Week5: ML prediction model Deployment on Heroku

Name: Madoka Fujii

Batch Code: LISUM23: 30

Submission Date: Aug 4, 2023

Submitted to: Data Glacier

Step1: Select any toy data (simple data).

I found a toy data from Kaggle.

https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished

Originally, there are 13 variables.

```
# Separate the dependent variable and indepedent variables
Y = df['price']

X = df.drop(columns = {'price', 'mainroad', 'guestroom', 'airconditioning', 'prefarea', 'furnishingstatus'})
```

```
X = pd.get_dummies(X, drop_first = True)
```

```
X.head()
```

	area	bedrooms	bathrooms	stories	parking	basement_yes	hotwaterheating_yes
0	7420	4	2	3	2	0	0
1	8960	4	4	4	3	0	0
2	9960	3	2	2	2	1	0
3	7500	4	2	2	3	1	0
4	7420	4	1	2	2	1	0

But I dropped price and several variables from X. This is because price is dependent variable. And other several variables are dummy variables with Yes or No. In order to fit the flask deploying page, they were dropped. (That is not good idea to have so many variables in web to people fill out.)

I kept 7 independent variables to predict housing price.

Step2: Save the model

```
import pickle
```

```
# Saving model to disk
pickle.dump(gradient_reg, open('model.pkl','wb'))
```

```
# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[8000, 3, 2, 3,2,1,1]]))
```

```
[8357433.19312691]
```

Using pickle, saved the model to deploy on Flask.

Step3. Preparation of file to deploy

In PyCharm, created app2.py file.

Categorical items are Yes or No choice. Set the code if Yes=1 else 0 so that program works.

app2.py | index2.html

```
import numpy as np
from flask import Flask, request, render_template
import pickle

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index2.html')

@app.route('/predict', methods=['POST'])
def predict():

    flag = False
    if request.method == "POST":
        area = int(request.form.get('area'))
        bedrooms = int(request.form.get('bedrooms'))
        bathrooms = int(request.form.get('bathrooms'))
        stories = float(request.form.get('stories'))
        parking = float(request.form.get('parking'))

        basement_input = request.form.get('basement_yes')
        basement_yes = 1 if basement_input == "Yes" else 0
```

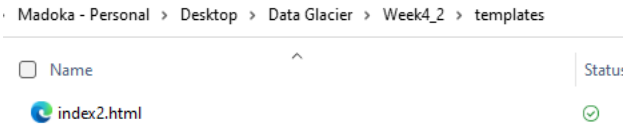
```
        hotwaterheating_input = request.form.get('hotwaterheating_yes')
        hotwaterheating_yes = 1 if basement_input == "Yes" else 0

        final_features = [area, bedrooms, bathrooms, stories, parking, basement_yes, hotwaterheating_yes]
        features = [np.array(final_features)]

    prediction = model.predict(features)

    output = round(prediction[0], 2)

    return render_template('index2.html', flag=True, prediction_text=f'Housing price should be ${output}.')

if __name__ == "__main__":
    app.run(debug=True)
```

Created index2.html file.

app2.py | index2.html

```
        <label for="bedrooms">Bedrooms:</label>
        <input type="text" name="bedrooms" id="bedrooms" placeholder="eg:5" required="required" />

        <label for="bathrooms">Bathrooms:</label>
        <input type="text" name="bathrooms" id="bathrooms" placeholder="eg:3" required="required" />

        <label for="stories">Stories:</label>
        <input type="text" name="stories" id="stories" placeholder="eg:2" required="required" />

        <label for="parking">Parking:</label>
        <input type="text" name="parking" id="parking" placeholder="eg:1" required="required" />

        <label for="basement_yes">Basement:</label>
        <select name="basement_yes" id="basement_yes">
            <option value="Yes">Yes</option>
            <option value="No">No</option>
        </select>

        <label for="hotwaterheating_yes">HotwaterHeating:</label>
        <select name="hotwaterheating_yes" id="hotwaterheating_yes">
            <option value="Yes">Yes</option>
            <option value="No">No</option>
        </select>
```

Under the templates folder, saved the index2.html file.



Added requirement.txt

```
Flask == 1.1.2
gunicorn == 20.1.0
itsdangerous==2.0.1
Jinja2==2.11.3
MarkupSafe==2.0.1
Werkzeug==2.0.3
numpy>=1.9.2
scipy>=0.15.1
scikit-learn>=0.18
matplotlib>=1.4.3
pandas>=0.19
```

Added Procfile.

Procfile

```
web: gunicorn app2:app
```

Added MIT License.

Code Blame 21 lines (17 loc) · 1.04 KB

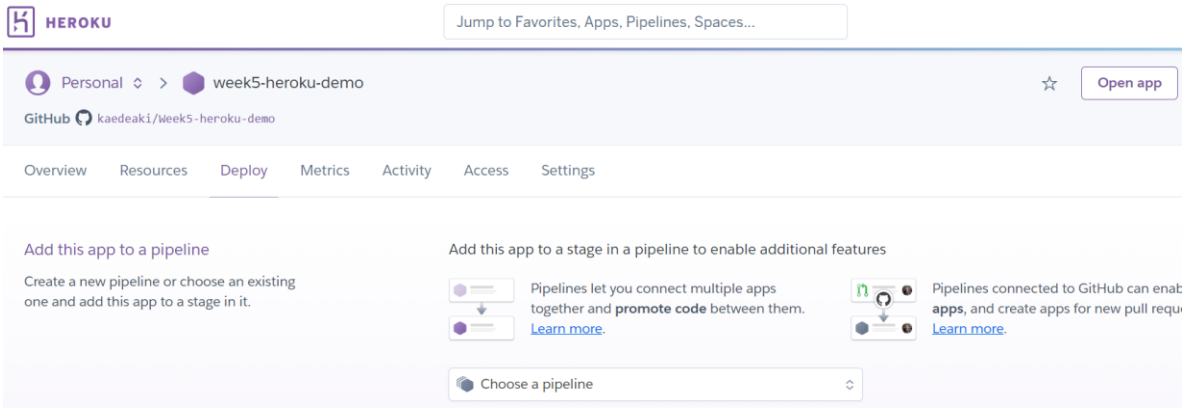
```
MIT License

Copyright (c) 2023 Madoka Fujii

Permission is hereby granted, free of charge, to any person obtaining a c
of this software and associated documentation files (the "Software"), to
in the Software without restriction, including without limitation the rig
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included i
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
```

Step4.Upload to GitHub all of the files and folders.

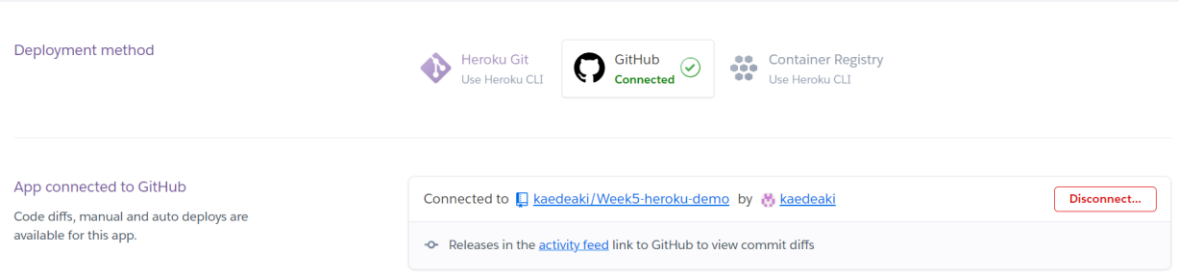
static/CSS	Add files via upload
templates	Add files via upload
LICENSE	Initial commit
Procfile	Add files via upload
README.md	Initial commit
Week4_housing_price.ipynb	Add files via upload
app2.py	Add files via upload
housing.csv	Add files via upload
model.pkl	Add files via upload
requirements.txt	Add files via upload

README.md

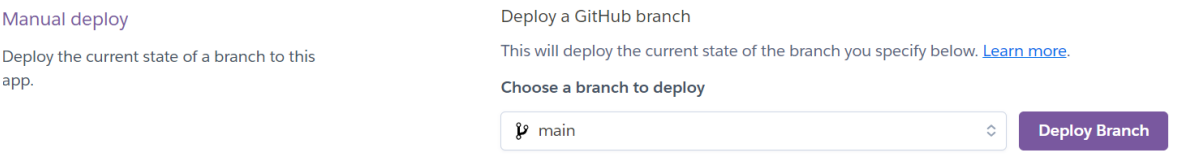
Step5.Create Heroku account and a new app.



Step6.Connect with a branch in GitHub



Step7.Manual deploy



Step8.Install all requirements and build completion

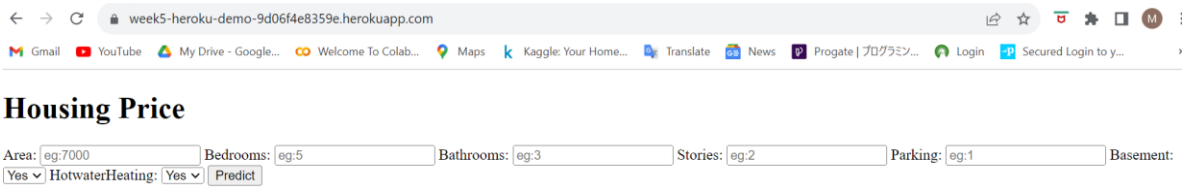
```
       To use a different version, see: https://devcenter.heroku.com/articles/python-runtimes
-----> No change in requirements detected, installing from cache
-----> Using cached install of python-3.11.4
-----> Installing pip 23.2.1, setuptools 68.0.0 and wheel 0.41.0
-----> Installing SQLite3
-----> Installing requirements with pip
-----> Discovering process types
       Procfile declares types -> web
-----> Compressing...
       Done: 149.8M
-----> Launching...
       Released v7
       https://week5-heroku-demo-9d06f4e8359e.herokuapp.com/ deployed to Heroku
```

Build finished

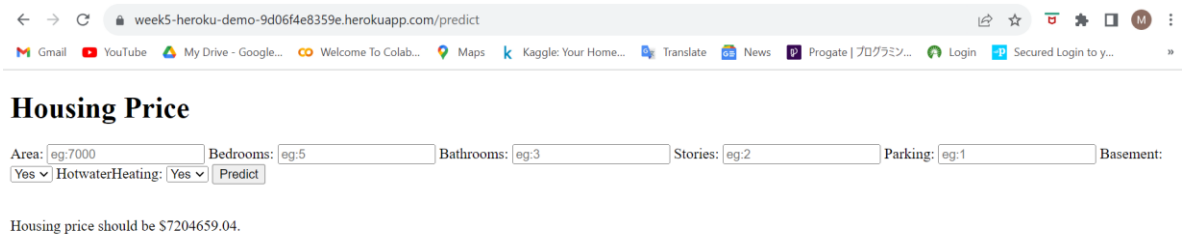
Step9.Generate the web link of the application

https://week5-heroku-demo-9d06f4e8359e.herokuapp.com/

Step10.Open the ML model in web



Step11.Test the ML model in web



Press the predict button. Then, the prediction result appears.