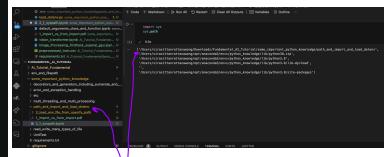
1. sys.path

sys.path in Python

Sys is a built-in Python module that contains parameters specific to the system i.e. it contains variables and methods that interact with the interpreter and are also governed by it.

sys.path

syspath is a built-in variable within the sys module. It contains a list of directories that the interpreter will search in for the required module.



- **DEFAULT-** By default, the interpreter looks for a module **within the current directory**. To make the interpreter search in some other directory you **just simply have to change the current directory**. The following example depicts a default path taken by the interpreter:

A screenshot of a Jupyter Notebook cell. The code imports 'sys' and prints 'sys.path'. The output shows the path: '/Users/rohit/PycharmProjects/Python_Lecture_1'. A blue arrow points from the text 'current path is ...' to the output. Another blue arrow points from 'current path is ...' to the line 'import sys'. A green circle highlights the output text. A red circle highlights the line 'import sys'.

```
import sys
print(sys.path)
```

Output:

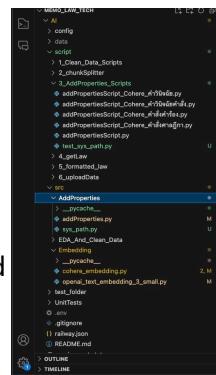
```
['/Users/rohit/PycharmProjects/Python_Lecture_1']
```

① → loan module's folder is memo_law_tech
↓
current path is ...
② → file

current: s>Add...
upt: script
upt: A2
upt: memo_law_tech

A module is a Python file that's intended to be imported into scripts or other modules.

A package is a collection of python files. These files are specifically placed within a folder. Ex, python class files dog.py, cat.py, and rat.py would be placed under folder animal, then folder animal would be the package.



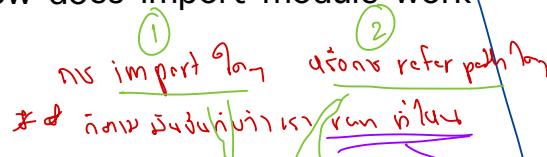
(file)
module → cohete_embedding.py

(folder)
package → AddProperties

2. How to set folder structure and how does import module work

Firstly, we need to understand that every path would relate to how(where) we run the script

I would use these 3 files as an example



module1 (as a script)

```
memo_low.py
def test():
    print("This is module1")
    print("This is module1")
    print("This is module1")
```

module2 (in src) (call module1)

```
memo_low_technology.py
from memo_low import test
test()
```

module3

```
memo_low_technology.py
from memo_low_technology import test
test()
```

When we run we would run at directory of module1 (scripts)

```
% (memo_low) sirasittamrattanawong@irisits-air ~_AddProperties_Scripts % python test_sys_path
project root is : ../../.
before sys.path is ['~/Users/sirasittamrattanawong/Downloads/memo_low/tech/ai/script/_AddProperties_Scripts', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8.zip', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8/lib-dynload', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8/site-packages']
Index 0 of sys.path (current path/dir) is : /Users/sirasittamrattanawong/Downloads/memo_low/tech/ai/script/_AddProperties_Scripts
After sys.path is ['../../', '/Users/sirasittamrattanawong/Downloads/memo_low/tech/ai/script/_AddProperties_Scripts', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8.zip', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8/lib-dynload', '/Users/sirasittamrattanawong/opt/anaconda3/envs/memo_low/tech/lib/python3.8/site-packages']

[0.0197408, 0.0362488, -0.0002957, 0.038223267, -0.00053652, 0.01549255, 0.0002063675, -0.037322996, -0.035951416, -0.02005398, -0.04647236, -0.007835388, -0.002668713, 0.
```

after run in folder AI_Scripts
with path injection
when we run the script
not path from that module.

now run 'ls -ld'
first location path in code

ກີ່ຈະເຮັດຕິດການຂົບສົກເກມ

1. ໃນ code ອາຍຸ module 1 ໂດຍ run ຮຽນ run ອັນ folder 3-A22 —

ໃຫ້ຈາກລາຍກື sys.path.insert ມີຄວາມ search ອີ່ memo_low_tech ດ້ວຍ

2. ; ໃນ module 2 ມີ import module 3 ປໍ່ນາງ

From AZ.svc.Embedding coherence-embedding

ໃຊ້ໄດ້ພູມໄຫວ່າພຽງວ່າ ລາຍລະອຽດ ມີຄວາມ search ອີ່ໂທ່ອນບັນຍາ

memo_low_tech ວິທີອຸ່ນຍຸ່ນຕັ້ງ run script
ດີຫຼັມເນັ້ນ 3-A22 — ໃນ
ຖຸກຕາມຕຳຫຼາດໃຫຍ່
path ໃຫຍ່ —>

