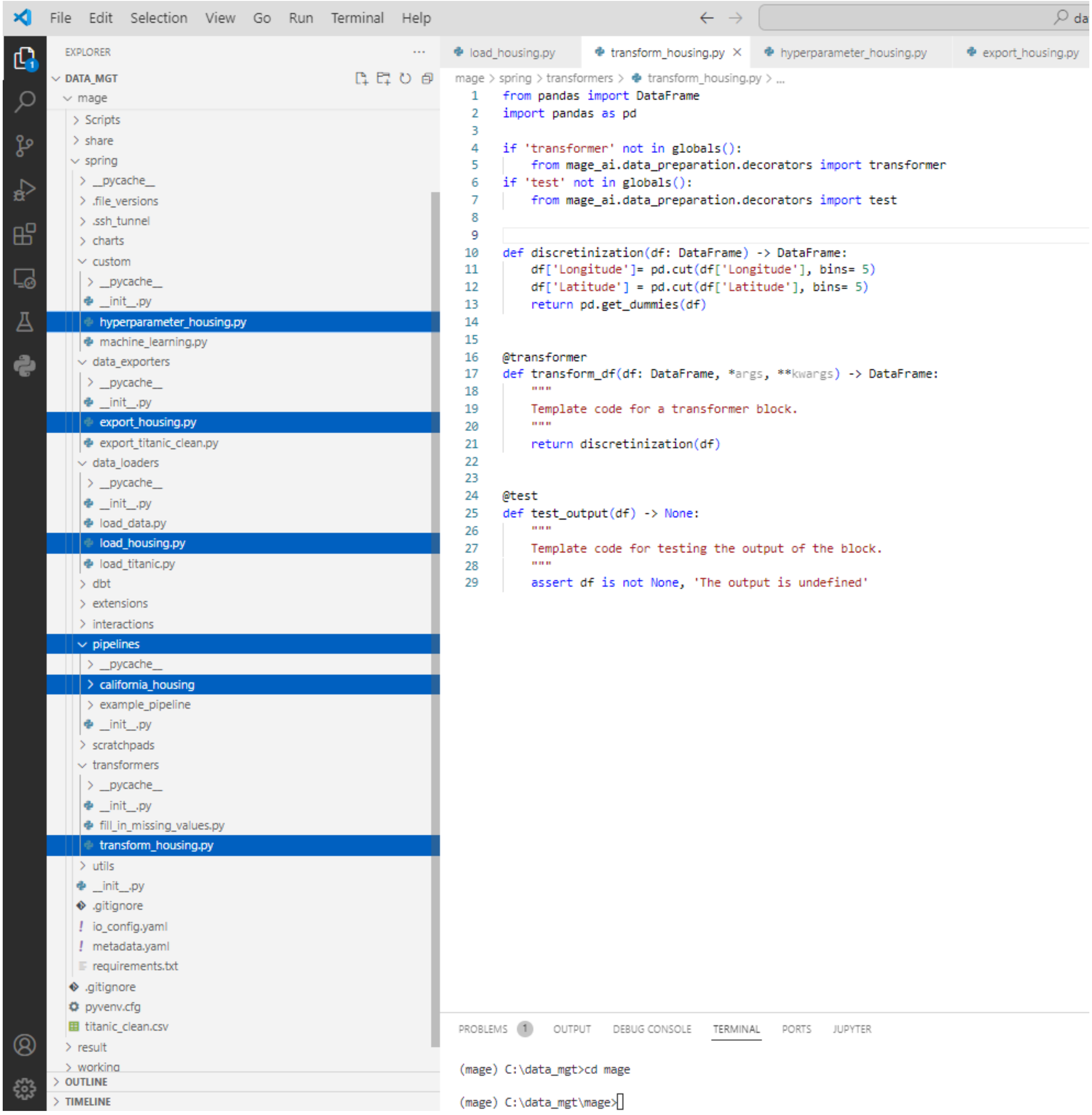
4. hyperparameter_for_mage

Hyperparameter tuning getting_started & model_persistence

4-1. Vs Code



```python
from pandas import DataFrame
import pandas as pd

if 'transformer' not in globals():
    from mage_ai.data_preparation.decorators import transformer
if 'test' not in globals():
    from mage_ai.data_preparation.decorators import test


def discretinization(df: DataFrame) -> DataFrame:
    df['Longitude']= pd.cut(df['Longitude'], bins= 5)
    df['Latitude'] = pd.cut(df['Latitude'], bins= 5)
    return pd.get_dummies(df)


@transformer
def transform_df(df: DataFrame, *args, **kwargs) -> DataFrame:
    """
    Template code for a transformer block.
    """
    return discretinization(df)


@test
def test_output(df) -> None:
    """
    Template code for testing the output of the block.
    """
    assert df is not None, 'The output is undefined'
```

```
(mage) C:\data_mgt>cd mage

(mage) C:\data_mgt\mage>
```

위 그림의 spring 폴더아래 하위 폴더인 4개 블록에 해당되는 폴더아래에 각각 4개의 .py 파일 생성/저장

Dataloader block

```python
from sklearn.datasets import fetch_california_housing
from pandas import DataFrame
import pandas as pd


if 'data_loader' not in globals():
    from mage_ai.data_preparation.decorators import data_loader
if 'test' not in globals():
    from mage_ai.data_preparation.decorators import test

print(fetch_california_housing().keys())

@data_loader
def load_data_from_api(**kwargs) -> DataFrame:
    X = pd.DataFrame(fetch_california_housing().data,
                 columns=fetch_california_housing().feature_names)
    y = pd.DataFrame(fetch_california_housing().target,
                 columns=fetch_california_housing().target_names)
    return pd.merge(X, y, left_index=True, right_index=True)

@test
def test_output(df) -> None:
    """
    Template code for testing the output of the block.
    """
    assert df is not None, 'The output is undefined'
```

Datatransformer block

```
from pandas import DataFrame
import pandas as pd

if 'transformer' not in globals():
    from mage_ai.data_preparation.decorators import transformer
if 'test' not in globals():
    from mage_ai.data_preparation.decorators import test

def discretinization(df: DataFrame) -> DataFrame:
    df['Longitude']= pd.cut(df['Longitude'], bins= 5)
    df['Latitude'] = pd.cut(df['Latitude'], bins= 5)
    return pd.get_dummies(df)

@transformer
def transform_df(df: DataFrame, *args, **kwargs) -> DataFrame:
    """
    Template code for a transformer block.
    """
    return discretinization(df)


@test
def test_output(df) -> None:
    """
    Template code for testing the output of the block.
    """
    assert df is not None, 'The output is undefined'
```

@custom

```
import pandas as pd
import os
import pickle
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import train_test_split
from scipy.stats import randint

if "custom" not in globals():
    from mage_ai.data_preparation.decorators import custom
if "test" not in globals():
    from mage_ai.data_preparation.decorators import test

def _model_save(rf_model):
    """
    Save RandomForest model to a file.
    """
    # Here you would implement the logic to save your trained model to a file
    # Example:
    os.makedirs('../result', exist_ok= True)
    with open('../result/random_forest_model.pkl', 'wb') as file:
        pickle.dump(rf_model, file)
    pass

@custom
def random_forest_train(df: pd.DataFrame, *args, **kwargs):
    """
    Train a Random Forest Classifier and predict the 'Survived' column.

    Args:
        df: Data frame containing the training data.

    Returns:
        Data frame with a new column 'Survived_predict' with predictions.
    """
    # Prepare the data
    X = df.drop(['MedHouseVal'], axis=1)
    y = df['MedHouseVal']
    X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

    # define the parameter space that will be searched over
    param_distributions = {'n_estimators': randint(1, 5),
                           'max_depth': randint(5, 10)}

    # now create a searchCV object and fit it to the data
    search = 
RandomizedSearchCV(estimator=RandomForestRegressor(random_state=0),
                               n_iter=10,
                               param_distributions=param_distributions,
                               random_state=0)

    # Initialize the Random Forest Classifier
    rf_model = RandomForestRegressor(random_state=0),

    # Train the model, now create a searchCV object and fit it to the data
    search = 
RandomizedSearchCV(estimator=RandomForestRegressor(random_state=0),
                               n_iter=10,
                               param_distributions=param_distributions,
                               random_state=0)

    search.fit(X_train, y_train)
    tf = pd.DataFrame(search.cv_results_)[['param_max_depth', 
'param_n_estimators', 'params', 'mean_test_score', 'rank_test_score']]
    print(tf.sort_values('rank_test_score'))
    print(f"The Best model's parameters is {search.best_params_}")
    print(f'The Best accuracy score of model is {search.score(X_test, 
y_test)}')

    # Predict using the trained model
    y_pred = search.predict(X_test)

    # Optionally save the model
    _model_save(search)

    # Assign predictions to a new column in the dataframe
    tf = pd.merge(X_test, y_test, left_index=True, right_index=True)
    tf['Hosing_pred'] = y_pred
    return tf

@test
def test_output(output, *args) -> None:
    """
    Template code for testing the output of the block.

    Args:
        output: The output from the random_forest_train function.
    """
    assert output is not None, "The output is undefined"
    assert 'Survived_predict' in output.columns, "Prediction column is missing 
in the output dataframe"
    # You can add more tests to check the quality of your predictions,
    # such as accuracy score, confusion matrix, etc.
```

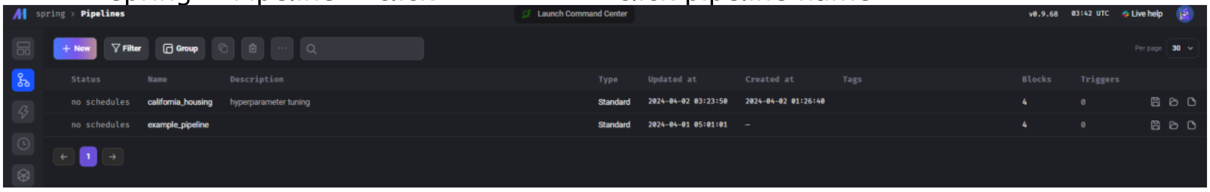
@data_exporter

```python
from mage_ai.io.file import FileIO
from pandas import DataFrame

if 'data_exporter' not in globals():
    from mage_ai.data_preparation.decorators import data_exporter

@data_exporter
def export_data_to_file(df: DataFrame, **kwargs) -> None:

    filepath = '../result/housing_predict.csv'
    FileIO().export(df, filepath)
```

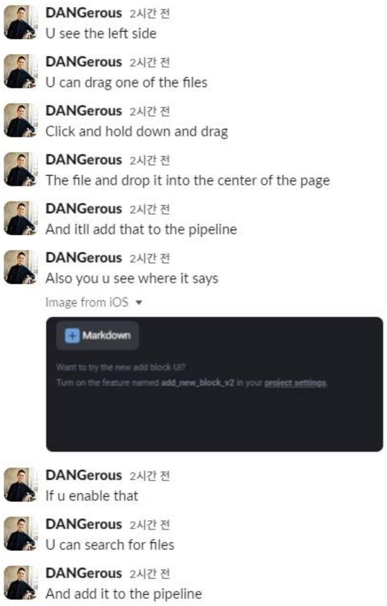
각 블록을 연결하여 데이터 파이프라인을 만들기

4-2. Mage.ai WEB UI

- (mage) C:\data_mgt\mage>mage start spring
- spring > Pipeline > click + New > click pipeline name



Status	Name	Description	Type	Updated at	Created at	Tags	Blocks	Triggers
no schedules	california_housing	hyperparameter tuning	Standard	2024-04-02 03:23:50	2024-04-02 01:26:40		4	0
no schedules	example_pipeline		Standard	2024-04-01 05:01:01	–		4	0



DANGerous 2시간 전
U see the left side

DANGerous 2시간 전
U can drag one of the files

DANGerous 2시간 전
Click and hold down and drag

DANGerous 2시간 전
The file and drop it into the center of the page

DANGerous 2시간 전
And itll add that to the pipeline

DANGerous 2시간 전
Also you u see where it says

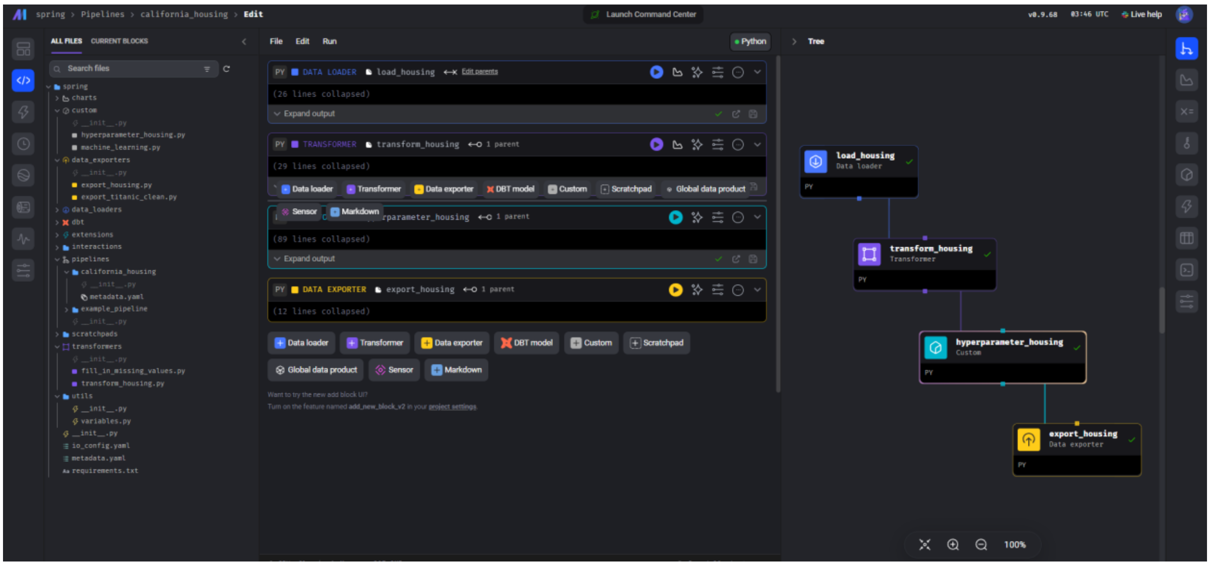
Image from iOS ▾

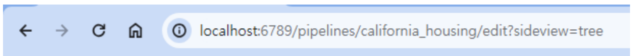
Want to try the new add block UI?
Turn on the feature named **add_new_block_v2** in your project settings.

DANGerous 2시간 전
If u enable that

DANGerous 2시간 전
U can search for files

DANGerous 2시간 전
And add it to the pipeline



- py파일을 편집하고 싶으면 VS code에서 편집하고 저장(ctrl + s),  localhost:6789/pipelines/california_housing/edit?sideview=tree 화면 reset