BOSTON HOUSING PRICE PREDICTOR

➡ House_price_prediction.pkl

--Code

(1) Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from xgboost import XGBRegressor
```

(2) Dataset Loading

```
dataset=pd.read_csv('/content/BostonHousing.csv')
dataset
```

output:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	price
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9

506 rows × 14 columns

(3) Basic Functions

dataset.shape

dataset.head()

dataset.tail()

(4) Finding Null Values

dataset.isnull().sum()

output:

	0
crim	0
zn	0
indus	0
chas	0
nox	0
rm	0
age	0
dis	0
rad	0
tax	0
ptratio	0
b	0
lstat	0
price	0

dtype: int64

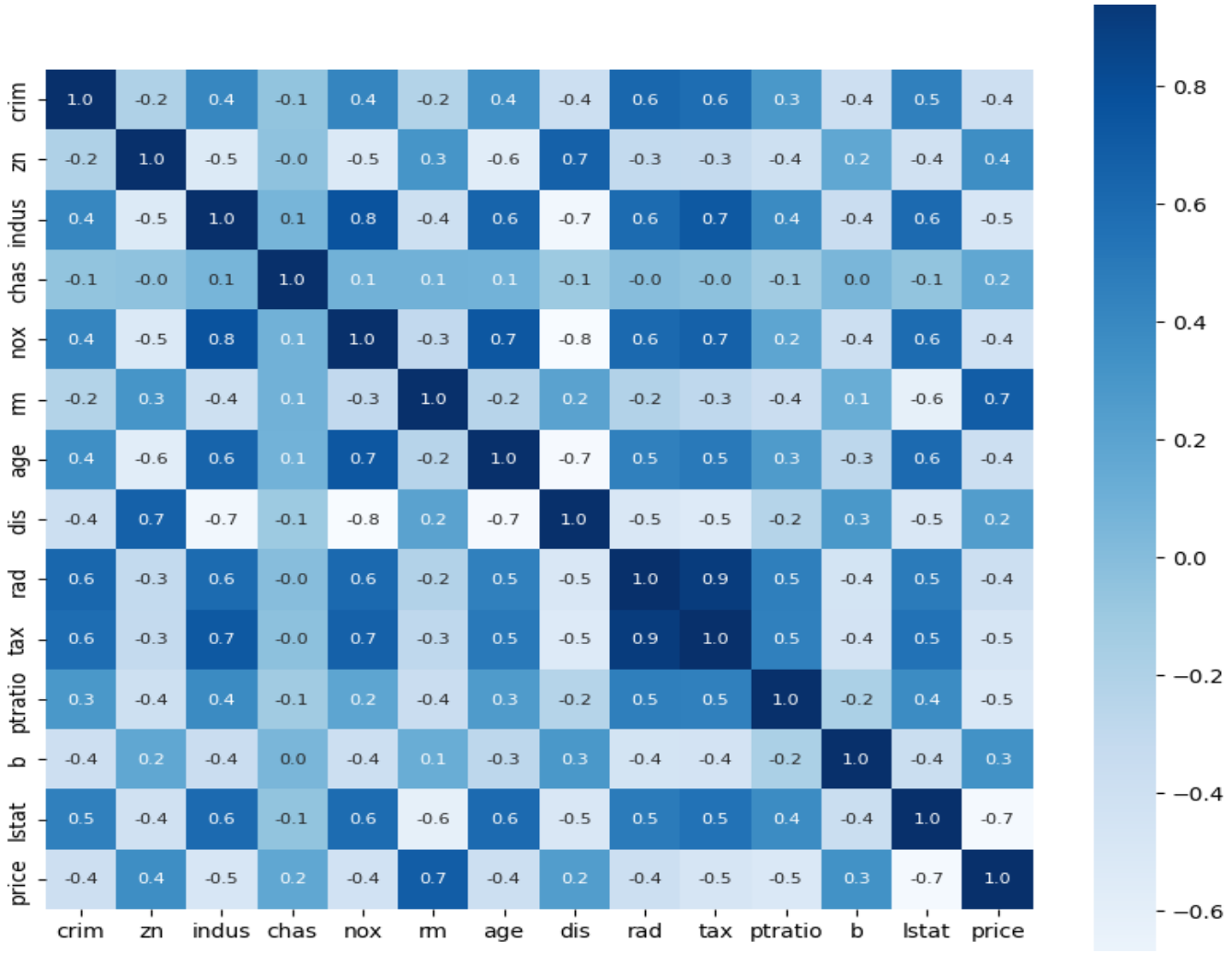
(5) Data Exploration

correlation = dataset.corr()

plt.figure(figsize=(10,10))

sns.heatmap(correlation,cbar=True,square=True,fmt='.1f',annot=True,annot_kws={'size':8},cmap='Blues')

output:



(6) Data Splitting

X=dataset.drop('price',axis=1)

Y=dataset['price']

X

Output:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33

(7) Training and Testing

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=31)
print(X.shape,X_train.shape,X_test.shape)
```

output:

```
(506, 13) (404, 13) (102, 13)
```

(8) Model Fitting

```
model2=XGBRegressor()
model2.fit(X_train,Y_train)
```

output:

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

(9) Metrics

```
model_prediction=model2.predict(X_train)
score1=metrics.r2_score(Y_train,model_prediction)
print("R2 score = ",score1)
```

output:

```
R2 score =  0.9999971222391121
```

```
score2=metrics.mean_absolute_error(Y_train,model_prediction)
print("Mean Absolute Error = ",score2)
```

output:

```
Mean Absolute Error =  0.0111977718844272
```

(10) Model Evalution

input=np.array([[0.04527, 0.0, 11.93, 0, 0.573, 6.120, 76.7, 2.2875, 1 ,273 ,21.0 ,396.90, 9.08]])

model2.predict(input)

output:

```
array([20.597471], dtype=float32)
```

Flask FrameWork

App.py

```
from flask import Flask,render_template,request
import pickle
import numpy as np
app=Flask(__name__)
with open('house_price_prediction.pkl','rb') as f:
    model2=pickle.load(f);
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict',methods=['POST'])
def predict():
    features=[
        float(request.form['CRIM']),
        float(request.form['ZN']),
        float(request.form['INDUS']),
        float(request.form['CHAS']),
        float(request.form['NOX']),
        float(request.form['RM']),
        float(request.form['AGE']),
        float(request.form['DIS']),
        float(request.form['RAD']),
```

```
        float(request.form['TAX']),
        float(request.form['PTRATIO']),
        float(request.form['B']),
        float(request.form['LSTAT']),
    ]
    features_array =np.array([features])
    prediction=model2.predict(features_array)
    output=round(prediction[0],2)
    return render_template('index.html',prediction_text=f"predicted price: {output}")
if __name__ == "__main__":
     app.run(debug=True)
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Boston Housing Price Prediction</title>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            font-family: 'Poppins', sans-serif;
        }

        body {
            background: linear-gradient(135deg, #667eea, #764ba2);
            min-height: 100vh;
            display: flex;
            justify-content: center;
```

```
    align-items: center;
}

.container {
    background: rgba(255, 255, 255, 0.1);
    backdrop-filter: blur(12px);
    padding: 40px;
    border-radius: 20px;
    box-shadow: 0 12px 30px rgba(0, 0, 0, 0.3);
    width: 100%;
    max-width: 500px;
    color: #fff;
}

h2 {
    text-align: center;
    margin-bottom: 30px;
    font-weight: 600;
    color: #f0f0f0;
}

label {
    font-size: 14px;
    display: block;
    margin: 10px 0 5px;
}

input[type="text"] {
    width: 100%;
    padding: 10px 15px;
    border: none;
    border-radius: 12px;
    background-color: rgba(255, 255, 255, 0.2);
    color: #fff;
    transition: 0.3s;
```

```
        }

        input[type="text"]:focus {
            outline: none;
            background-color: rgba(255, 255, 255, 0.3);
            box-shadow: 0 0 8px #a29bfe;
        }

        input[type="submit"] {
            margin-top: 20px;
            width: 100%;
            padding: 12px;
            background-color: #6c5ce7;
            color: white;
            border: none;
            border-radius: 12px;
            font-size: 16px;
            font-weight: 600;
            cursor: pointer;
            transition: background-color 0.3s ease;
        }

        input[type="submit"]:hover {
            background-color: #4834d4;
        }

        .prediction {
            margin-top: 25px;
            text-align: center;
            font-size: 18px;
            color: #ffeaa7;
        }
    </style>
</head>
<body>
```

```
<div class="container">
    <h2>🏠 Boston Housing Price Predictor</h2>
    <form action="/predict" method="post">
        <label>CRIM:</label>
        <input type="text" name="CRIM">
        <label>ZN:</label>
        <input type="text" name="ZN">
        <label>INDUS:</label>
        <input type="text" name="INDUS">
        <label>CHAS:</label>
        <input type="text" name="CHAS">
        <label>NOX:</label>
        <input type="text" name="NOX">
        <label>RM:</label>
        <input type="text" name="RM">
        <label>AGE:</label>
        <input type="text" name="AGE">
        <label>DIS:</label>
        <input type="text" name="DIS">
        <label>RAD:</label>
        <input type="text" name="RAD">
        <label>TAX:</label>
        <input type="text" name="TAX">
        <label>PTRATIO:</label>
        <input type="text" name="PTRATIO">
        <label>B:</label>
        <input type="text" name="B">
        <label>LSTAT:</label>
        <input type="text" name="LSTAT">
        <input type="submit" value="Predict">
    </form>

    {% if prediction_text %}
        <div class="prediction">{{ prediction_text }}</div>
    {% endif %}
```

```
    </div>
</body>
</html>
```

Output: