



Week 4: Deployment on Flask

Name: Niyusha Baghayi

Batch Code: LICAN01

Submission Data: 3/21/2021

Submitted to: Data Glacier

Table of Contents

Introduction.....2

Dataset.....4

Model.....5

Flask Deployment.....6

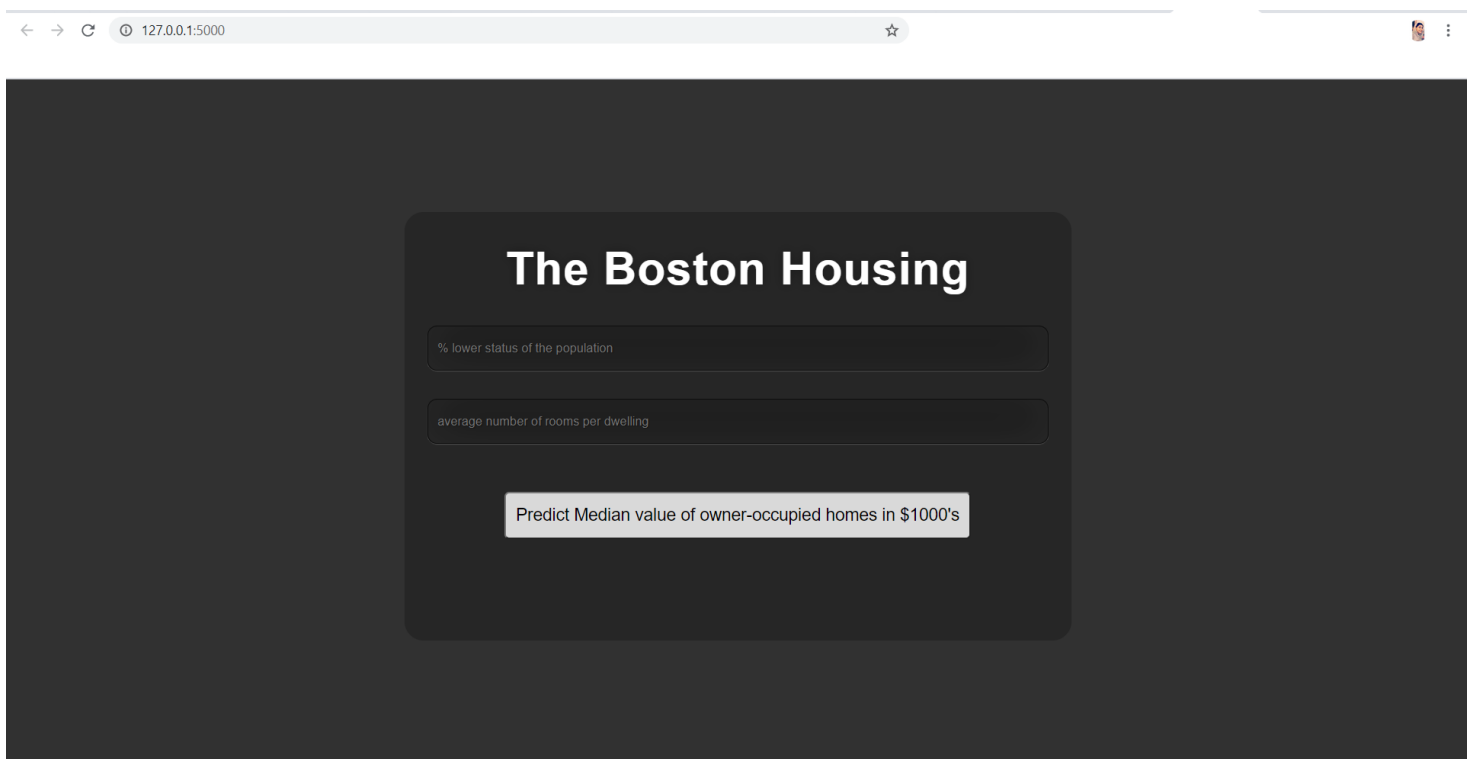
Running the Flask Web App10



Introduction

The flask web app which is implemented do in the way that get the value of two variables from clients and after that predict the result and show it. The dataset that I have used for this project is “Boston Housing” dataset, I trained a model to predict the “MEDV” by two features “LSTAT” and “RM”. In the following sections I will explain each part of the project separately in detail.

Here is the picture of the web app interface:



After a user put values and press the predict button, the answer will come in a short time:

← → ↻ 127.0.0.1:5000/predict ☆

The Boston Housing

3.13

8.040

Predict Median value of owner-occupied homes in \$1000's

MEDV (Median value of owner-occupied homes in \$1000's) should be \$ 37.59

Dataset

The dataset that I used it is “Boston Housing” Dataset. The below picture illustrates all features of this dataset.

The Boston Housing Dataset

The Boston Housing Dataset is a derived from information collected by the U.S. Census Service concerning housing in the area of [Boston MA](#). The following describes the dataset columns:

- CRIM - per capita crime rate by town
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.
- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling
- AGE - proportion of owner-occupied units built prior to 1940
- DIS - weighted distances to five Boston employment centres
- RAD - index of accessibility to radial highways
- TAX - full-value property-tax rate per \$10,000
- PTRATIO - pupil-teacher ratio by town
- B - $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town
- LSTAT - % lower status of the population
- MEDV - Median value of owner-occupied homes in \$1000's

¹ <https://www.kaggle.com/prasadperera/the-boston-housing-dataset>

Model

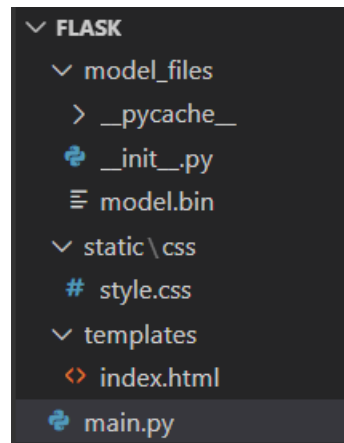
I have deployed a model to predict “MEDV” based on “RM” and “LSTAT” as they have almost linear correlation with the target (MEDV). In this regard I applied Linear Regression to predict the target and train the model with whole dataset as train data. You can find the code in the linked [picture](#).

The codes are written in Jupyter Notebook, at last I stored the model to the .bin file that I used it in the Flask.



Flask Deployment

For deploying Flask first of all we should install it, after that we can use it in python codes. The below picture show the structure of the flask deployment in this project:



As you can see here, we have some important files like .py, .html and .css files, we will explore each of these files in detail:

model.bin

Is the model that I stored as .bin file.

style.css

This file helps to have more beautiful view of this app. Here is the complete .css file:

```
static > css > # style.css > .login
Explorer (Ctrl+Shift+E) url(https://fonts.googleapis.com/css?family=Open+Sans);
2
3 html { width: 100%; height:100%; overflow:hidden; }
4
5 body {
6     width: 100%;
7     height:100%;
8     font-family: 'Helvetica';
9     background: #000;
10    color: #fff;
11    font-size: 24px;
12    text-align:center;
13    letter-spacing:1.4px;
14
15 }
16 .login {
17     position: absolute;
18     top: 40%;
19     left: 50%;
20     margin: -150px 0 0 -350px;
21     width:700px;
22     height:450px;
23     text-align: center;
24     background: #rgb(38, 38, 38);
25     border-radius: 20px;
26 }
27
28 .login h1, .login h2 { color: #fff; text-shadow: 0 0 10px #rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }
```

```
static > css > # style.css > body
29
30 input {
31     width: 90%;
32     height: 25px;
33     margin-bottom: 30px;
34     background: #rgba(0,0,0,0.3);
35     border: none;
36     outline: none;
37     padding: 10px;
38     font-size: 13px;
39     color: #fff;
40     text-shadow: 1px 1px 1px #rgba(0,0,0,0.3);
41     border: 1px solid #rgba(0,0,0,0.5);
42     border-radius: 10px;
43     box-shadow: inset 0 -5px 45px #rgba(100,100,100,0.2), 0 1px 1px #rgba(255,255,255,0.2);
44     -webkit-transition: box-shadow .5s ease;
45     -moz-transition: box-shadow .5s ease;
46     -o-transition: box-shadow .5s ease;
47     -ms-transition: box-shadow .5s ease;
48     transition: box-shadow .5s ease;
49     margin-left: 0;
50     margin-right: 0;
51 }
52
53 .btn_submit {
54     height: 50px;
55     border-radius: 5px;
56     font-size: 18px;
57     padding: 10px;
58     margin-top: 20px;
59     background: #d9d9d9;
60 }
61
62 .prediction_text{
63     font-size: 16px;
64 }
```


index.html

This is the main view page that contains all static elements which we can see in the UI as a client:

```
templates > index.html > html > head > title
15 <div class="login">
16 <h1>The Boston Housing</h1>
17
18 <!-- Main Input For Receiving Query to our ML -->
19 <form action="{{ url_for('predict')}}" method="post">
20   <input type="text" name="LSTAT" placeholder="% lower status of the population" required="required" />
21   <input type="text" name="RM" placeholder="average number of rooms per dwelling" required="required" />
22   <button type="submit" class="btn btn-primary btn-block btn-large btn_submit">Predict Median value of owner-occupied homes in $1000's</button>
23 </form>
24
25 <br>
26 <br>
27 <div class="prediction_text">{{ prediction_text }}</div>
28 </div>
29 </body>
30 </html>
```

main.py

This is the last and the most important file of this project, first of all I imported all necessary libraries, after that the model is read from the file, also we should declare something to do for each rout and also, we should declare some routs that we can see webpages only by their routes.

The first rout shows the index.html.

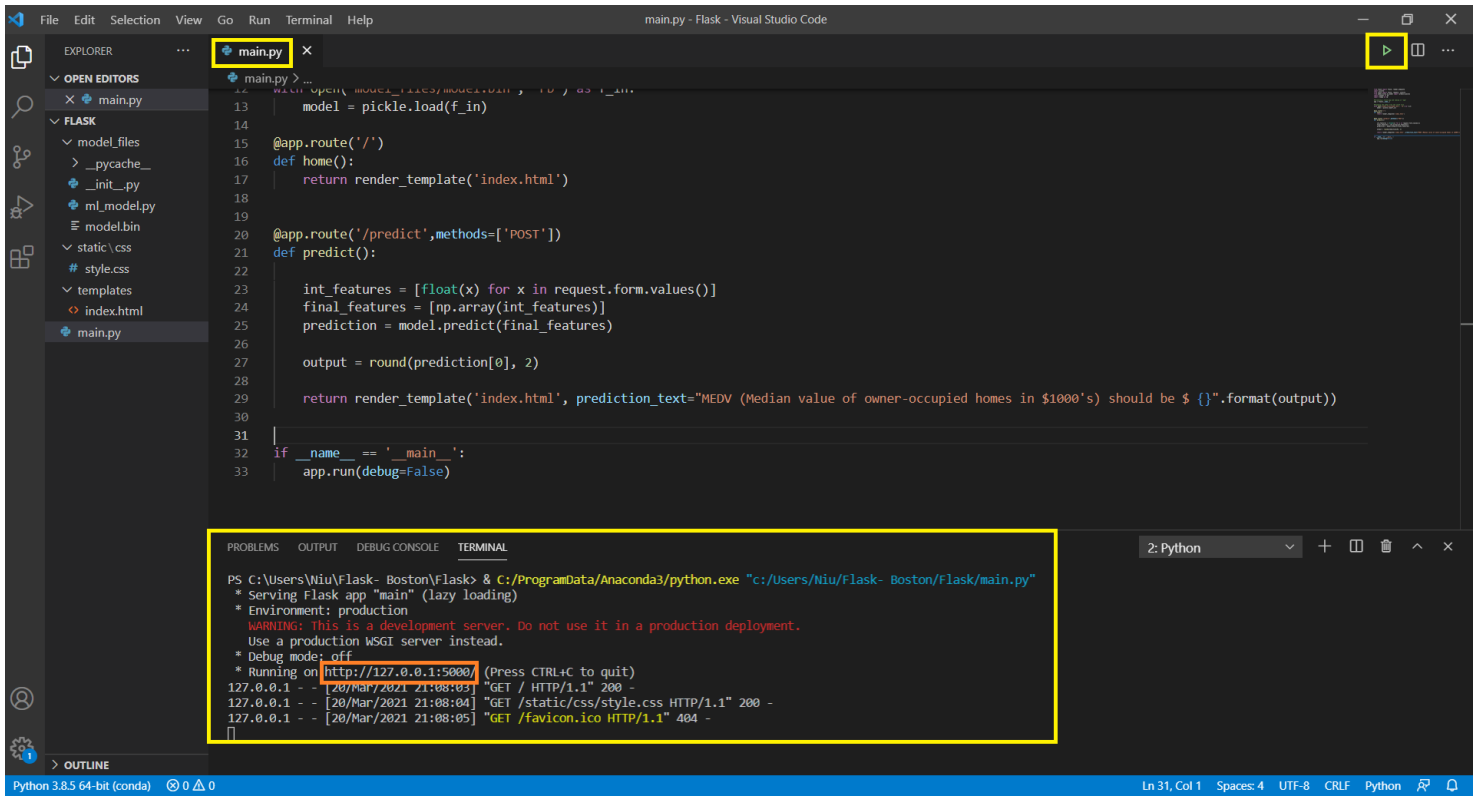
The second rout which has “/predict” and that is the destination URL of the form that mentioned in “index.html”, when the predict button is pressed the form leads us to this URL and the following steps will happen:

Read data from the form (the values for “LSTAT” and “RM” entered by user), use the model and predict the “MEDV”, at last load the “index.html” with the result value of “MEDV” and show it.

```
main.py x
main.py > ...
1  from flask import Flask, render_template
2  import pickle
3  from flask import Flask, request, jsonify
4  import pandas as pd
5  import numpy as np
6
7  ##creating a flask app and naming it "app"
8  app = Flask(__name__)
9
10 ##loading the model from the saved file
11 with open('model_files/model.bin', 'rb') as f_in:
12     model = pickle.load(f_in)
13
14 @app.route('/')
15 def home():
16     return render_template('index.html')
17
18
19 @app.route('/predict', methods=['POST'])
20 def predict():
21
22     int_features = [float(x) for x in request.form.values()]
23     final_features = [np.array(int_features)]
24     prediction = model.predict(final_features)
25
26     output = round(prediction[0], 2)
27
28     return render_template('index.html', prediction_text="MEDV (Median value of owner-occupied homes in $1000's) should be $ {}".format(output))
29
30
31 if __name__ == '__main__':
32     app.run(debug=False)
```

Running the Flask Web App

For running the application, we should run the main.py which is the python code and find the address of our application as it shows, we can change the URL of our application as well.



```
File Edit Selection View Go Run Terminal Help
main.py - Flask - Visual Studio Code

EXPLORER
main.py
OPEN EDITORS
main.py
FLASK
  model_files
  > __pycache__
  > _init_.py
  > ml_model.py
  > model.bin
  > static\css
  > style.css
  > templates
  > index.html
  > main.py

main.py
12 with open('model_files/model.bin', 'rb') as f_in:
13     model = pickle.load(f_in)
14
15 @app.route('/')
16 def home():
17     return render_template('index.html')
18
19
20 @app.route('/predict', methods=['POST'])
21 def predict():
22
23     int_features = [float(x) for x in request.form.values()]
24     final_features = [np.array(int_features)]
25     prediction = model.predict(final_features)
26
27     output = round(prediction[0], 2)
28
29     return render_template('index.html', prediction_text="MEDV (Median value of owner-occupied homes in $1000's) should be $ {}".format(output))
30
31
32 if __name__ == '__main__':
33     app.run(debug=False)
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\Niu\Flask- Boston\Flask> & C:/ProgramData/Anaconda3/python.exe "c:/Users/Niu/Flask- Boston/Flask/main.py"
* Serving Flask app "main" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [20/Mar/2021 21:08:03] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [20/Mar/2021 21:08:04] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [20/Mar/2021 21:08:05] "GET /favicon.ico HTTP/1.1" 404 -
```

Python 3.8.5 64-bit (conda) 0 0 0 Ln 31, Col 1 Spaces: 4 UTF-8 CRLF Python