In [1]:

```
import pandas as pd
```

In [3]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [5]:

```
df = pd.read_csv('BostonHousing.csv')
df
```

Out[5]: crim zn indus chas nox rm age dis rad tax ptratio b lstat medv 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

In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   crim     506 non-null    float64
 1   zn       506 non-null    float64
 2   indus    506 non-null    float64
 3   chas     506 non-null    int64  
 4   nox      506 non-null    float64
 5   rm       501 non-null    float64
 6   age      506 non-null    float64
 7   dis      506 non-null    float64
 8   rad      506 non-null    int64  
 9   tax      506 non-null    int64  
 10  ptratio  506 non-null    float64
 11  b        506 non-null    float64
 12  lstat    506 non-null    float64
 13  medv     506 non-null    float64
dtypes: float64(11), int64(3)
memory usage: 55.5 KB
```

In [17]:

```
df.head(10)
```

Out[17]: crim zn indus chas nox rm age dis rad tax ptratio b lstat medv 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 5 0.02985 0.0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 394.12 5.21 28.7 6 0.08829 12.5 7.87 0 0.524 6.012 66.6 5.5605 5 311 15.2 395.60 12.43 22.9 7 0.14455 12.5 7.87 0 0.524

6.172 96.1 5.9505 5 311 15.2 396.90 19.15 27.1 8 0.21124 12.5 7.87 0 0.524 5.631 100.0 6.0821 5 311 15.2 386.63 29.93 16.5 9 0.17004 12.5 7.87 0 0.524 6.004 85.9 6.5921 5 311 15.2 386.71 17.10 18.9
In [26]:

```
## Independent features and dependent features
X=df
y=df.medv
```

In [40]:

```
##Dividing the dataset into independent and dependent features
X=df.iloc[:,:-1]##independent features
y=df.iloc[:,-1]##dependent features
```

In [41]:

```
X.head()
```

Out[41]: 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
In [27]:

```
y
```

Out[27]:

```
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
       ...
501    22.4
502    20.6
503    23.9
504    22.0
505    11.9
Name: medv, Length: 506, dtype: float64
```

In [28]:

```
## train test split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.30, random_state=42)
```

In [29]:

```
X_train
```

Out[29]: crim zn indus chas nox rm age dis rad tax ptratio b lstat medv 5 0.02985 0.0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 394.12 5.21 28.7 116 0.13158 0.0 10.01 0 0.547 6.176 72.5 2.7301 6 432 17.8 393.30 12.04 21.2 45 0.17142 0.0 6.91 0 0.448 5.682 33.8 5.1004 3 233 17.9 396.90 10.21 19.3 16 1.05393 0.0 8.14 0 0.538 5.935 29.3 4.4986 4 307 21.0 386.85 6.58 23.1 468 15.57570 0.0 18.10 0 0.580 5.926 71.0 2.9084 24 666 20.2 368.74 18.13 19.1 ... ... 106 0.17120 0.0 8.56 0 0.520 5.836 91.9 2.2110 5 384 20.9 395.67 18.66 19.5 270 0.29916 20.0 6.96 0 0.464 5.856 42.1 4.4290 3 223 18.6 388.65 13.00 21.1 348 0.01501 80.0 2.01 0 0.435 6.635 29.7 8.3440 4 280 17.0 390.94 5.99 24.5 435 11.16040 0.0 18.10 0 0.740 6.629 94.6 2.1247 24 666 20.2 109.85 23.27 13.4 102 0.22876 0.0 8.56 0 0.520 6.405 85.4 2.7147 5 384 20.9 70.80 10.63 18.6

354 rows × 14 columns

In [30]:

```
y_train
```

Out[30]:

```
5      28.7
116    21.2
45     19.3
16     23.1
468    19.1
       ...
106    19.5
270    21.1
348    24.5
435    13.4
102    18.6
Name: medv, Length: 354, dtype: float64
```

In [32]:

```
X_test
```

Out[32]: crim zn indus chas nox rm age dis rad tax ptratio b lstat medv 173 0.09178 0.0 4.05 0 0.510 6.416 84.1 2.6463 5 296 16.6 395.50 9.04 23.6 274 0.05644 40.0 6.41 1 0.447 6.758 32.9 4.0776 4 254 17.6 396.90 3.53 32.4 491 0.10574 0.0 27.74 0 0.609 5.983 98.8 1.8681 4 711 20.1 390.11 18.07 13.6 72 0.09164 0.0 10.81 0 0.413 6.065 7.8 5.2873 4 305 19.2 390.91 5.52 22.8 452 5.09017 0.0 18.10 0 0.713 6.297 91.8 2.3682 24 666 20.2 385.09 17.27 16.1 ... ... 441 9.72418 0.0 18.10 0 0.740 6.406 97.2 2.0651 24 666 20.2 385.96 19.52 17.1 23 0.98843 0.0 8.14 0 0.538 5.813 100.0 4.0952 4 307 21.0 394.54 19.88 14.5 225 0.52693 0.0 6.20 0 0.504 8.725 83.0 2.8944 8 307 17.4 382.00 4.63 50.0 433 5.58107 0.0 18.10 0 0.713 6.436 87.9 2.3158 24 666 20.2 100.19 16.22 14.3 447 9.92485 0.0 18.10 0 0.740 6.251 96.6 2.1980 24 666 20.2 388.52 16.44 12.6

152 rows × 14 columns

In [33]:

```
y_test
```

Out[33]:

```
173    23.6
274    32.4
491    13.6
72     22.8
452    16.1
       ...
441    17.1
23     14.5
225    50.0
433    14.3
447    12.6
Name: medv, Length: 152, dtype: float64
```

In [34]:

```
## standardizing the dataset
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

In [37]:

```
x_train=scaler.fit_transform(X_train)
```

In [42]:

```
X_test=scaler.transform(X_test)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:486: UserWarning: X has featu
  warnings.warn(
```

In [43]:

```
scaler.inverse_transform(X_train)
```

Out[43]:

```
array([[-0.41425879, -0.50512499, -1.29214218, ...,  0.39651419,
        -1.01531611,  0.60629225],
       [-0.40200818, -0.50512499, -0.16208345, ...,  0.3870674 ,
        -0.05366252, -0.19368088],
       [-0.39721053, -0.50512499, -0.60948856, ...,  0.42854113,
        -0.31132373, -0.39634074],
       ...,
       [-0.41604586,  3.03838247, -1.3166773 , ...,  0.35987906,
        -0.90549329,  0.1583073 ],
       [ 0.92611293, -0.50512499,  1.00549958, ..., -2.87841346,
         1.52750437, -1.02565293],
       [-0.39030549, -0.50512499, -0.37135358, ..., -3.32828832,
        -0.25218837, -0.4710049 ]])
```

In [45]:

```
from sklearn.linear_model import LinearRegression
##cross validation
from sklearn.model_selection import cross_val_score
```

In [64]:

```
from sklearn.impute import SimpleImputer
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression

# Fill missing values with the mean of each column
imputer = SimpleImputer(strategy='mean')
X_train_imputed = imputer.fit_transform(x_train)

# Perform cross-validation
regression = LinearRegression()
scores = cross_val_score(regression, X_train_imputed, y_train, scoring='neg_mean_squared_
print("Cross-validation scores:", scores)
```

```
Cross-validation scores: [-1.22343561e-27 -5.56409892e-28 -7.60900169e-28 -5.11613549e-27
 -4.77756703e-28 -1.87622678e-27 -1.10703781e-26 -8.77866954e-28
 -3.14507573e-27 -5.10967747e-27]
```

In [63]:

```
np.mean(scores)
```

Out[63]:

```
-3.021386289067014e-27
```

In [69]:

```
from sklearn.pipeline import Pipeline

pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='mean')),
    ('regressor', LinearRegression())
])

# Fit the pipeline
pipeline.fit(X_train, y_train)

# Predict
reg_pred = pipeline.predict(X_test)
```

In [70]:

```
reg_pred
```

Out[70]:

```
array([244.27325077, 326.77602186, 150.52010181, 236.77299886,
       173.95838905, 210.52211714, 189.89642437, 154.27022776,
       206.77199119, 180.52110947, 224.58508949, 200.20927076,
        88.64302349, 221.77249502, 196.4591448 , 302.40020313,
       199.27173927, 118.64403116, 491.78156405, 155.20775925,
       259.27375461, 295.83748271, 142.0823184 , 233.0228729 ,
       156.14529074, 152.39516478, 213.33471161, 162.70801117,
       226.46015247, 194.58408182, 239.58559332, 246.14831375,
       163.64554266, 218.02236906, 202.08433374, 204.89692821,
       348.33924613, 205.8344597 , 251.77350269, 242.39818779,
       207.70952268, 287.3996993 , 491.78156405, 186.14629841,
       234.89793588, 164.58307415, 145.83244436, 249.89843971,
       209.58458565, 248.02337673, 200.20927076, 354.90196655,
       165.52060564, 271.46166397, 430.84201722, 221.77249502,
       195.52161331, 290.21229377, 247.08584524, 196.4591448 ,
       257.39869163, 354.90196655, 318.33823846, 212.39718012,
       248.96090822, 210.52211714, 145.83244436, 255.52362865,
       311.77551803, 142.0823184 , 210.52211714, 245.21078226,
       124.26922009, 216.14730608, 218.02236906,  69.89239369,
       211.45964863, 477.7185917 , 125.20675158,  88.64302349,
       218.95990055, 184.27123543, 218.95990055, 113.95637371,
       204.89692821, 294.89995122, 176.77098352, 257.39869163,
       257.39869163, 183.33370394, 240.52312481, 120.51909414,
       206.77199119, 184.27123543, 280.83697887, 238.64806183,
       491.78156405, 190.83395586, 113.01884222, 184.27123543,
       233.96040439, 223.647558  , 135.51959797, 209.58458565,
       204.89692821, 148.64503883, 193.64655033, 253.64856567,
       220.83496353, 254.58609716, 104.58105881, 280.83697887,
       217.08483757, 362.40221847, 319.27576995, 132.7070035 ,
       396.1533521 , 153.33269627, 227.39768396, 245.21078226,
       188.02136139, 251.77350269, 105.5185903 , 203.02186523,
       260.2112861 , 214.2722431 , 239.58559332, 378.3402538 ,
       169.2707316 , 448.65511552, 170.20826309, 234.89793588,
       158.95788521, 198.33420778, 189.89642437, 173.95838905,
       216.14730608, 319.27576995, 295.83748271, 169.2707316 ,
       187.0838299 , 233.96040439, 204.89692821, 203.95939672,
       102.70599583, 216.14730608, 182.39617245, 183.33370394,
       158.95788521, 491.78156405, 157.08282223, 141.14478691])
```

In [71]:

```
import seaborn as sns
sns.displot(reg_pred-y_test,kind='kde')
```

Out[71]:

```
<seaborn.axisgrid.FacetGrid at 0x1cecde30a10>
```

In [72]:

```
from sklearn.metrics import r2_score
```

In [73]:

```
score=r2_score(reg_pred,y_test)
```

In [74]:

```
score
```

Out[74]:

```
-6.047563639377269
```

In [ ]: