ENERGY EFFICIENCY ESTIMATION

Low Level Design

Table of Contents

1. INTRODUCTION

1.1 Why This Low-Level Document

The goal of LLD or a low-level design document (LLD) is to give the internal logical design of the actual program code for Food Recommendation System. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

1.2 Scope

Low-level design (LLD) is a component-level design process that follows a step-by-Step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

2. ARCHITECTURE

2.1 Model Flow

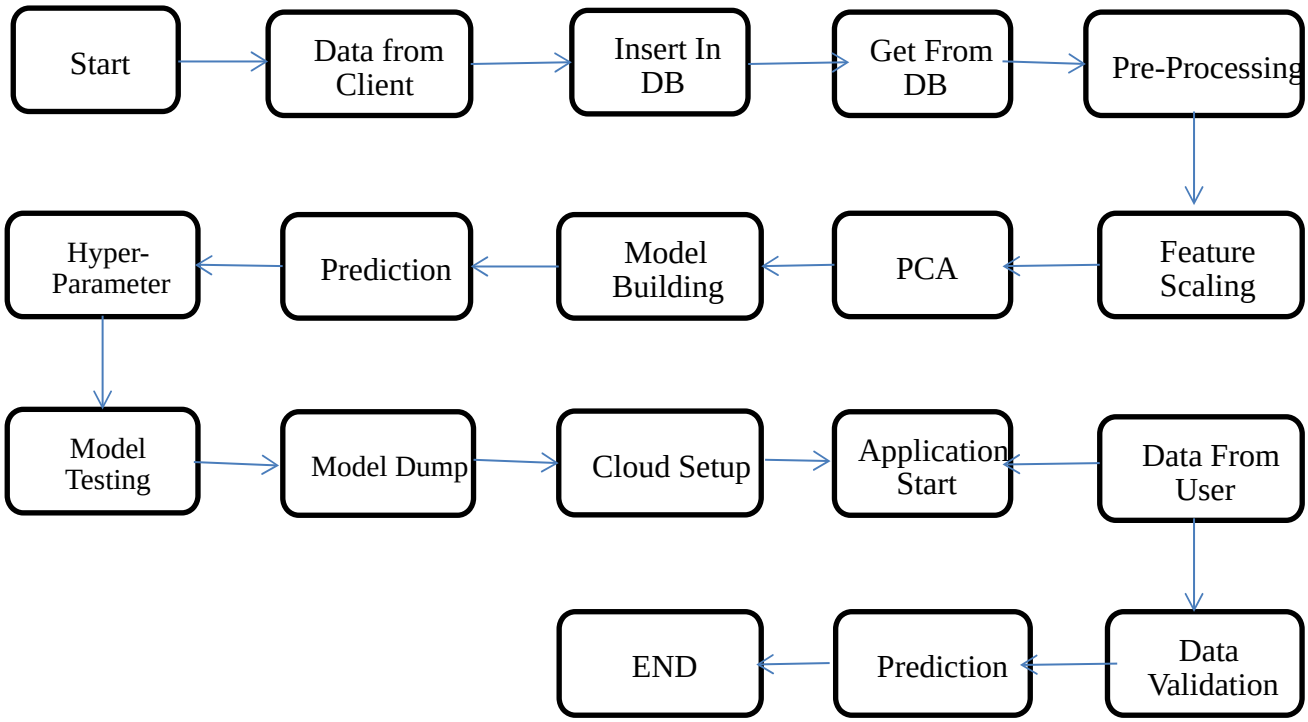

Figure No. 2 The Total Model Process

3. ARCHITECTURE DESCRIPTION

3.1 Data Description

The data can collect form the client in any file format. They perform energy analysis using 12 different building shapes simulated in Ecotect. The buildings differ with respect to the glazing area, the glazing area distribution, and the orientation, amongst other parameters. We simulate various settings as functions of the afore-mentioned characteristics to obtain 768 building shapes. The dataset comprises 768 samples and 8 features, aiming to predict two real valued responses. It can also be used as a multi-class classification problem if the response is rounded to the nearest integer.

The dataset contains eight attributes (or features, denoted by X1...X8) and two responses (or outcomes, denoted by y1 and y2). The aim is to use the eight features to predict each of the two responses.

Indication	Names
X1	Relative Compactness
X2	Surface Area
X3	Wall Area
X4	Roof Area
X5	Overall Height
X6	Orientation
X7	Glazing Area
X8	Glazing Area Distribution
Y1	Heating Load
Y2	Cooling Load

3.2 Data Insertion into Database

- **Database Creation & Connection** - Create a database with name energyefficiency having keyspace name energy and try to create the connection.

