



## • Dataset Selection

Housing price dataset is selected which contains the columns namely price, area bedrooms, bathrooms and stories. Taking the price value as the output and all other as input a simple linear regression model is created.

	A	B	C	D	E	F
1	price	area	bedrooms	bathroom	stories	
2	13300000	7420	4	2	3	
3	12250000	8960	4	4	4	
4	12250000	9960	3	2	2	
5	12215000	7500	4	2	2	
6	11410000	7420	4	1	2	
7	10850000	7500	3	3	1	
8	10150000	8580	4	3	4	
9	10150000	16200	5	3	2	
10	9870000	8100	4	1	2	
11	9800000	5750	3	2	4	
12	9800000	13200	3	1	2	
13	9681000	6000	4	3	2	
14	9310000	6550	4	2	2	
15	9240000	3500	4	2	2	
16	9240000	7800	3	2	2	
17	9100000	6000	4	1	2	
18	9100000	6600	4	2	2	
19	8960000	8500	3	2	4	
20	8890000	4600	3	2	2	
21	8855000	6420	3	2	2	
22	8750000	4320	3	1	2	
23	8680000	7155	3	2	1	
24	8645000	8050	3	1	1	
25	8645000	4560	3	2	2	
26	8575000	8800	3	2	2	
27	8540000	6540	4	2	2	
28	8463000	6000	3	2	4	
29	8400000	8875	3	1	1	
30	8400000	7950	5	2	2	
31	8400000	5500	4	2	2	
32	8400000	7475	3	2	4	
33	8400000	7000	3	1	4	
34	8295000	4880	4	2	2	
35	8190000	5960	3	3	2	
36	8120000	6840	5	1	2	
37	8080940	7000	3	2	4	
38	8043000	7482	3	2	3	
39	7980000	9000	4	2	4	
40	7962500	6000	3	1	4	

◀ ▶
Housing
⊕

Ready
 
 Accessibility: Unavailable


 10°C

## • Model Selection

After trying various regression models such as Linear Regression, Decision Tree Regressor, Random Forest Regressor and Gradient Boosting Regressor based on the cross validation score linear regression model is selected which is then trained on the splitted dataset into training and test. And a pickle file is then generated within the notebook namely model\_1.pkl.

```
Housing_price_prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, ExtraTreesRegressor, GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import mean_absolute_error, r2_score, mean_squared_error
from sklearn.model_selection import train_test_split

import pickle

from sklearn.model_selection import cross_val_score, cross_val_predict

[41] df=pd.read_csv('Housing.csv')
print(df.head())

   price  area  bedrooms  bathrooms  stories
0  13300000  7420         4           2         3
1  12250000  8960         4           4         4
2  12250000  9960         3           2         2
3  12215000  7500         4           2         2
4  11410000  7420         4           1         2

[42] X=df[['area','bedrooms','bathrooms','stories']].values
y=df['price'].values
```

```
[43] x_train, x_test, y_train, y_test = train_test_split(X, y, train_size=0.8, test_size=0.2, random_state=100)

regression = {'LINEAR':LinearRegression(),
              'DECISION':DecisionTreeRegressor(),
              'FOREST':RandomForestRegressor(n_estimators=100, max_depth=10),
              'GRADIENT':GradientBoostingRegressor(n_estimators=100, max_depth=10),
              'EXTRA':ExtraTreesRegressor(n_estimators=100, max_depth=10)}

for nome, model in regression.items():
    print('>=<=*20')
    print('NOME:', nome)

    score = cross_val_score(model, x_train, y_train, cv=5, n_jobs=-1)

    y_pred = cross_val_predict(model, x_train, y_train, cv=5, n_jobs=-1)

    print('Cross Val Score:',score.mean())

NOME: LINEAR
Cross Val Score: 0.5258095932067959
NOME: DECISION
Cross Val Score: -0.08082370435521644
NOME: FOREST
Cross Val Score: 0.33605145236900544
NOME: GRADIENT
Cross Val Score: -0.01040313814675924
NOME: EXTRA
Cross Val Score: 0.34545068789643446
```

```
NOME: EXTRA
Cross Val Score: 0.34545068789643446

[47] model=LinearRegression()

model.fit(x_train,y_train)

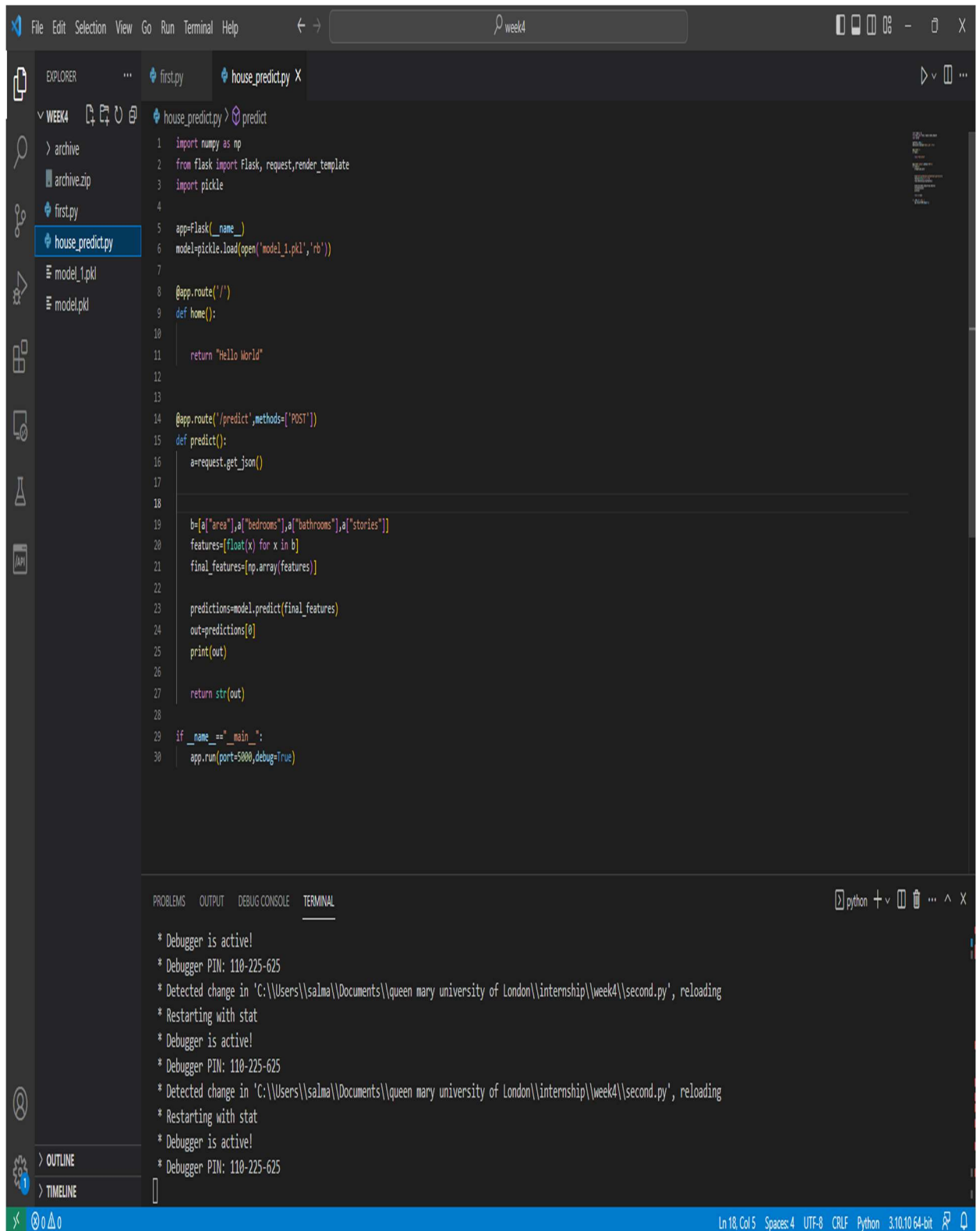
[48] y_predict = model.predict(x_test)

pickle.dump(model,open('model_1.pkl','wb'))

[ ]
```

## • API Creation

Using Flask framework a post api is created which takes values of area, number of bedrooms, number of bathrooms, number of stories in the form of json and then return the predicted price of the house in US Dollars.



```
File Edit Selection View Go Run Terminal Help
week4

EXPLORER
WEEK4
  archive
  archive.zip
  first.py
  house_predict.py
  model_1.pkl
  model.pkl

house_predict.py > predict
1 import numpy as np
2 from flask import Flask, request, render_template
3 import pickle
4
5 app=Flask(__name__)
6 model=pickle.load(open('model_1.pkl','rb'))
7
8 @app.route('/')
9 def home():
10     return "Hello World"
11
12
13
14 @app.route('/predict',methods=['POST'])
15 def predict():
16     a=request.get_json()
17
18
19     b=[a["area"],a["bedrooms"],a["bathrooms"],a["stories"]]
20     features=[float(x) for x in b]
21     final_features=np.array(features)
22
23     predictions=model.predict(final_features)
24     out=predictions[0]
25     print(out)
26
27     return str(out)
28
29 if __name__=="__main__":
30     app.run(port=5000,debug=True)

TERMINAL
python
* Debugger is active!
* Debugger PIN: 110-225-625
* Detected change in 'C:\\Users\\salma\\Documents\\queen mary university of London\\internship\\week4\\second.py', reloading
* Restarting with stat
* Debugger is active!
* Debugger PIN: 110-225-625
* Detected change in 'C:\\Users\\salma\\Documents\\queen mary university of London\\internship\\week4\\second.py', reloading
* Restarting with stat
* Debugger is active!
* Debugger PIN: 110-225-625
```

- **API Functioning**

The rest Api is then ran using postman as the method used is post and which directly does not run in browser. To avoid the creation of html form postman is used for convinence. The result is then displayed in the response.