3. In modules

```
dotenv_path = Path('.../../.env')
config = load_dotenv(dotenv_path=dotenv_path)
```

ກ່າວົກສອງ ພົມທີ່ໃນ run script ດີກົດ

ໃນ folder 3-A22 —

ກ່າວົກສອງ ພົມທີ່ໃນ modules ໄດ້ປັບປຸງກົດ

• occasions to make us had practice
inflating sympathetic to depolarize

3. Questions

1. The module

在終端 sys.path.insert(0, '/usr/lib/jvm')

j memo - leu - koh (1m)

କ୍ରିଏଟିଭ ପ୍ରାଣୀ ମୁଦ୍ରା ଲିମିଟେଡ

```
 3_AddProperties_Scripts/test_csys_path.py
[1]: python3 test_csys_path.py
['3_AddProperties_Scripts.py', 'Cohere_AltV1646.py', 'Cohere_AltV1646_2.py']
```

2. Web structure via ດາວໂຫຼດຂອງ ອົບ

3. $\sigma = 1 \text{ s.d}$ setup.py file

4. ດ້ວຍເຫັນ script ຮັດໃຫຍ່ໄປຂອງ ໄລັດ

run

ອີ່ນໃຈ run ທີ່ມານີ້ ຮັດໃຫຍ່ໄປ
ໄຕເຈົ້າໃຫຍ່

ພົກເວລັນໂລຣ ສcript ຮັດໃຫຍ່ໄປ

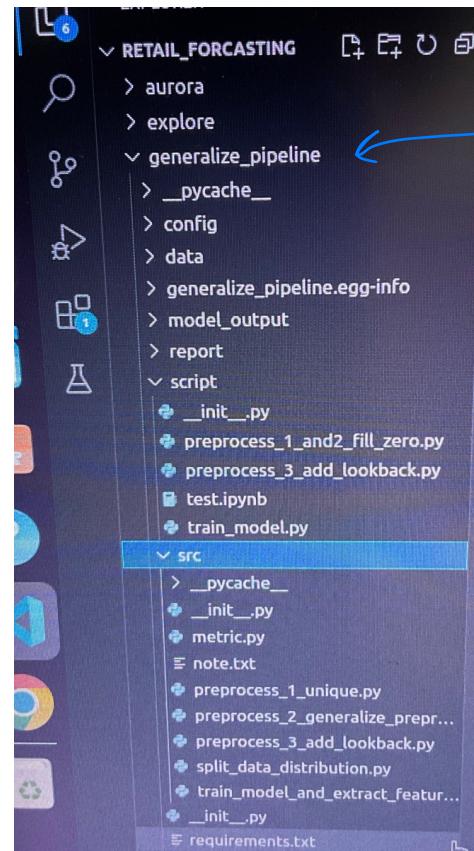
ເລີນແຈ້ງ import ພົກເວລັນໂລຣ input

ທຳມະນຸ

ພົນລະບຸຮັບກ່າຍ project ອະນຸມາຮັກ ເພື່ອອັດໃຈຮະຄົນຜູ້ອານຸ

an internship based on individual vision

→ impact $t + \Delta t$



Folder 2

মোবাইল

import src

١٦١

until now!!

```
> explore
  +-- generate_pipeline
    +-- __pycache__
    +-- __init__.py
    +-- generate_pipeline_esp_info
    +-- model_output
    +-- report
    +-- script
      +-- __init__.py
      +-- preprocess_1_and_fill_pseudo.py
      +-- preprocess_2_and_lookback.py
      +-- test.ipynb
    +-- train_model_and_extract_feature
      +-- __init__.py
      +-- requirement.txt
```

```
1   from generalize_pipeline.src.train_model_and_extract_fea
2   from generalize_pipeline.src.preprocess_1.unique import
3
4   def main():
5     unique_processing = UniqueProcessing()
6     read_path = f'./generalize_pipeline/data/group_of_train'
7     group_df_train.withLookback.all_data_before_split =
8
9
10
11
12   average_total_months_back_baseline0 = 3
13   save_path_baseline0 = f'./generalize_pipeline/data/bas
14   overwritten_baseline0 = False
15
16
17   train_model_and_extract_feature = TrainModelAndExtractF
18   average_list, y_test_prediction, model_output = train_m
19   print(f'model output: {model_output}')
20   print(f'average_list: {average_list}, len(y_test_prediction) = {len(y_te
21
22
23   if __name__ == '__main__':
24     main()
```

Solve1. Set python anaconda path (from chatgpt first)

How Python search for imported module/package

<https://medium.com/@sachinsoni600517/how-python-search-for-imported-module-package-76cf0da5f690>

How to set environment variables in a conda virtual environment

<https://guillaume-martin.github.io/saving-environment-variables-in-conda.html>

set anaconda python path

<https://www.geeksforgeeks.org/how-to-setup-anaconda-path-to-environment-variable/>

1. Create conda env like before.
2. Conda env list to find path of conda env

```
1 day   ● (retail2) ( .venv ) loolootech@loolootech:~/Downloads/retail_forcasting$ conda env list
2 days
# conda environments:
#
base          /home/loolootech/anaconda3
italasia      /home/loolootech/anaconda3/envs/italasia
retail2        * /home/loolootech/anaconda3/envs/retail2
retail_forcast_pipeline /home/loolootech/anaconda3/envs/retail_forcast_pipeline
```

3. set path → *will reference to the previous step*

2. Navigate to the Activation Directory:

Conda environments have a directory for activation scripts. We will create one if it doesn't exist.

```
sh
mkdir -p /home/loolootech/anaconda3/envs/retail_forcast_pipeline/etc/conda/activate.d
```

3. Create the Activation Script:

Create a new script file named 'env_vars.sh' in the 'activate.d' directory.

```
sh
vim /home/loolootech/anaconda3/envs/retail_forcast_pipeline/etc/conda/activate.d/env_vars.sh
```

4. Add the Following Content to Set the 'PYTHONPATH':

Press 'i' to enter insert mode, then add the following line:

```
sh
export PYTHONPATH=/home/loolootech/Downloads/retail_forecasting:$PYTHONPATH
```

5. Save and Close the File:

- Press Esc to exit insert mode.
- Type :wq and press Enter to save and quit vim.

3. Create Deactivation Script (Optional but Recommended)

To clean up the environment variable when you deactivate the Conda environment, you can create a deactivation script.

1. Navigate to the Deactivation Directory:

```
sh
mkdir -p /home/loolootech/anaconda3/envs/retail_forcast_pipeline/etc/conda/deactivate.d
```

2. Create the Deactivation Script:

Create a new script file named 'env_vars.sh' in the 'deactivate.d' directory.

```
sh
vim /home/loolootech/anaconda3/envs/retail_forcast_pipeline/etc/conda/deactivate.d/env_vars.sh
```

3. Add the Following Content to Unset the 'PYTHONPATH':

Press 'i' to enter insert mode, then add the following line:

```
sh
unset PYTHONPATH
```

4. Save and Close the File:

 - Press Esc to exit insert mode.
 - Type :wq and press Enter to save and quit vim.

Running Your Script

Now, whenever you activate your Conda environment, the 'PYTHONPATH' will be set automatically.

1. Activate Your Conda Environment:

```
sh
conda activate retail_forcast_pipeline
```

2. Navigate to Your Project Directory:

```
sh
cd /home/loolootech/Downloads/retail_forecasting
```

3. Run Your Script:

```
sh
python generalize_pipeline/script/preprocess.py
```

reference

Now, we need to create scripts that will set our environment variables whenever we activate our virtual environment and unset them when we deactivate. The path to our environment directory is set as \$CONDA_PREFIX. We cd to that directory and create a /etc/conda/activate.d and a /etc/conda/deactivate.d directory:

```
(my_environment)$ cd $CONDA_PREFIX
(my_environment)$ mkdir -p ./etc/conda/activate.d
(my_environment)$ mkdir -p ./etc/conda/deactivate.d
```

In each of those directories, we create a env_var.sh script where we'll write the commands to set and unset our variables:

```
(my_environment)$ touch ./etc/conda/activate.d/env_vars.sh
(my_environment)$ touch ./etc/conda/deactivate.d/env_vars.sh
```

or run pip install -r requirements.txt

```
RETAIL_FORCASTING
    > aurora
    > explore
    > generalize_pipeline
        > __pycache__
        > config
        > data
        > generalize_pipeline.egg-info
        > model_output
        > report
    > script
        > __init__.py
        > preprocess_1_and2_fill_zero.py
        > preprocess_3_add_lookback.py
        > test.ipynb
        > train_model.py
    > src
        > __pycache__
        > __init__.py
        > metric.py
        > note.txt
        > preprocess_1_unique.py
        > preprocess_2_generalize_prep...
        > preprocess_3_add_lookback.py
        > split_data_distribution.py
        > train_model_and_extract_featur...
        > __init__.py
        > requirements.txt
    > generalize_pipeline.egg-info
    > italiasa
```

```
generalize_pipeline > script > train_model.py > main
1   from generalize_pipeline.src.train_model_and_extract_feature import TrainModelAndExtractFeature
2   from generalize_pipeline.src.preprocess_1_unique import UniqueProcessing
3
4   def main():
5       unique_processing = UniqueProcessing()
6       read_path = "./generalize_pipeline/data/group_df_train_with_lookback_all_data_before_split.csv"
7       group_df_train_with_lookback_all_data_before_split = unique_processing.read_file(read_path)
8
9
10      average_total_months_back_baseline0 = 3
11      save_path_baseline0 = "./generalize_pipeline/data/baseline_0_{average_total_months_back_baseline0}_months_
12      overwritten_baseline0 = False
13
14
15      train_model_and_extract_feature = TrainModelAndExtractFeature()
16      average_list, y_test_prediction, model_output = train_model_and_extract_feature.runall(group_df_train_with_lo
17
18      print(len(average_list), len(y_test_prediction))
19
20
21
22
23  if __name__ == "__main__":
24      main()
```

:, run script

ஒவ்வொரு போக்குவரத்து
இன் சிகிச்சை என்று

கீழ் run என்றால் பதில் இருக்கும்

```
(retail2) (.venv) loolootech@loolootech:~/Downloads/retail_forcasting$ pwd
/home/loolootech/Downloads/retail_forcasting
(retail2) (.venv) loolootech@loolootech:~/Downloads/retail_forcasting$ python generalize_pipeline/script/train_model.py
```

Important key

1. Set anaconda python path to retail_store

2.1 when we want to run anything we would run at retail_store folder ex. python generalize_pipeline/script/train_model.py

2.2 same as before path would relate to where we run the script.

2.3 we set anaconda python path and run the script at retail_store folder when we import we would think that that is our folder path ex. from generalize_pipeline.src.train_model_and_extract_feature import TrainModelAndExtractFeature

```
RETAIL_FORCASTING
  > aurora
  > explore
  > generalize_pipeline
    > .pycache_
    > config
    > data
    > generalize_pipeline.egg-info
    > model_output
    > report
  > script
    > __init__.py
    > preprocess_1_and2_fill_zero.py
    > preprocess_3_add_lookback.py
    > test.ipynb
  > train_model.py
  > src
    > __init__.py
    > metric.py
    > note.txt
    > preprocess_1_unique.py
    > preprocess_2_generalize_prep...
    > preprocess_3_add_lookback.py
    > split_data_distribution.py
    > train_model_and_extract_featu...
    > __init__.py
    > requirements.txt
    > generalize_pipeline.egg-info
    > setup.py

generalize_pipeline > script > train_model.py > main
main.py
```

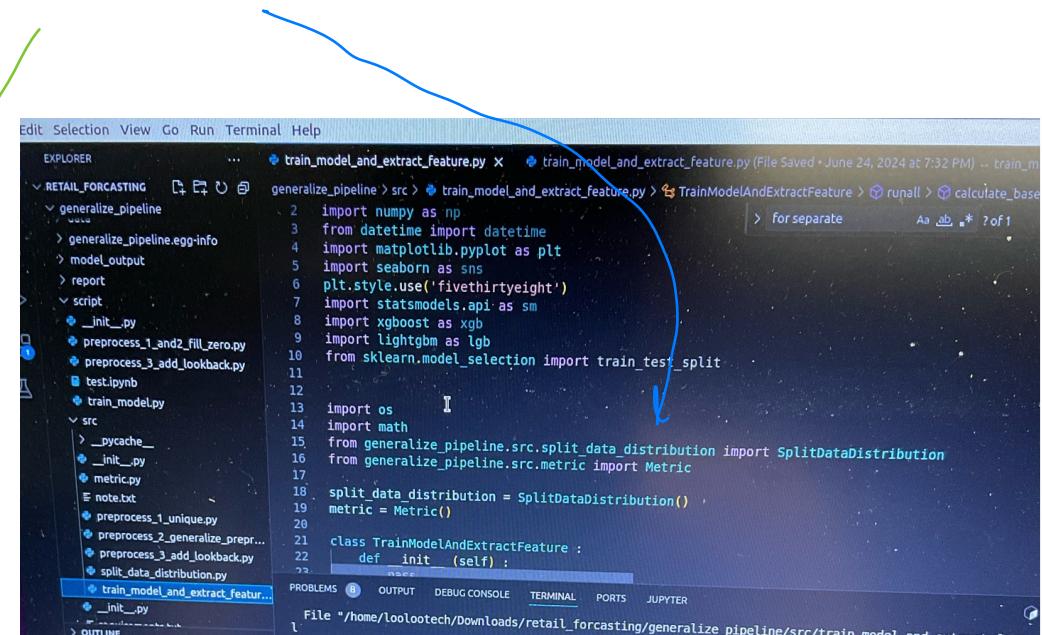
```
1 from generalize_pipeline.src.train_model_and_extract_feature import TrainModelAndExtractFeature
2
3 def main():
4     unique_processing = UniqueProcessing()
5     read_path = ".generalize_pipeline/data/group_df_train_with_lookback_all_data_before_split.csv"
6     group_df_train_with_lookback_all_data_before_split = unique_processing.read_file(read_path)
7
8     average_total_months_back_baseline0 = 3
9     save_path_baseline0 = "./generalize_pipeline/data/baseline_0_{average_total_months_back_baseline0}_months"
10    overwritten_baseline0 = False
11
12    train_model_and_extract_feature = TrainModelAndExtractFeature()
13    average_list, y_test_prediction, model_output = train_model_and_extract_feature.runall(group_df_train_with_lo
14
15    if __name__ == "__main__":
16        main()
```

```
RETAIL_FORCASTING
  > aurora
  > explore
  > generalize_pipeline
    > .pycache_
    > config
    > data
    > generalize_pipeline.egg-info
    > model_output
    > report
  > script
    > __init__.py
    > preprocess_1_and2_fill_zero.py
    > preprocess_3_add_lookback.py
    > test.ipynb
  > train_model.py
  > src
    > __init__.py
    > metric.py
    > note.txt
    > preprocess_1_unique.py
    > preprocess_2_generalize_prep...
    > preprocess_3_add_lookback.py
    > split_data_distribution.py
    > train_model_and_extract_featu...
    > __init__.py
    > requirements.txt
    > generalize_pipeline.egg-info
    > setup.py

generalize_pipeline > script > preprocess_3_add_lookback.py > main
main.py
```

```
1 from generalize_pipeline.src.preprocess_3_add_lookback import LookBack
2 from generalize_pipeline.src.preprocess_1_unique import UniqueProcessing
3
4 def main():
5     lookback = LookBack()
6     unique_processing = UniqueProcessing()
7     read_path = ".generalize_pipeline/data/encoded_df_fill_zero.csv"
8     encoded_df_fill_zero = unique_processing.read_file(read_path)
9
10    # this should be in config
11    save_path = ".generalize_pipeline/data/group_df_train_with_lookback_all_data_before_split.csv"
12    group_df_train_with_lookback_all_data_before_split = lookback.add_look_back(encoded_df_fill_zero, save
13
14    if __name__ == "__main__":
15        main()
```

```
1 from generalize_pipeline.src.preprocess_1.unique import UniqueProcessing
2 from generalize_pipeline.src.preprocess_2.generalize_preprocess_fill_zero import GeneralizePreprocessingFillZero
3
4 def main():
5     unique_processing = UniqueProcessing()
6     generalize_processing_fill_zero = GeneralizePreprocessingFillZero()
7
8     file_path = "./generalize_pipeline/data/ITALASIA SHOP DATA 2023.parquet" # this should be in config
9     original_df = unique_processing.read_file(file_path)
10    rename_dict = {
11        "សុខភាព": "date",
12        "តម្លៃ": "price",
13        "សុណុំ": "volume",
14        "តម្លៃ": "total_price",
15        "កម្មសាររបស់ VAT": "total_revenue",
16        "សុទ្ធសាន្ត": "unit",
17        "Outlet Name": "outlet",
18        "Outlet Code": "outlet_code",
19        "តម្លៃសុណុំ": "discount",
20        "សុខភាព": "is_return",
21        "SKU": "SKU ID",
22        "SKU Name": "SKU Name",      # តើជាប្រព័ន្ធដីណា (brand)
23        "Product Group": "Brand",
24    }
25
26    df = unique_processing.unique_preprocess(original_df, rename_dict)
27
28    # should be in config
29    selected_cols = ["outlet_code", "Brand", "Product Type", "SKU ID", "unit", "volume", "date"]
30    categorical_cols = ["Brand", "Product Type", "unit", "outlet_code"]
31    feature_cols = [col for col in selected_cols if col not in ["outlet_code", "SKU ID", "volume", "date"]]
32    cols_to_be_grouped_by_first = ["Brand", "Product Type", "unit"]
33    cols_to_be_grouped_by_sum = ["volume"]
34
35    save_path = "./generalize_pipeline/data/encoded_df_fill_zero.csv"
36
37    generalize_processing_fill_zero.run_all(df, selected_cols, categorical_cols, feature_cols, cols_to_be_grouped_by_first, c
38
39    # will track recent
40    you save them unless
41    leave as default in top
```



file path

Initialization → import path

Iteration → file path

```
● (retail2) (.venv) loolootech@loolootech:~/Downloads/retail_forcasting$ pwd  
/home/loolootech/Downloads/retail_forcasting  
● (retail2) (.venv) loolootech@loolootech:~/Downloads/retail_forcasting$ python generalize_pipeline/script/train_model.py
```

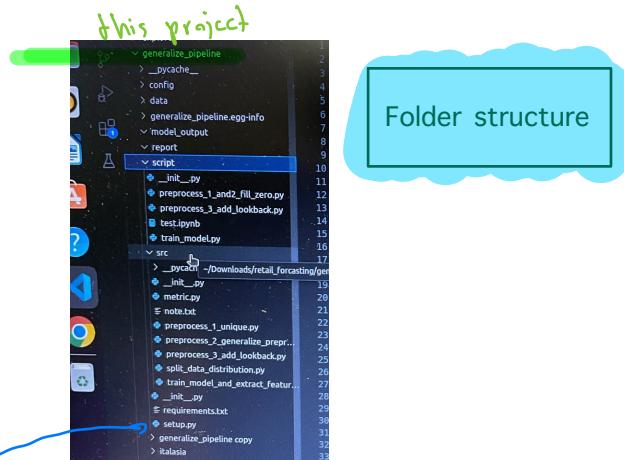

Solve2 setup.py

setup.py file python tutorial

<https://xebia.com/blog/a-practical-guide-to-using-setup-py/>



อันไหนเป็น folder แล้วจะเรียกไฟล์ในนั้นต้องใส่ `__init__.py` (แค่ไฟล์เปล่าๆนี่แหละ) ด้วย



setup.py

```
1 from setuptools import setup, find_packages
2
3 setup(
4     name='generalize_pipeline',
5     version='0.1',
6     packages=find_packages(), # Finds all packages under the current directory
7     python_requires='>=3.10', # Specify the required Python version
8     install_requires=[
9         'pandas==2.2.2',
10        'pyarrow==16.1.0',
11        'fastparquet==2024.5.0',
12        'matplotlib==3.9.0',
13        'plotly==5.22.0',
14        'nbformat==5.10.4',
15        'seaborn==0.13.2',
16        'statsmodels==0.14.2',
17        'xgboost==2.0.3',
18        'lightgbm==4.3.0',
19        'pyproject-toml==0.8.10',
20        'scikit-learn==1.5.0',
21    ],
22)
23
```

0. crack contd env.

1. Create setup.py file at root of the project (in `generalize_pipeline`) (1 step inside compared to before)
2. pip install -e . at the folder that consists of setup.py
3. Now path would be `generalize_pipeline` not `retail_store` so you need to change import code and path compared to above

In scripts folder example (one step inside compare to solve)

preprocess_1_and2_fill_zero.py

```
from src.preprocess_1.unique import UniqueProcessing
from src.preprocess_2.generalize_preprocess_fill_zero import GeneralizePreprocessingFillZero

def main():
    unique_processing = UniqueProcessing()
    generalize_processing_fill_zero = GeneralizePreprocessingFillZero()

    file_path = "./data/ITALASIA SHOP DATA 2023.parquet" # this should be in config
    original_df = unique_processing.read_file(file_path)
    rename_dict = [
        {"ชื่อ": "date",
        "รายละเอียด": "price",
        "จำนวน": "volume",
        "มูลค่า": "total_price",
        "มูลค่ารวม VAT": "total_revenue",
        "หน่วย": "unit",
        "Outlet Name": "outlet",
        "Outlet Code": "outlet_code",
        "ส่วนลด": "discount",
        "ผลิตภัณฑ์": "is_return",
        "SKU": "SKU ID",
        "SKU Name": "SKU Name", # ชื่อในชิ้นงาน (brand)
        "Product Group": "Brand", }
    ]
    df = unique_processing.unique_preprocess(original_df, rename_dict)

    # should be in config
    selected_cols = ["outlet_code", "Brand", "Product Type", "SKU ID", "unit", "volume", "date"]
    categorical_cols = ['SKU ID', 'Brand', 'Product Type', 'unit', 'outlet_code']
    feature_cols = [col for col in selected_cols if col not in ['outlet_code', 'SKU_ID', 'volume', 'date']]
    cols_to_be_grouped_by_first = ['Brand', 'Product Type', 'unit']
    cols_to_be_grouped_by_sum = ['volume']
    save_path = "./data/encoded_df_fill_zero.csv"
    generalize_processing_fill_zero.run_all(df, selected_cols, categorical_cols, feature_cols, cols_to_be_grouped_by_sum)
```

```
che requirements.txt setup.py
tech@loolotech:~/Downloads/retail_forcasting/generalize_pipeline$ python script/preprocess_1_and2_fill_zero.py
```

run at

Config file

Method1

1. Write yaml file and load

```
RETAIL_FORECASTING
  > aurora
  > explore
  > generalize_pipeline
    > __pycache__
    > config
      ! preprocess1_config.yaml
      ! preprocess2_config.yaml
    > data
    > generalize_pipeline.egg-info
    > model_output
    > outputs
    > report
    > report.txt
    > script
      < __init__.py
      < preprocess_1_unique_script...
      < preprocess_2_fill_zero.py
      < preprocess_3_add_lookback.py
      test.ipynb
      train_model.py
    > src
    > temp
    > report.txt
    < __init__.py
    < .gitignore
  README.md
```

```
generalize_pipeline > config > ! preprocess2_config.yaml
1 # generalize_pipeline/config/preprocess2_config.yaml
2
3 group_outlet_code: true
4 file_path: "./data/preprocessed_data.csv"
5 save_path: "./data/encoded_df_fill_zero.csv"
6 selected_cols:
7   - SKU_ID
8   - year
9   - month
10  - day
11  - target
12  - Brand
13  - Product Type
14  - unit
15 categorical_cols:
16   - SKU_ID
17   - Brand
18   - Product Type
19   - unit
20 feature_cols:
21   - Brand
22   - Product Type
23   - unit
24 cols_to_be_grouped_by_first:
25   - Brand
26   - Product Type
27   - unit
28 cols_to_be_grouped_by_sum:
29   - target
```

relative to where you run
(from setup.py file)

search command : yaml file tutorial medium

link : <https://medium.com/buildpiper/all-you-need-to-know-about-yaml-files-8fa319b1f26f>

search command : how to load yaml file in python

link : <https://python.land/data-processing/python-yaml>

```
RETAIL_FORECASTING
  > aurora
  > explore
  > generalize_pipeline
    > __pycache__
    > config
      ! preprocess1_config.yaml
      ! preprocess2_config.yaml
    > data
    > generalize_pipeline.egg-info
    > model_output
    > outputs
    > report
    > report.txt
    > script
      < __init__.py
      < preprocess_1_unique_script...
      < preprocess_2_fill_zero.py
      < preprocess_3_add_lookback.py
      test.ipynb
      train_model.py
    > src
    > temp
    > report.txt
    < __init__.py
    < .gitignore
  README.md
```

```
generalize_pipeline > config > ! preprocess2_config.yaml
1 import yaml
2 from src.preprocessing_1.unique import UniqueProcessing
3 from src.preprocessing_2.fill_zero import GeneralizePreprocessingFillZero
4
5 def load_config(config_path):
6     with open(config_path, 'r') as file:
7         config = yaml.safe_load(file)
8     return config
9
10 def main():
11     # Main function to preprocess and transform data.
12
13     # This function reads a parquet file, renames its columns, and applies preprocessing steps.
14     # Finally, it runs a general preprocessing step to fill missing values and saves the transformed
15     # data to a new parquet file.
16
17     # Arguments
18     # None
19
20     # Returns
21     # None
22
23     # Load configuration
24     config = load_config('./config/preprocess2_config.yaml')
25
26     unique_processing = UniqueProcessing()
27     generalize_processing_fill_zero = GeneralizePreprocessingFillZero()
28
29     # Get configuration values
30     group_outlet_code = config['group_outlet_code']
31     file_path = config['file_path']
32
33     # Read the original data file
34     df = pd.read_parquet(file_path)
35
36     # Get selected columns
37     selected_cols = config['selected_cols']
38     categorical_cols = config['categorical_cols']
39     feature_cols = config['feature_cols']
40
41     # Add 'outlet_code' if group_outlet_code is True
42     if group_outlet_code:
43         required_cols.append('outlet_code')
44
45     # Read the original data file
46     df = unique_processing.read_file(file_path)
47
48     # Generalize processing fill zero
49     generalize_processing_fill_zero.run_all(df,
50                                              required_cols,
51                                              categorical_cols,
52                                              feature_cols,
```

how to load

load and use

Method2 hydra (a little different from openthaigt)