- **Create Table** – Check the table inside the keyspace having name with it energy_data

- **Insert File** – Insert the data that given by client into the database with help of python script file.
- **Check Data** – Then check that the excel file is uploaded in the dataset or not with the command SELECT * FROM energy.energy_data.

3.3 Export Data from Database

Data Export from Database - The data that we uploaded in database, now we need to pull out from the database for model building.

3.4 Data Pre-processing

In data pre-processing, we don't do anything special. The reason behind that there is no null value or categorical column in the dataset.

3.5 Feature Scaling

The Feature Scaling technique can be used to scale down the entire feature column in one scale, so that there is not much dispersion in dataset. In our dataset we had used the Min Max scalar technique which brings the entire feature column in 0 to 1 range.

3.6 PCA

PCA is an unsupervised statistical technique that is used to reduce the dimensions of the dataset. ML models with many input variables or higher dimensionality tend to fail when operating on a higher input dataset. PCA helps in identifying relationships among different variables & then coupling them.

3.7 Model Building

After the unsupervised technique we divide the data into train & test. The train & test data passed to the model that we are using in project i.e. Linear Regression, Ridge Regression, Lasso Regression, Elastic Net and Random Forest Regressor. Based on the score we select the model for deployment purpose. Before that we need to tune the parameter of each and every module.

3.8 Hyper Parameter Tuning

In hyper parameter tuning we have implemented ensemble techniques like random forest regressor & regularization technique such as ridge, lasso, elastic net etc. We also done grid search cv and from that we implemented cross validation techniques for different model. From that we have

chosen best parameters according to hyper parameter tuning and best score from their accuracies so we got 0.99 score for heating load and 0.96 score for cooling load for the random forest regressor model.

3.9 Model Testing

After hyper parameter tuning we put all the best parameter in our all ML model. From that we test our data & the score of the model from that we concluded the data score has been increase.

3.10 Model Dump

After comparing all accuracies and checked the score we have chosen hyper parameterized random forest regression as our best model by their results so we have dumped these model in a pickle File format with the help of pickle python module.

3.11 Cloud Setup

After model building we want to deploy the model to server. In deployment we can use different services such as Amazon Web Service (AWS), Azure Service, and Google Cloud Service (GCP). In that we deploy our model in AWS, for that 1st we have to create the environment and then after we have to create the code pipeline. After that we have to connect our pipeline with the environment.

3.12 Data from User

Here we can collect the data from the user. In which we can collect the different type of data such as Relative Compactness, Surface Area, Wall Area, Roof Area, Overall Height, Orientation, Glazing Area, and Glazing Area Distribution.

3.13 Data Validation

In data validation the data from user we need to scale it down in between 0 to 1 & then we have to apply PCA so that our model can understand the data and gives a proper output.

3.14 Prediction

After entering the data and data validation when user hit the submit button we get the heating load and cooling load at the screen.

4 . Technology Stack

Front End	HTML/CSS
Back End	Python/Flask
Database	Cassandra
Deployment	Heroku
Model	1. Linear Regression 2. Random Forest
IDE	VS Code

5. UNIT TEST CASE

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	1. Application URL is accessible 2. Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether user is able to see input fields	1. Application is accessible 2. User is logged in to the application	User should be able to see input fields
Verify whether user is able to edit all input fields	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be able to edit all input fields
Verify whether user gets submit button to submit the inputs	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be presented with recommended results on clicking submit
Verify whether the recommended results are in accordance to the selections user made	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	The recommended results should be in accordance to the selections user made

