lementation

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
#california House Pricing dataset
from sklearn.datasets import fetch_california_housing
california_df=fetch_california_housing()
```

In [3]:
```
california_df
```

Out[3]:
```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55555556,
           37.88      , -122.23      ],
        [   8.3014    ,   21.        ,    6.23813708, ...,    2.10984183,
           37.86      , -122.22      ],
        [   7.2574    ,   52.        ,    8.28813559, ...,    2.80225989,
           37.85      , -122.24      ],
        ...,
        [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
           39.43      , -121.22      ],
        [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
           39.43      , -121.32      ],
        [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
           39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing dataset\n--------------
------------\n\n**Data Set Characteristics:**\n\n    :Number of Instances: 20640\n\n
:Number of Attributes: 8 numeric, predictive attributes and the target\n\n    :Attribute
Information:\n        - MedInc        median income in block group\n        - HouseAge
    median house age in block group\n        - AveRooms      average number of rooms per
household\n        - AveBedrms     average number of bedrooms per household\n        - P
opulation    block group population\n        - AveOccup      average number of household
members\n        - Latitude      block group latitude\n        - Longitude     block gro
up longitude\n\n    :Missing Attribute Values: None\n\nThis dataset was obtained from th
e StatLib repository.\nhttps://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html\n\nT
he target variable is the median house value for California districts,\nexpressed in hun
dreds of thousands of dollars ($100,000).\n\nThis dataset was derived from the 1990 U.S.
census, using one row per census\nblock group. A block group is the smallest geographica
l unit for which the U.S.\nCensus Bureau publishes sample data (a block group typically
has a population\nof 600 to 3,000 people).\n\nA household is a group of people residing
within a home. Since the average\nnumber of rooms and bedrooms in this dataset are provi
ded per household, these\ncolumns may take surprisingly large values for block groups wi
th few households\nand many empty houses, such as vacation resorts.\n\nIt can be downloa
ded/loaded using the\n:func:`sklearn.datasets.fetch_california_housing` function.\n\n..
topic:: References\n\n    - Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregress
ions,\n      Statistics and Probability Letters, 33 (1997) 291-297\n'}
```

In [4]:
```
#independent features
x=pd.DataFrame(california_df.data,columns=california_df.feature_names)
#dependent features
y=california_df.target
```

In [5]:
```
x.head()
```

Out[5]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25

In [6]:
```
#train tes split
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=55)
```

In [7]:
```
from sklearn.tree import DecisionTreeRegressor
regressor=DecisionTreeRegressor()
```

In [8]:
```
regressor
```

Out[8]:
```
▾ DecisionTreeRegressor
DecisionTreeRegressor()
```

In [9]:
```
regressor.fit(x_train,y_train)
```

Out[9]:
```
▾ DecisionTreeRegressor
DecisionTreeRegressor()
```

In [10]:
```
y_pred=regressor.predict(x_test)
```

In [11]:
```
y_pred
```

Out[11]:
```
array([3.266   , 5.00001, 1.253   , ..., 3.382   , 5.00001, 2.425   ])
```

In [12]:
```
from sklearn.metrics import r2_score
score=r2_score(y_pred,y_test)
```

In [13]:
```
score
```

Out[13]:
```
0.6383580185781638
```

In [24]:
```
#Hyperparameter Tunning
parameter={
    'criterion':['squared_error','friedman_mse','absolute_error','poisson'],
    'splitter':['best','random'],
    'max_depth':[1,2,3,4,5,6,7,8,9,10,11,12],
    'max_features':['auto','sqrt','log2']
}
regressor=DecisionTreeRegressor()
```

In [25]:
```
from sklearn.model_selection import GridSearchCV
regressorcv=GridSearchCV(regressor,param_grid=parameter,cv=2,scoring='neg_mean_squared_e
```

In [26]:
```
regressorcv.fit(x_train,y_train)
```

```
C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\model_selection\_validation.py:42
5: FitFailedWarning:
192 fits failed out of a total of 576.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='ra
ise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
192 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\model_selection\_validati
on.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\base.py", line 1145, in w
rapper
    estimator._validate_params()
  File "C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\base.py", line 638, in _v
alidate_params
    validate_parameter_constraints(
  File "C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\utils\_param_validation.p
y", line 96, in validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'max_features' parameter of D
ecisionTreeRegressor must be an int in the range [1, inf), a float in the range (0.0, 1.
0], a str among {'sqrt', 'log2'} or None. Got 'auto' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\PRATHMESH\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:979: U
serWarning: One or more of the test scores are non-finite: [        nan         nan -1.2
6756302 -1.26394246 -1.04933426 -1.20926412
         nan         nan -0.90941067 -1.3167412  -0.96336551 -1.13530629
         nan         nan -0.89765204 -1.22266281 -0.8194725  -1.16047002
         nan         nan -0.82936368 -1.11733375 -0.68385504 -1.23381367
         nan         nan -0.7363701  -1.21572025 -0.61947432 -1.00697387
         nan         nan -0.60765938 -0.93612642 -0.57998378 -0.79912332
         nan         nan -0.56383743 -0.96490213 -0.50648974 -0.69423087
         nan         nan -0.5226716  -1.03298827 -0.47912671 -0.62301724
         nan         nan -0.51532394 -0.82695573 -0.50062118 -0.75495578
         nan         nan -0.5253521  -0.82748909 -0.47730626 -0.72398682
         nan         nan -0.53681235 -0.76899792 -0.47982818 -0.61907425
         nan         nan -0.58574715 -0.76460975 -0.54330263 -0.57612351
         nan         nan -1.09848566 -1.32050716 -1.09230812 -1.32506352
         nan         nan -0.84916881 -1.31550982 -0.9733992  -1.09089511
         nan         nan -0.83984538 -1.11901026 -0.73728681 -1.22173646
         nan         nan -0.7112623  -1.06709855 -0.62350863 -1.24038936
         nan         nan -0.69986695 -1.0824697  -0.70425429 -1.08178152
         nan         nan -0.65733587 -0.84758118 -0.50822288 -1.15133402
         nan         nan -0.56332556 -0.93857039 -0.53851877 -0.78103823
         nan         nan -0.57726642 -0.85076399 -0.54419435 -0.91015114
         nan         nan -0.53224976 -0.91237414 -0.48621413 -0.69489859
         nan         nan -0.60668585 -0.74890955 -0.49877568 -0.69936659
         nan         nan -0.52346155 -0.71346275 -0.48675659 -0.68713075
         nan         nan -0.53682675 -0.63966064 -0.50952928 -0.62661578
         nan         nan -1.0834203  -1.40573407 -1.21775928 -1.40832607
         nan         nan -1.04068069 -1.37712915 -1.04123523 -1.04792463
         nan         nan -1.07322736 -1.2930419  -0.86015962 -1.14933021
         nan         nan -0.84261244 -1.32873543 -0.75674981 -1.25529111
         nan         nan -0.77235862 -1.14424401 -0.69981935 -1.01482663
         nan         nan -0.69664376 -0.85340288 -0.55511523 -1.02976664
         nan         nan -0.64551887 -0.7534348  -0.55527349 -1.03128405
         nan         nan -0.49724714 -0.97112672 -0.52791448 -0.77158749
         nan         nan -0.61669348 -0.76981406 -0.47918176 -0.80374554
         nan         nan -0.56045734 -0.70142825 -0.52209857 -0.78899667
         nan         nan -0.55020263 -0.62589042 -0.5140678  -0.62571661
```

```
         nan          nan -0.54545787 -0.67977992 -0.48347056 -0.61019087
         nan          nan -1.10574069 -1.08867572 -1.10729868 -1.1358576
         nan          nan -1.02064393 -1.14348179 -0.97662094 -1.30591281
         nan          nan -0.99126519 -1.23097528 -0.7358691  -1.01807956
         nan          nan -0.84851862 -1.24147169 -0.6902697  -0.91379016
         nan          nan -0.63992601 -1.1280495  -0.56223245 -1.01510721
         nan          nan -0.7131085  -0.89136564 -0.58917554 -0.84099316
         nan          nan -0.62476576 -0.90674482 -0.57322074 -1.03372151
         nan          nan -0.52509904 -0.88353466 -0.5476596  -0.67147941
         nan          nan -0.54545609 -0.75612919 -0.49888268 -0.71018014
         nan          nan -0.54933827 -0.80467659 -0.51155163 -0.73590181
         nan          nan -0.53302507 -0.81934364 -0.50334704 -0.68298238
         nan          nan -0.53394173 -0.61603722 -0.55942379 -0.66299489]
  warnings.warn(
```

Out[26]:

```
▸ GridSearchCV
▸ estimator: DecisionTreeRegressor
    ▸ DecisionTreeRegressor
```

In [29]:

```
regressorcv.best_params_
```

Out[29]:

```
{'criterion': 'squared_error',
 'max_depth': 10,
 'max_features': 'log2',
 'splitter': 'best'}
```

In [30]:

```
y_pred=regressorcv.predict(x_test)
```

In [31]:

```
r2_score(y_pred,y_test)
```

Out[31]:

```
0.5894976592027634
```

In [ ]: