Julia Donato LISUM17

ML Model Deployment on Flask

1. Found toy data USA_Housing.csv

	A	B	C	D	E	F	G
1	Avg. Area Inc	Avg. Area Ho	Avg. Area Nu	Avg. Area Nu	Area Populat	Price	Address
2	79545.4586	5.68286132	7.00918814	4.09	23086.8005	1059033.56	208 Michael
3	79248.6425	6.00289981	6.73082102	3.09	40173.0722	1505890.91	188 Johnson
4	61287.0672	5.86588984	8.51272743	5.13	36882.1594	1058987.99	9127
5	63345.24	7.18823609	5.58672866	3.26	34310.2428	1260616.81	USS Barnett
6	59982.1972	5.04055452	7.83938779	4.23	26354.1095	630943.489	USNS
7	80175.7542	4.98840776	6.10451244	4.04	26748.4284	1068138.07	06039
8	64698.4634	6.02533591	8.14775959	3.41	60828.2491	1502055.82	4759 Daniel

2. Wrote ML Model for toy data

```
import pandas as pd
import pickle
from sklearn.model_selection import train_test_split
# Read Data

df = pd.read_csv('USA_Housing.csv')

# Select independent and dependent variables

X = df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
        'Avg. Area Number of Bedrooms', 'Area Population',]]

y = df['Price']

# Split data into train and test

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_st

# Training the Model

from sklearn.linear_model import LinearRegression

lm = LinearRegression()

lm.fit(X_train,y_train)

# Make pickle file

pickle.dump(lm, open("model.pkl","wb"))
```

3. Created a server app.py to load the model and predict results

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

# Create flask app
app = Flask(__name__)

# Load pickle model
model = pickle.load(open('model.pkl', 'rb'))

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

@app.route('/predict', methods=['Post'])
def predict():
    int_features = [int(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)
    return render_template('index.html', prediction_text='House price should be $

if __name__ == "__main__":
    app.run(port=5000, debug=True)
```

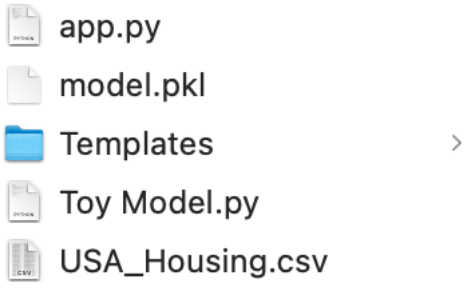
4. Created an HTML form, index.html containing all the different options to select from each attribute. Allows users to enter details and displays results

```
<!doctype html>
<html>

<head>
    <meta charset="UTF-8"
    <title>ML API</title>
</head>
<body>
 <div class= "Login">
    <h1>Predict House Price</h1>
<form action="{{ url_for('predict') }}" method="post">
        <legend>Input values:</legend>
        Avg. Area Income:
        <input name="Avg. Area Income" type="text" required>
        <br>
        <br> Avg. Area House Age:
        <input name="Avg. Area House Age" type="text" required>
        <br>
        <br> Avg. Area Number of Rooms:
        <input name="Avg. Area Number of Rooms" type="text" required>
        <br>
        <br> Avg. Area Number of Bedrooms:
        <input name="Avg. Area Number of Bedrooms" type="text" required>
        <br>
        <br> Area Population:
        <input name="Area Population" type="text" required>
        <br>
        <br>
        <input type="submit">
    </form>
    <br>
    <br>
    {{ prediction_text }}
</div>

<body>
</html>
```

5. Placed HTML file in file Templates



6. Ran app.py

```
In [1]: runfile('/Users/juliadonato/Desktop/Data Glacier/Week 4/app.py', wdir='/Users/
juliadonato/Desktop/Data Glacier/Week 4')
 * Serving Flask app "app" (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: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with watchdog (fsevents)
 * Debugger is active!
 * Debugger PIN: 240-319-905
```

7. Tested the web app

ML API

Predict House Price

Input values:
Avg. Area Income:

Avg. Area House Age:

Avg. Area Number of Rooms:

Avg. Area Number of Bedrooms:

Area Population:

Submit

House price should be $ 10567388.29