6. MODULAR CODE STRUCTURE

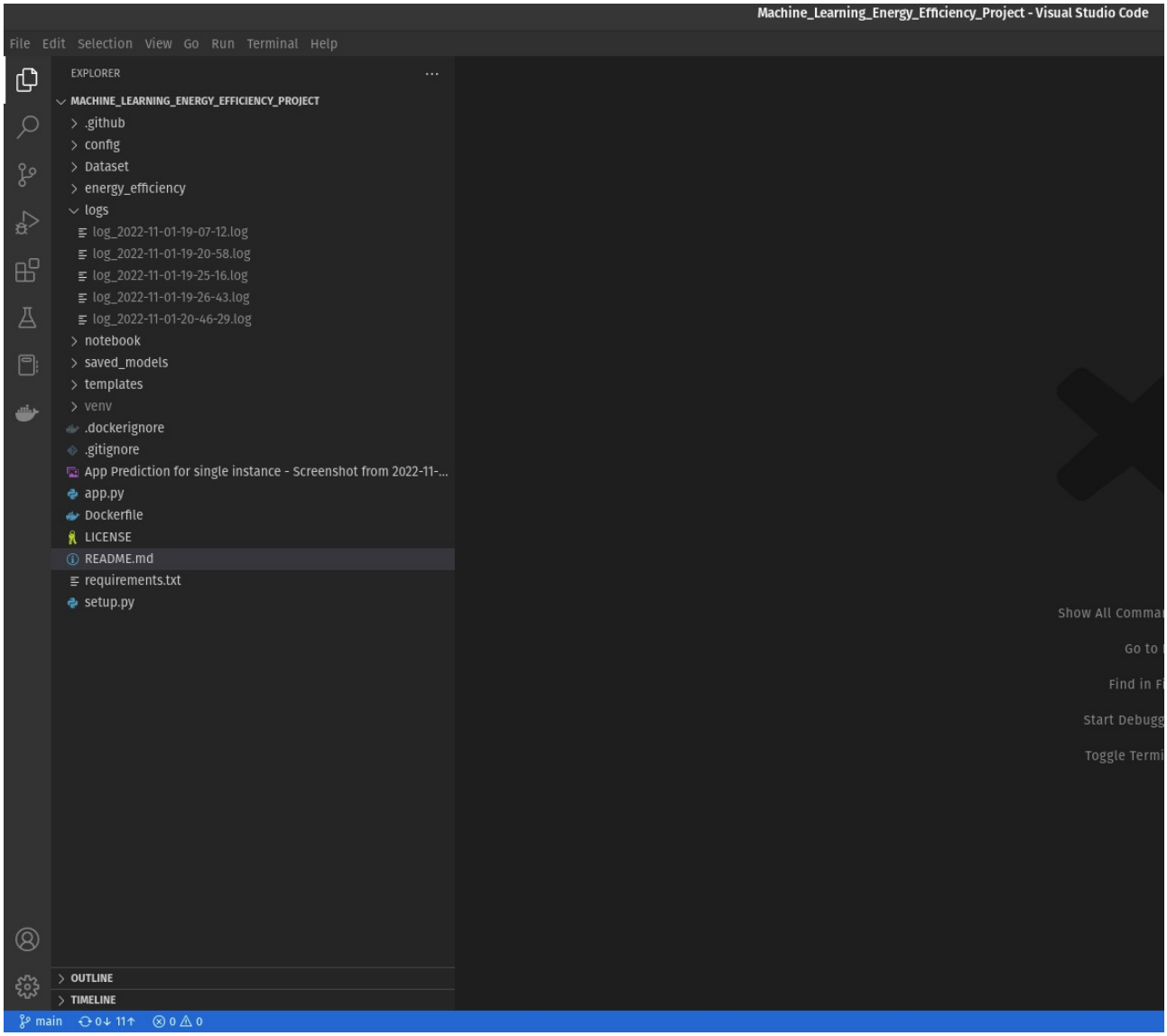

Fig. 1 Project folder structure

