In [1]:
```
#import numpy as np
#a=np.array([1,2,3])
#print(a)
#print(type(a))
#print(a.ndim)
#print(a.shape)
#print(len(a))
#np.arange(15).reshape(3,5)
```

In [153]:
```
#import numpy as np

#np.linspace(1,8,4)
#np.linspace(1,8,4,endpoint=False)
#np.random.randint(1,100,6)
#np.random.randint(1,100,6).reshape(3,2)
#np.random.rand(4)
#np.eye(5)
#np.zeros(3)
#np.random.randn(3,2)
#np.empty((3,3))
#np.arange(0, 2, 0.3)
#np.ones((2,3,4),dtype=np.int16)
#np.array([[1,2],[3,4]],dtype=complex)
#np.array([(1.5,2,3),(4,5,6)])
```

In [169]:
```
A = np.array([[2,1],[0,1]])
B = np.array([[2,0],[3,4]])
#A+B
#np.add(A,B)
#A * B
#A@B
#A.dot(B)
#B.T
#A.flatten()
#B < 3
#A.sum()
#A.sum(axis=0)
#A.sum(axis=1)
#A.cumsum(axis=1)
#A.min()
#A.max()
```

In [179]:
```
a = np.arange(4)**3
#print(a)
#a[2]
#a[::-1]
#a[0:4:1]
#a[1,...]
#a[a>5]
#x = a[0:4]
#print(x)
#x[:]=99
#print(x)
```

In [2]:
```
#import numpy as np
#import pandas as pd
#arr=np.array([1,3,5,7,9])
#s2=pd.Series(arr)
#print(s2)
#print(type(s2))
```

In [3]:
```
#import pandas as pd
#s2=pd.Series([10,20,30])
#print(s2)
#print(type(s2))
#s3=pd.DataFrame([[1,2],[3,4]],columns=['A','B'],index=['C','D'])
#print(s3)
#print(type(s3))
```

In [4]:
```
#s4=pd.DataFrame(np.random.randn(20,7), columns=['A','B','C','D','E','F','G'])
#print(s4)
#s4[(5<s4.index) & (s4.index<10)]  # s4 with values that satisfy both conditio
ns
#s4.head()                 # First five rows of s4
#s4.tail()                 # Last five rows of s4
#s4.describe()             # statistical information of data
#s4['B']                   # data of the 'B' column
#s4[['B','E']]             # data of the 'B' and 'E' column
#s4[0:3]                   # data of the 1~3 rows
#s4.iloc[0:3]              # data of the 1~3 rows
#s4.loc[[2,3],['A','B']]   # value of row 2,3 column 'A' ,'B'
#s4[2 < s4]                # s4 with values matching conditions
#s4.mean()                 # means of each column
#s4.mean(1)                # mean of each row
#s4.drop(5,axis=0)              # delete row 5
#s4.drop('D',axis=1)       # delete 'D' column
#s4.drop('D',axis=1, inplace=True) # delete 'D' column permanently
#print(s4)
#s4['H']=np.random.rand(20,1)      # add a new column of same length.
#print(s4)
```

In [ ]:
```

```

In [1]:
```
#from sklearn.datasets import load_boston
```

In [2]:
```
#boston = load_boston()
```

In [3]:
```
#type(boston)
```

Out[3]:
```
sklearn.utils.Bunch
```

In [1]:
```
#print(boston.data)
```

In [2]:
```
#print(boston.DESCR)
```

In [3]:
```
#print(boston.feature_names)
```

In [4]:
```
#type(boston.feature_names)
```

In [5]:
```
#print(boston.target)
```

In [6]:
```
#type(boston.target)
```

In [7]:
```
#boston.shape
```

In [9]:
```
#print(boston.data.shape)
```

In [10]:
```
#print(boston.target.shape)
```

In [13]:
```
#import pandas as pd
```

In [14]:
```
#data = pd.DataFrame(boston.data,columns = boston.feature_names)
```

In [8]:
```
#print(data)
```

In [12]:
```
#print(data.head())
```

In [11]:
```
#print(data.head(10))
```

In [13]:
```
#data['MEDV'] = pd.DataFrame(boston.target)
```

In [14]:
```
#print(data.head())
```

In [15]:
```
#pd.DataFrame(data.corr().round(2))
```

In [21]:
```
x = data['RM']
y = data['MEDV']
```

In [16]:
```
#type(x)
```

In [17]:
```
#type(y)
```

In [18]:
```
#pd.DataFrame([x,y]).transpose().head()
```

In [19]:
```
#pd.DataFrame([x,y]).head()
```

In [20]:
```
#pd.DataFrame([x,y]). transpose().head()
```

In [21]:
```
#from sklearn.linear_model import LinearRegression
```

In [22]:
```
#model1 = LinearRegression()
```

In [23]:
```
#x=pd.DataFrame(x)
#y=pd.DataFrame(y
```

In [30]:
```
#x=pd.DataFrame(x)
#y=pd.DataFrame(y)
```

In [24]:
```
#type(x)
```

In [25]:
```
#model1.fit(x,y)
```

In [26]:
```
#x_test=[6.41,6.66,7.5]
```

In [27]:
```
#y_pred = model1.predict(x_test)
```

In [28]:
```
#x_test=pd.DataFrame(x_test)
```

In [29]:
```
#type(x_test)
```

In [30]:
```
#y_pred = model1.predict(x_test)
```

In [31]:
```
#print(y_pred,)
```

In [32]:
```
#print(y_pred)
```

In [ ]:
```

```