In [ ]:

```
from sklearn.datasets import load_boston
```

In [ ]:

```
boston=load_boston()
```

In [ ]:

```
import pandas as pd
```

In [ ]:

```
data=pd.DataFrame(boston.data,columns=boston.feature_names)
```

In [ ]:

```
data['MEDV']=pd.DataFrame(boston.target)
```

In [ ]:

```
data
```

Out[ ]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1.0	273.0	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1.0	273.0	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1.0	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1.0	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0.0	0.573	6.030	80.8	2.5050	1.0	273.0	21.0	396.90	7.88	11.9

506 rows × 14 columns

In [ ]:

```
pd.DataFrame(data.corr().round(2))
```

Out[ ]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	1.00	-0.20	0.41	-0.06	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39
ZN	-0.20	1.00	-0.53	-0.04	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36
INDUS	0.41	-0.53	1.00	0.06	0.76	-0.39	0.64	-0.71	0.60	0.72	0.38	-0.36	0.60	-0.48
CHAS	-0.06	-0.04	0.06	1.00	0.09	0.09	0.09	-0.10	-0.01	-0.04	-0.12	0.05	-0.05	0.18
NOX	0.42	-0.52	0.76	0.09	1.00	-0.30	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43
RM	-0.22	0.31	-0.39	0.09	-0.30	1.00	-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.70
AGE	0.35	-0.57	0.64	0.09	0.73	-0.24	1.00	-0.75	0.46	0.51	0.26	-0.27	0.60	-0.38
DIS	-0.38	0.66	-0.71	-0.10	-0.77	0.21	-0.75	1.00	-0.49	-0.53	-0.23	0.29	-0.50	0.25
RAD	0.63	-0.31	0.60	-0.01	0.61	-0.21	0.46	-0.49	1.00	0.91	0.46	-0.44	0.49	-0.38
TAX	0.58	-0.31	0.72	-0.04	0.67	-0.29	0.51	-0.53	0.91	1.00	0.46	-0.44	0.54	-0.47
PTRATIO	0.29	-0.39	0.38	-0.12	0.19	-0.36	0.26	-0.23	0.46	0.46	1.00	-0.18	0.37	-0.51
B	-0.39	0.18	-0.36	0.05	-0.38	0.13	-0.27	0.29	-0.44	-0.44	-0.18	1.00	-0.37	0.33
LSTAT	0.46	-0.41	0.60	-0.05	0.59	-0.61	0.60	-0.50	0.49	0.54	0.37	-0.37	1.00	-0.74
MEDV	-0.39	0.36	-0.48	0.18	-0.43	0.70	-0.38	0.25	-0.38	-0.47	-0.51	0.33	-0.74	1.00

In [ ]:

```
x=data[['RM','ZN']]
y=data['MEDV']
```

In [ ]:

```
print(x)
print(y)
```

```
        RM     ZN
0      6.575   18.0
1      6.421    0.0
2      7.185    0.0
3      6.998    0.0
4      7.147    0.0
..       ...    ...
501    6.593    0.0
502    6.120    0.0
503    6.976    0.0
504    6.794    0.0
505    6.030    0.0

[506 rows x 2 columns]
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
```

```
from sklearn.model_selection import train_test_split
```

In [ ]:

In [ ]:

```
x=pd.DataFrame(x) y=pd.DataFrame(y)
```

In [ ]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [ ]:

```
x_train
```

Out[ ]:

	RM	ZN
148	5.186	0.0
353	6.728	90.0
200	7.135	95.0
29	6.674	0.0
276	7.267	40.0
...	...	...
437	6.152	0.0
235	6.086	0.0
265	5.560	20.0
367	3.863	0.0
195	7.875	80.0

404 rows × 2 columns

In [ ]:

```
y_train
```

Out[ ]:

	MEDV
148	17.8
353	30.1
200	32.9
29	21.0
276	33.2
...	...
437	8.7
235	24.0
265	22.8
367	23.1
195	50.0

404 rows × 1 columns

In [ ]:

```
from sklearn.tree import DecisionTreeRegressor
```

In [ ]:

```
dt1=DecisionTreeRegressor(max_depth=20)
```

In [ ]:

```
dt1.fit(x_train,y_train)
```

Out[ ]:

```
DecisionTreeRegressor(max_depth=20)
```

In [ ]:

```
y_pred1=dt1.predict(x_test)
```

In [ ]:

```
import numpy as np
```

In [ ]:

```
from sklearn.metrics import mean_squared_error
```

In [ ]:

```
np.sqrt(mean_squared_error(y_test,y_pred1))
```

Out[ ]:

```
7.154264805132725
```

In [ ]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [ ]:

```
rf1=RandomForestRegressor()
```

In [ ]:

```
rf1.fit(x_train,y_train)
```

```
<ipython-input-24-18d10c9b918e>:1: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  rf1.fit(x_train,y_train) Out[ ]:
RandomForestRegressor()
```

In [ ]:

```
rf1.score(x_test,y_test)
```

Out[ ]:

```
0.3334644045136971
```

In [ ]:

```
y_pred2=rf1.predict(x_test)
```

In [ ]:

```
np.sqrt(mean_squared_error(y_test,y_pred2))
```

Out[ ]:

```
6.972159628039433
```

Decision Tree Classifier

In [ ]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [ ]:

```
from sklearn.datasets import load_digits
```

In [ ]:

```
digits=load_digits()
```

In [ ]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0.25)
```

In [ ]:

```
dt2=DecisionTreeClassifier(criterion="entropy")
```

In [ ]:

```
dt2.fit(x_train,y_train)
```

Out[ ]:

```
DecisionTreeClassifier(criterion='entropy')
```

In [ ]:

```
dt2.score(x_test,y_test)
```

Out[ ]:

```
0.86
```

In [ ]:

```
dt3=DecisionTreeClassifier(max_depth=30)
```

In [ ]:

```
dt3.fit(x_train,y_train)
```

Out[ ]:

```
DecisionTreeClassifier(max_depth=30)
```

In [ ]:

```
dt3.score(x_test,y_test)
```

Out[ ]:

```
0.8511111111111112
```