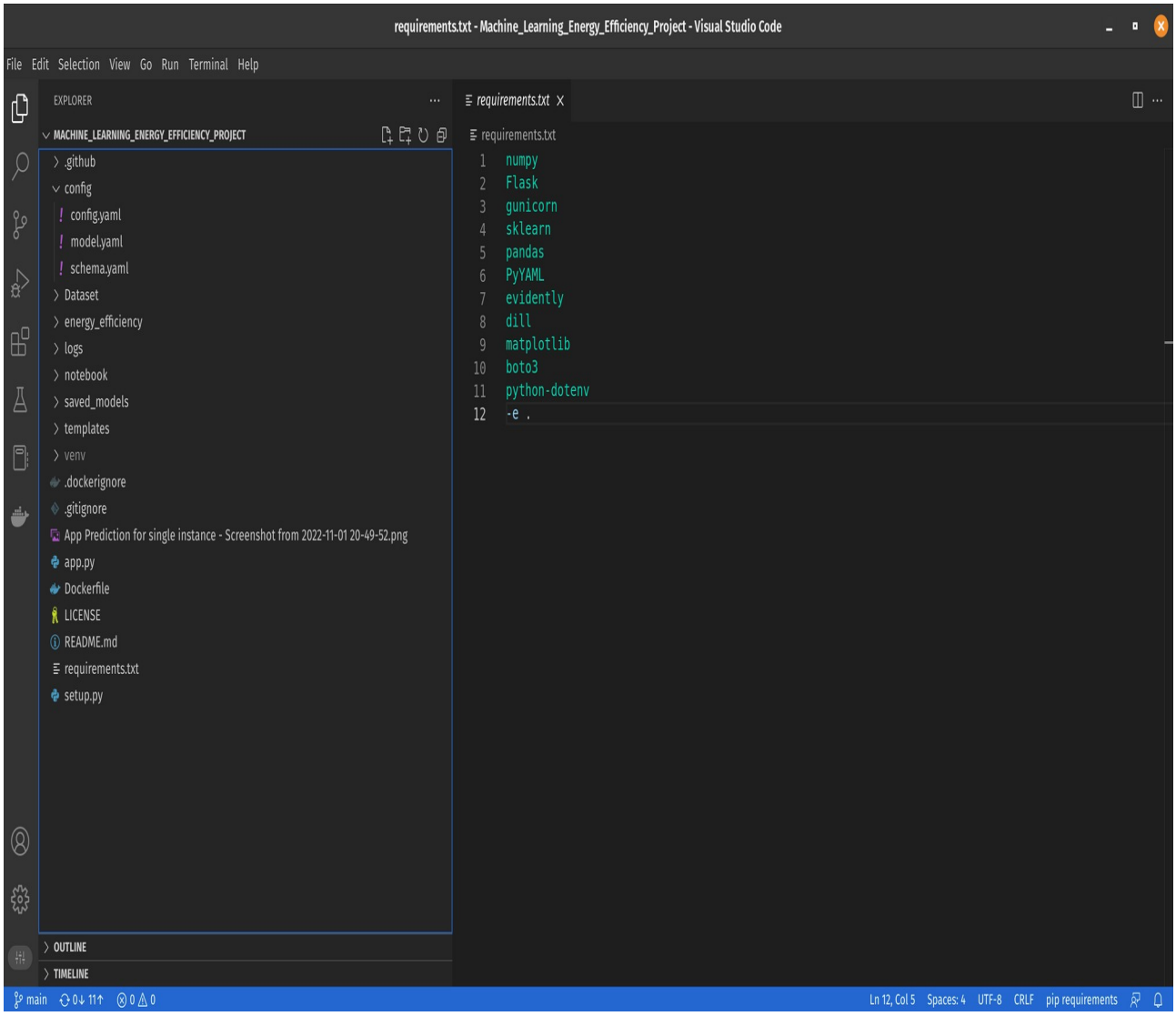

Fig. 2.Config_folder_structure with requiremnts-txt

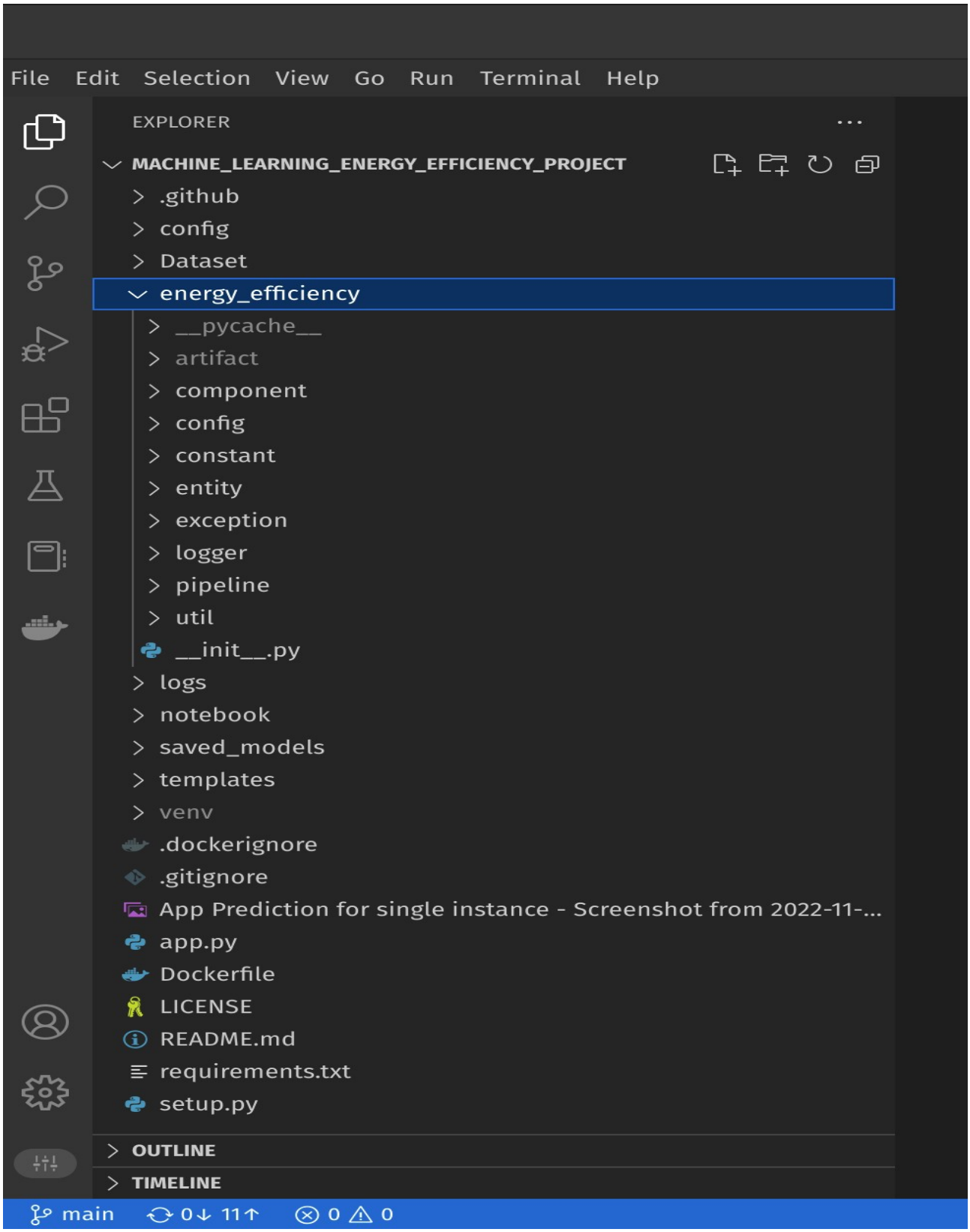

Fig. 3. energy_efficiency package structure

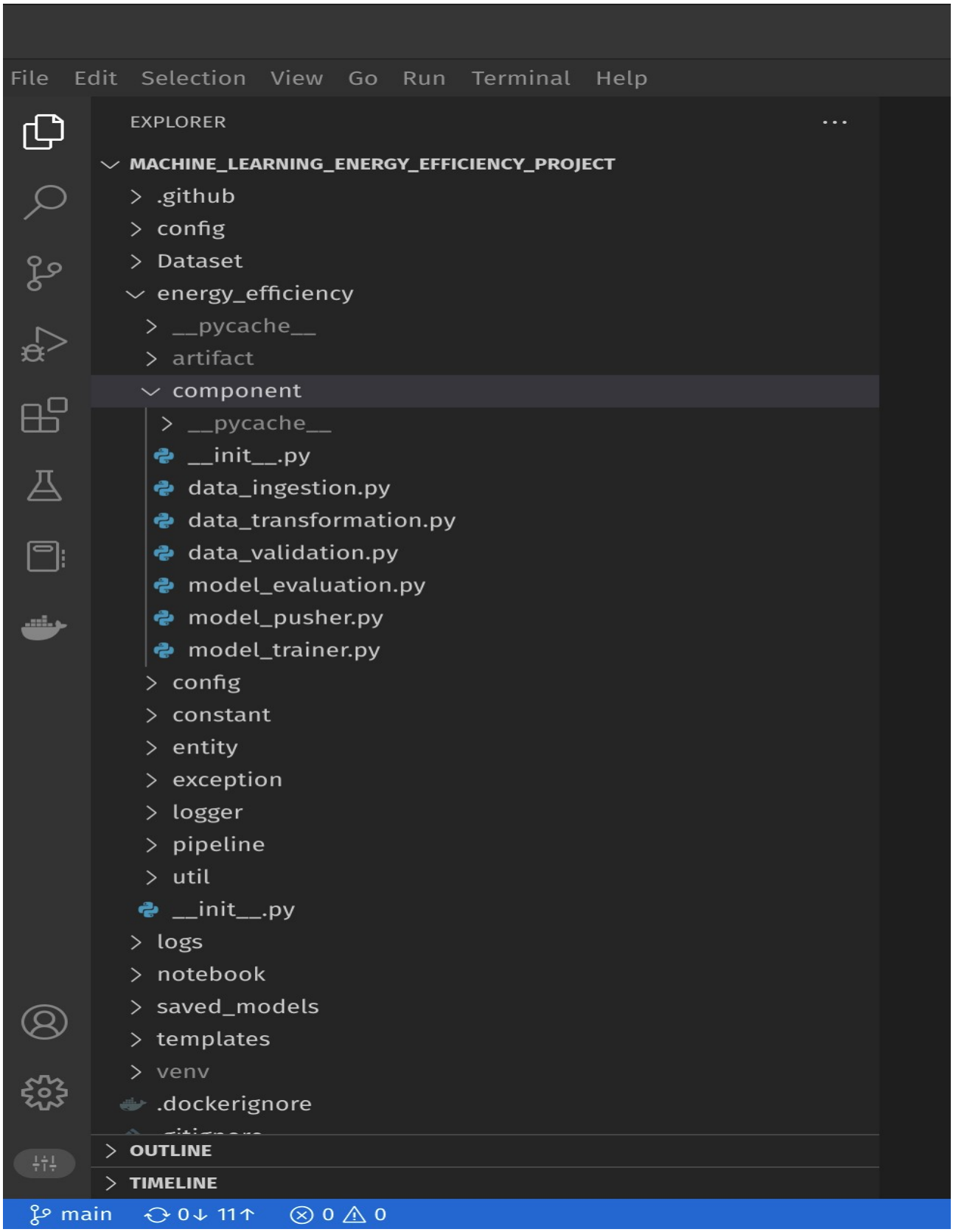

Fig. 4. component_module_structure

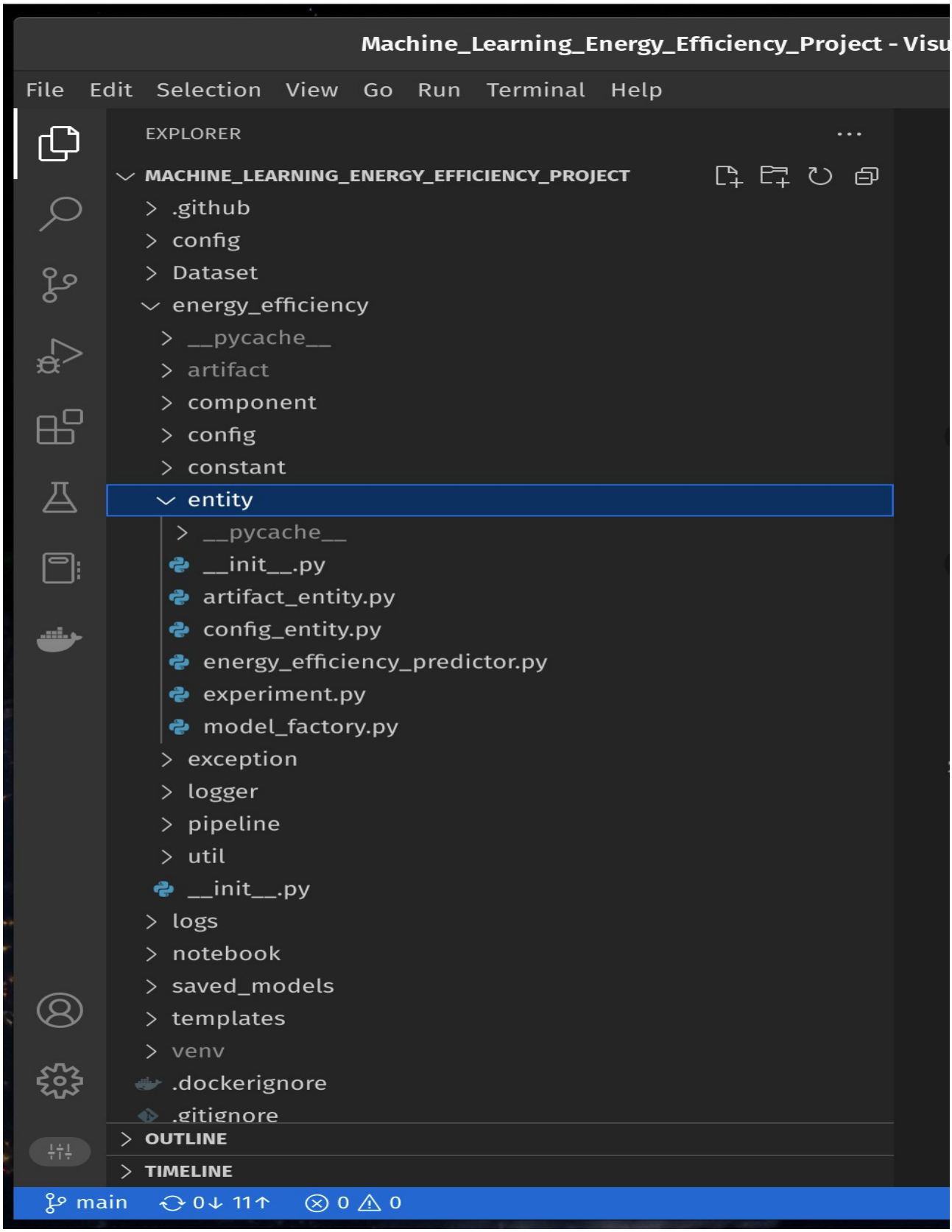

Fig. 5. entity_module-structure

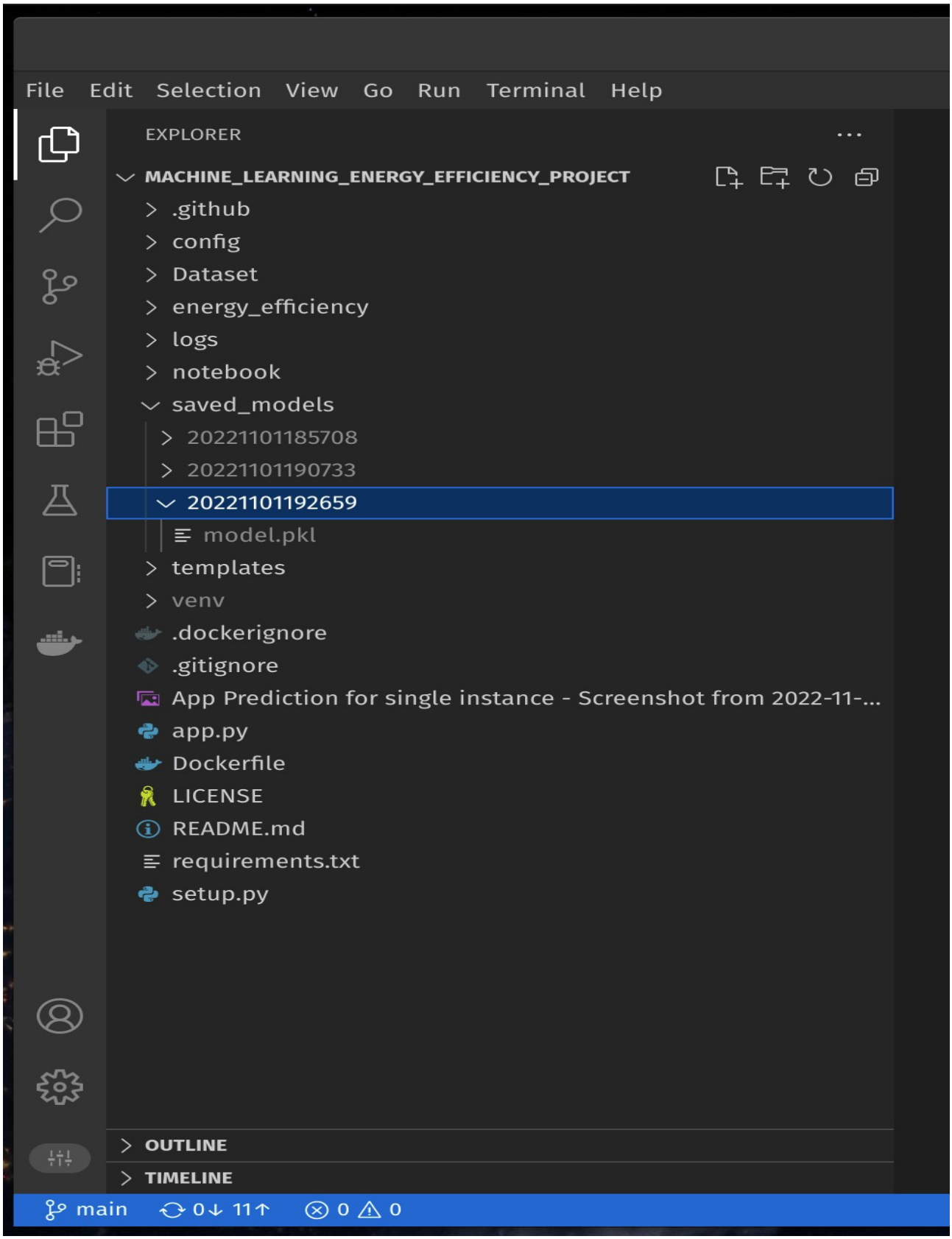

Fig. 6. saved_models with DateTimeStamp

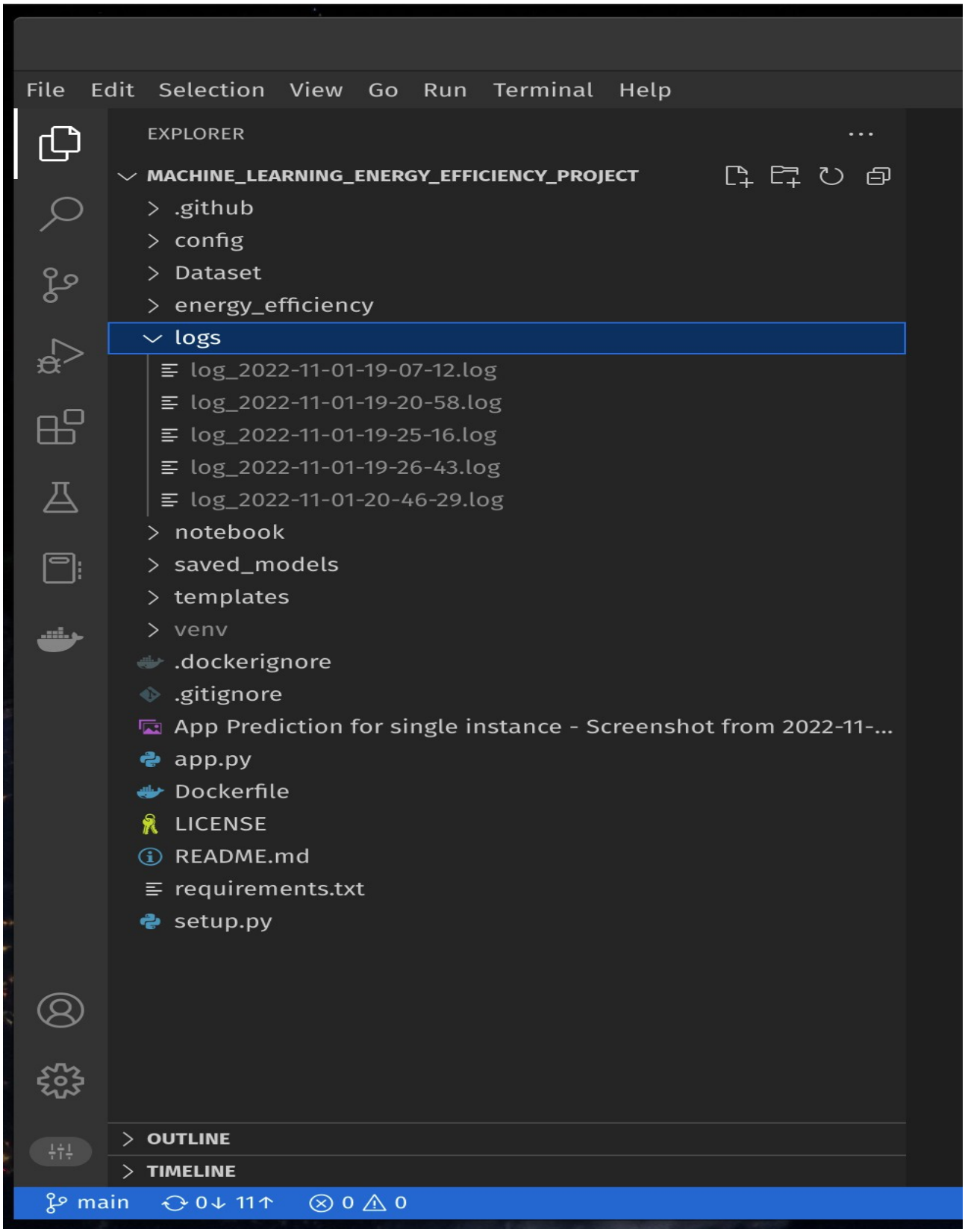

Fig. 7. logs_generated_for_reference