Name: Michael Udonna Egbuzobi

Batch code: LISUM36

Submission Date: 06-Sept-2024

Submitted to: Data Glacier

Cloud and API deployment on Heroku

Step 1:

Develop ML model:

The machine learning model was developed to predict house prices in King County, USA, utilizing a linear regression approach. To ensure efficiency and facilitate future use, the trained model was saved using pickle. This method allows for the model to be loaded and used later without needing to retrain it.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # MATLAB-like way of plotting
from scipy import stats
import pickle
# sklearn package for machine learning in python:
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.datasets import make_regression
from sklearn.metrics import mean_squared_error, r2_score
from mpl_toolkits.mplot3d import Axes3D

df = pd.read_csv('houseprice_data.csv')
print(df.head(20))

X = df.iloc[:, [1,3,8,12,13]].values
y = df.iloc[:, 0].values

df.drop('sqft_living15', axis=1, inplace=True)
df.drop('sqft_above', axis=1, inplace=True)

# Identify outliers using Z-score
z_scores = np.abs(stats.zscore(df[['sqft_living', 'price']]))
outliers = (z_scores > 2)

# Remove outliers
df_no_outliers = df[~outliers.any(axis=1)]

print("Original DataFrame shape:", df.shape)
print("New DataFrame shape:", df_no_outliers.shape)

# split the data into training and test sets:
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 1/3,
random_state=0)

# fit the linear least-squres regression line to the training data:
regr = LinearRegression()
regr.fit(X_train, y_train)
# The coefficients
print('Coefficients: ', regr.coef_)
# The intercept
print('Intercept: ', regr.intercept_)
# The mean squared error
print('Mean squared error: %.8f'
% mean_squared_error(y_test, regr.predict(X_test)))
# The R^2 value:
print('Coefficient of determination: %.2f'
% r2_score(y_test, regr.predict(X_test)))


# Save the trained model to a file using Pickle
with open('linear_regression_model.pkl', 'wb') as file:  # 'wb' means write in binary mode
    pickle.dump(regr, file)


print("Model has been saved to 'linear_regression_model.pkl'.")
```

Step 2:

Model Deployment on Flask

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

# Load your saved model
with open('linear_regression_model.pkl', 'rb') as f:
    model = pickle.load(f)

# Initialize the Flask app and specify the templates folder
app = Flask(__name__, template_folder='templates2')

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

@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Get data from the form
        bedrooms = float(request.form['bedrooms'])
        sqft_living = float(request.form['sqft_living'])
        condition = float(request.form['condition'])
        yr_built = float(request.form['yr_built'])
        yr_renovated = float(request.form['yr_renovated'])

        # Prepare data for prediction
        features = np.array([[bedrooms, sqft_living, condition, yr_built, yr_renovated]])

        # Predict using the loaded model
        prediction = model.predict(features)

        # Render the HTML template with the prediction result
        return render_template('index.html', prediction_text=f'Predicted House Price: ${prediction[0]:,.2f}')

    except ValueError as e:
        return render_template('index.html', prediction_text='Error: Please ensure all input values are numeric.')

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

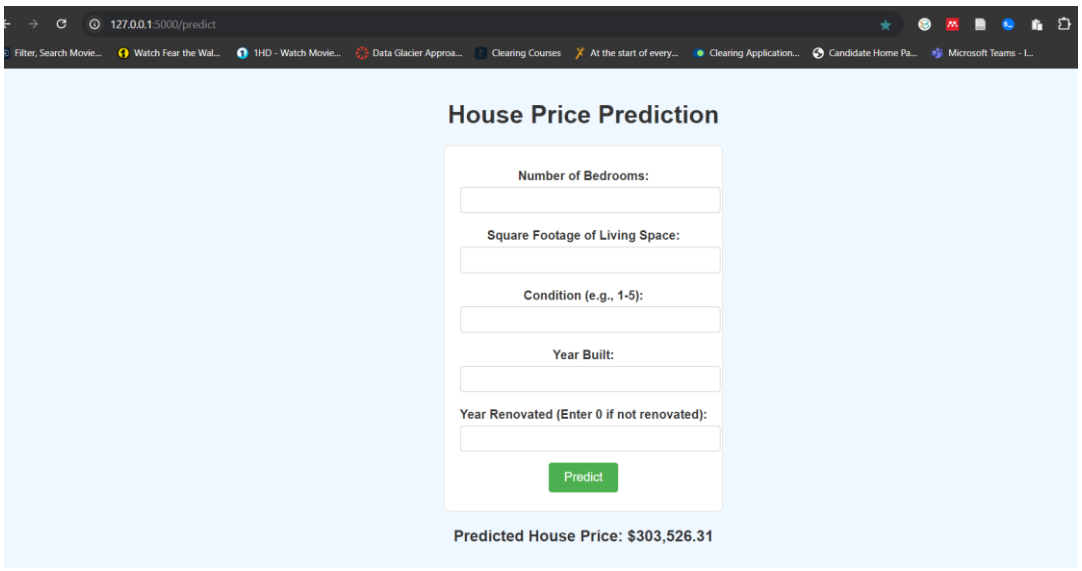
Step 3:

Checking python app.py file in CMD

```
C:\Users\egbuz>C:\Users\egbuz\AppData\Local\Programs\Python\Python312\python.exe app.py
C:\Users\egbuz\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:376: InconsistentVersionWarning: Trying to unpickle estimator Linea
rRegression from version 1.2.2 when using version 1.5.1. This might lead to breaking code or invalid results. Use at your own risk. For more info please ref
er to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
C:\Users\egbuz\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\base.py:376: InconsistentVersionWarning: Trying to unpickle estimator Linea
rRegression from version 1.2.2 when using version 1.5.1. This might lead to breaking code or invalid results. Use at your own risk. For more info please ref
er to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
 * Debugger is active!
 * Debugger PIN: 537-232-541
```

Step 4:

The web application has been successfully developed and is functioning as intended. An example of its accurate prediction capability is demonstrated by the predicted value of $303,526.31.



Step 5:

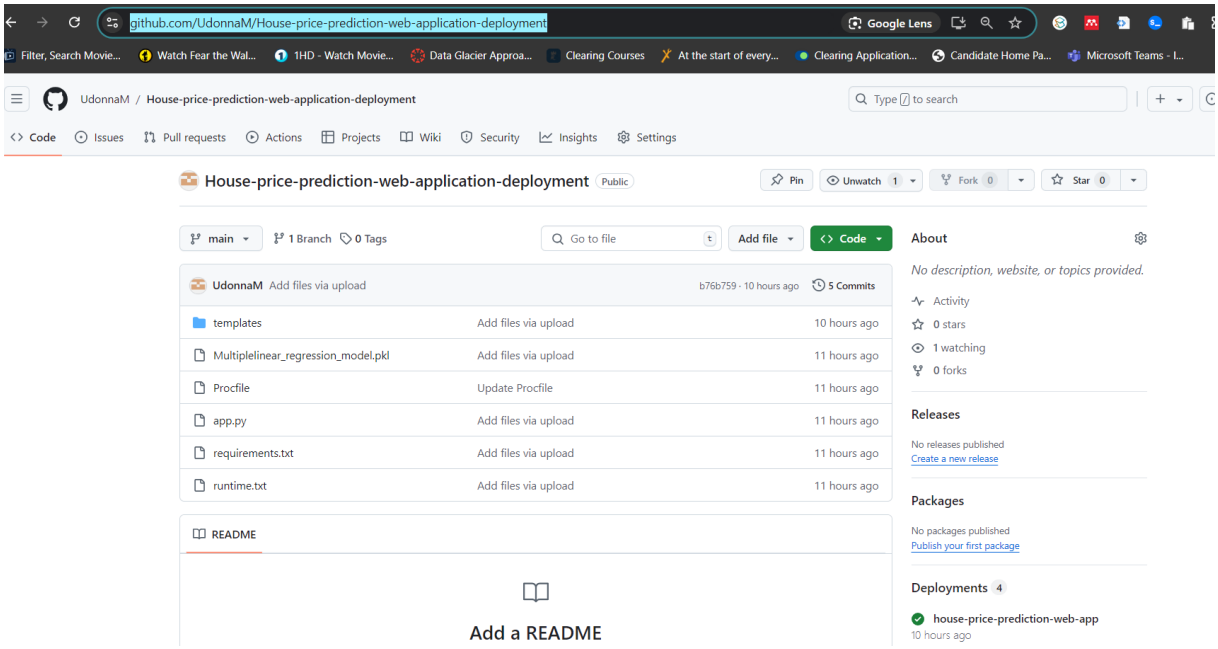
Create Procfile file which specifies the commands that are executed by a Heroku app on start-up. web: gunicorn app:app.

Running the command pip freeze > requirements.txt in CMD for creating requirement.txt file which will contain all of the dependencies for the flask app.

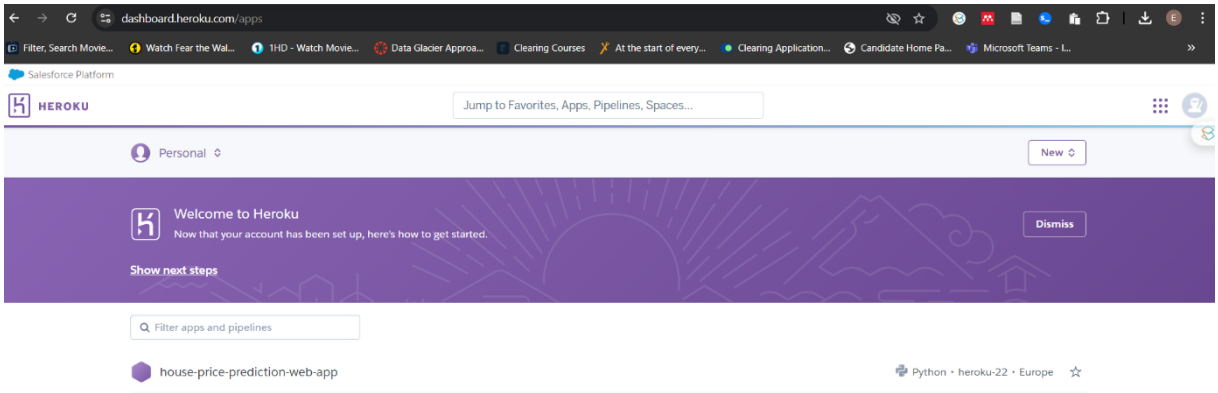
Step 6:

Create a repository in Github and Commit the code

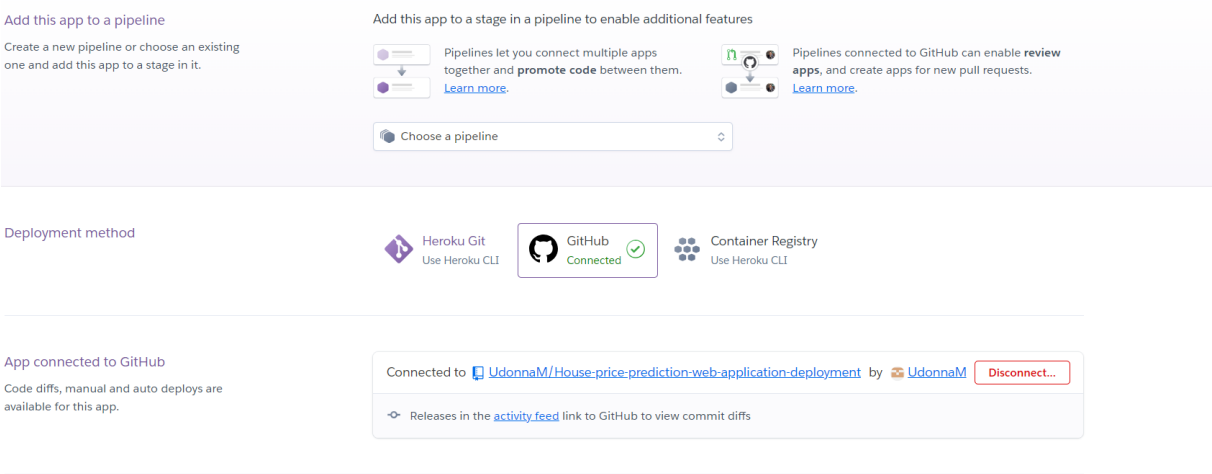
Link: https://github.com/UdonnaM/House-price-prediction-web-application-deployment



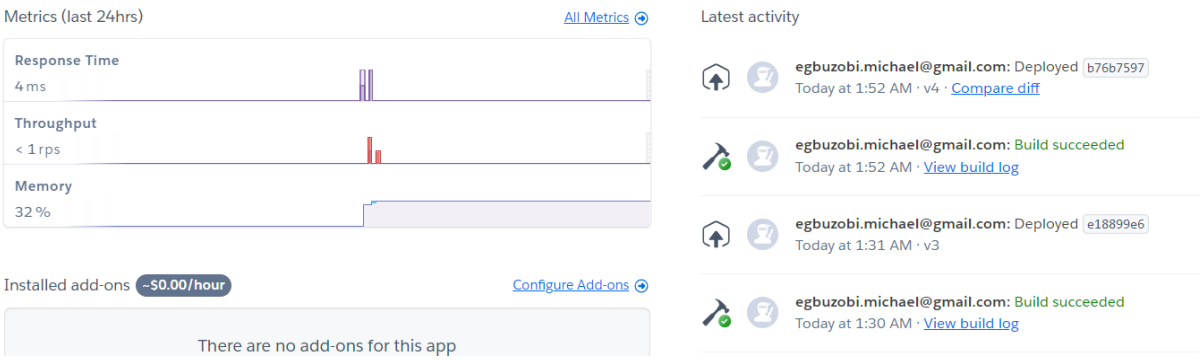
Step 7: Create an Account in Heroku and then create an app.



Step 8: Link Github Account with Heroku app



Step 9: App successfully deployed



Thank you.