Aim: Linear Regression by using Deep Neural Network

In [ ]:
```
pip install tensorflow --user --no-warn-script-location
```

In [1]:
```
import tensorflow as tf
from tensorflow.keras.datasets import boston_housing
from sklearn import preprocessing
```

In [2]:
```
(train_x, train_y),(test_x,test_y)=boston_housing.load_data()
```

In [3]:
```
print(train_x.shape)
print(test_x.shape)
print(train_y.shape)
print(test_y.shape)
```

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

In [4]:
```
train_x[0]
```

Out[4]:
```
array([  1.23247,   0.     ,   8.14   ,   0.     ,   0.538  ,   6.142  ,
        91.7    ,   3.9769 ,   4.     , 307.     ,  21.     , 396.9    ,
        18.72   ])
```

In [5]:
```
train_y[0]
```

Out[5]:
```
15.2
```

In [6]:
```
train_x=preprocessing.normalize(train_x)
test_x=preprocessing.normalize(test_x)
```

In [7]:
```
train_x[0]
```

Out[7]:
```
array([0.0024119 , 0.        , 0.01592969, 0.        , 0.00105285,
       0.01201967, 0.17945359, 0.00778265, 0.00782786, 0.6007879 ,
       0.04109624, 0.77671895, 0.03663436])
```

In [8]:
```
train_y[0]
```

Out[8]:
```
15.2
```

In [9]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import *

def HPPM():
    model=Sequential()
    model.add(Dense(128, activation='relu', input_shape=(train_x[0].shape)))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
    return model
```

In [10]:
```
import numpy as np
k=4
num_val_samples=len(train_x)
num_epochs=100
all_scores=[]
```

In [11]:
```
model=HPPM()
history=model.fit(x=train_x, y=train_y, epochs=num_epochs, batch_size=1, verb
```

```
Epoch 1/100
404/404 [==============================] - 1s 2ms/step - loss: 145.2648 -
mae: 8.8432 - val_loss: 70.4067 - val_mae: 6.0675
Epoch 2/100
404/404 [==============================] - 2s 4ms/step - loss: 69.6452 -
mae: 5.8083 - val_loss: 60.2890 - val_mae: 5.7936
Epoch 3/100
404/404 [==============================] - 2s 4ms/step - loss: 64.0329 -
mae: 5.4785 - val_loss: 58.2462 - val_mae: 5.5318
Epoch 4/100
404/404 [==============================] - 2s 4ms/step - loss: 62.4070 -
mae: 5.4575 - val_loss: 55.6510 - val_mae: 5.6276
Epoch 5/100
404/404 [==============================] - 1s 4ms/step - loss: 57.9584 -
mae: 5.3383 - val_loss: 54.0462 - val_mae: 5.4752
Epoch 6/100
404/404 [==============================] - 1s 3ms/step - loss: 58.4756 -
mae: 5.2441 - val_loss: 61.6325 - val_mae: 5.5175
Epoch 7/100
```

In [12]:
```
test_input=[(8.65407330e-05, 0.00000000e+00, 1.13392175e-02, 0.00000000e+00,
print("Actual output : 15.2")
print("Predicted output: ", model.predict(test_input))
```

```
Actual output : 15.2
1/1 [==============================] - 0s 105ms/step
Predicted output:  [[15.099294]]
```