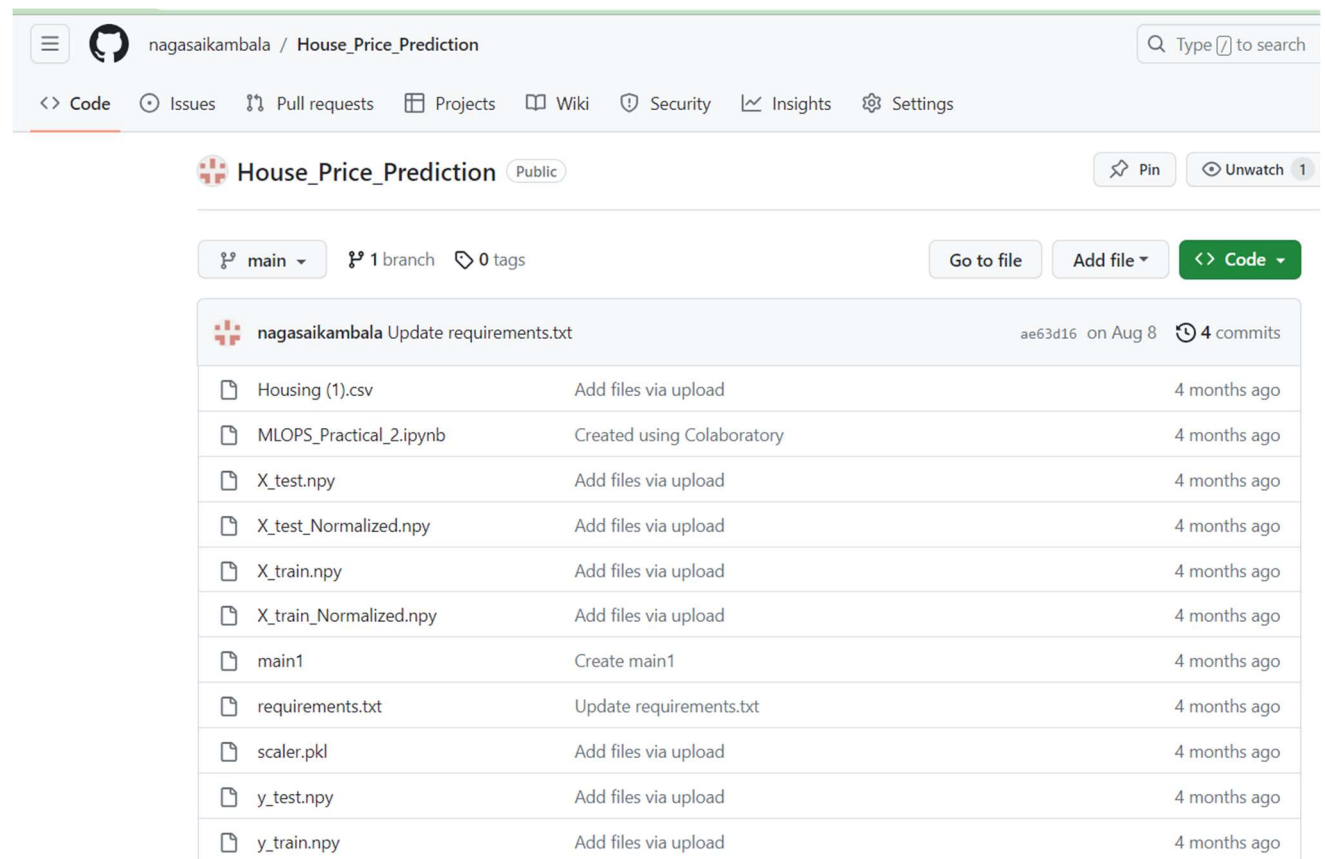


Practical-3

Generation of Reproducible and Interactive ML Project.

Task 1: Create the Github repository for the house rate prediction project created in practical 2.



The screenshot shows the GitHub interface for the repository 'House_Price_Prediction' by user 'nagasaikambala'. The repository is public and has 1 branch (main) and 0 tags. The file list includes:

File Name	Commit Message	Commit Date
Housing (1).csv	Add files via upload	4 months ago
MLOPS_Practical_2.ipynb	Created using Colaboratory	4 months ago
X_test.npy	Add files via upload	4 months ago
X_test_Normalized.npy	Add files via upload	4 months ago
X_train.npy	Add files via upload	4 months ago
X_train_Normalized.npy	Add files via upload	4 months ago
main1	Create main1	4 months ago
requirements.txt	Update requirements.txt	4 months ago
scaler.pkl	Add files via upload	4 months ago
y_test.npy	Add files via upload	4 months ago
y_train.npy	Add files via upload	4 months ago

Task 2: Integrate your repository with the binder to make your project interactive. (Hint: refer to the following link for the steps: (<https://mybinder.org/>))

Build and launch a repository

GitHub repository name or URL

GitHub ▾

Git ref (branch, tag, or commit)

Path to a notebook file (optional) File ▾

[launch](#)

Copy the URL below and share your Binder with others:

Expand to see the text below, paste it into your README to show a binder badge: [launch](#) [binder](#)

Waiting

Building

Build logs [view raw](#) [hide](#)

```
---> d9621d44ac85
Step 5/50 : ENV LC_ALL=en_US.UTF-8      LANG=en_US.UTF-8      LANGUAGE=en_US.UTF-8
---> Using cache
---> 4bed1b4925d0
Step 6/50 : ENV SHELL=/bin/bash
---> Using cache
---> ef88f77a3ff4
Step 7/50 : ARG NB_USER
---> Using cache
---> ec514bbd6350
Step 8/50 : ARG NB_UID
---> Using cache
---> b9fb967f54fa
```

The screenshot displays a JupyterLab environment. On the left, a file browser shows a directory structure with files like 'templates', 'app.py', 'model.pkl', 'requirements...', and 'scaler.pkl'. The main area shows the code for 'app.py', which is a Flask application. The code imports Flask, numpy, and pickle, defines a ValuePredictor function, and sets up a Flask app with routes for '/result' and '/'. A notification dialog in the bottom right corner asks if the user wants to receive official Jupyter news.

```
1 from flask import Flask, render_template, jsonify, request
2 import numpy as np
3 import pickle
4 app = Flask(__name__)
5 def ValuePredictor(to_predict_list):
6     X_test = np.array(to_predict_list).reshape(1, 1)
7     #Load the instance of StandardScaler object
8     scaler = pickle.load(open("scaler.pkl", "rb"))
9     #Normalize the data
10    X_test_Normalized = scaler.transform(X_test)
11    loaded_model = pickle.load(open("model.pkl", "rb"))
12    result = loaded_model.predict(X_test_Normalized)
13    return result[0]
14 @app.route('/result', methods = ['POST'])
15 def result():
16     if request.method == 'POST':
17         to_predict_list = request.form.to_dict()
18         to_predict_list = list(to_predict_list.values())
19         to_predict_list = list(map(int, to_predict_list))
20         prediction = ValuePredictor(to_predict_list)
21         return render_template("result.html", prediction = prediction)
22 @app.route("/")
23 def hello_world():
24     return render_template("home.html")
```

Would you like to receive official Jupyter news?
Please read the privacy policy.

[Open privacy policy](#) Yes No