Link : <https://medium.com/@jh.baek.sd/mastering-configuration-management-in-python-with-hydra-with-omeg.conf-a-comprehensive-guide-5cf1d38e01f7>

```
REPOSITORY  
generalize_pipeline  
  ├── generalize_pipeline.egg-info  
  |   └── top_level.pth  
  ├── config  
  |   ├── preprocess1_config.yaml  
  |   └── preprocess2_config.yaml  
  ├── data  
  |   └── report.txt  
  ├── generalize_pipeline.egg-info  
  |   └── top_level.pth  
  ├── model_output  
  ├── outputs  
  └── report  
      └── report.txt  
  ├── script  
  |   ├── __init__.py  
  |   ├── preprocess_1_unique_script.py  
  |   ├── preprocess_2_fill_zero.py  
  |   ├── preprocess_3_add_lookback.py  
  |   └── test.ipynb  
  ├── train_model.py  
  ├── src  
  ├── temp  
  └── report.txt  
  ├── __init__.py  
  ├── gitignore  
  └── README.md  
  └── reference.txt  
  └── requirements.txt  
  └── setup.py  
  └── generalize_pipeline copy  
  └── italiasa  
  
  > OUTLINE  
  > TIMELINE  
  preprocess_2_fill_zero.py  
  ○ File Saved now  
  ○ Undo / Redo 1 min  
  ○ File Saved 20 mins  
  ○ File Saved 23 mins  
  ○ File Saved 30 mins  
  ○ File Saved 31 mins  
  ○ File Saved 32 mins  
  ○ File Saved 33 mins  
  ○ File Saved 37 mins  
  ○ File Saved 46 mins  
  ○ File Saved 2 days  
  1 min  
  20 mins  
  23 mins  
  30 mins  
  31 mins  
  32 mins  
  33 mins  
  37 mins  
  46 mins  
  2 days  
  
generализированное import preprocessor.fill_zero import GeneralizePreprocessingFillZero  
1 import via  
2 import hydra  
3 from omegaconf import DictConfig  
4 from src.preprocess 1 unique import UniqueProcessing  
5 from src.preprocess 2 generalize_preprocess_fill_zero import GeneralizePreprocessingFillZero  
6  
7  
@hydra.main(config_path = '../config', config_name='preprocess2_config', version_base=None)  
8  
def main(cfg: DictConfig) -> None:  
    """  
        Main function to preprocess and transform data.  
      
    This function reads a parquet file, renames its columns, and applies preprocessing steps.  
    Finally, it runs a general preprocessing step to fill missing values and saves the transformed data.  
    """  
    Arguments  
    -----  
    None  
    Returns  
    -----  
    None  
    """  
    unique_processing = UniqueProcessing()  
    generalize_processing_fill_zero = GeneralizePreprocessingFillZero()  
    # Get configuration values  
    # group.outlet_code = config['group_outlet_code']  
    group_outlet_code = cfg.group_outlet_code  
    file_path = cfg.file_path  
    save_path = cfg.save_path  
    selected_cols = cfg.selected_cols  
    categorical_cols = cfg.categorical_cols  
    feature_cols = cfg.feature_cols  
    cols_to_be_grouped_by_first = cfg.cols_to_be_grouped_by_first  
    cols_to_be_grouped_by_sum = cfg.cols_to_be_grouped_by_sum  
    # Add 'outlet_code' if group_outlet_code is True  
    required_cols = ['SKU_ID', 'year', 'month', 'day', 'target']  
    if group_outlet_code:  
        required_cols.append("outlet_code")  
        selected_cols.append("outlet_code")  
        categorical_cols.append("outlet_code")  
    group_level = ['SKU_ID']  
    if group_outlet_code:  
        group_level.append("outlet_code")  
    # Read the original data file  
    df = unique_processing.read_file(file_path)  
    generalize_processing_fill_zero.run_all(  
        df,  
        selected_cols,  
        categorical_cols,  
        feature_cols,  
        cols_to_be_grouped_by_first
```

*** So, generally, path should always be used relate to where we run the scripts