In [1]:
```python
#This is a supplementary material to the lecture "Introduction to ML" to quickly revise, whenever need
ed
```

In [2]:
```python
#In general computing, we need to pass the data and write the rules (program) to figure out the output
#Machine learning in broad terms is about using some past data and corresponding output, let machine f
igure out the rules and then
#using thode rules, predict the output for unseen data (supervised machine learning)

#There are some use cases, like categorising users on some specific traits, we just need to give data
 to machine and
#let it figure out the patterns in user traits and categorising them
```

In [3]:
```python
#In supervised learning, we may have two types of problems to solve
#one is Classification (where the output to be predicted is not continuous i.e. labels)
#another is Regression (where the output to be predicted is continuous spectrum of values)
```

In [4]:
```python
#Now, for solving any machine learning task, there are some general steps
# 1. Get the data
# 2. Preprocess/clean it to make it compatible with input, output formats
# 3. Train the model
# 4. Predict output for unseen data
```

In [5]:
```python
#scikit-learn (sklearn) is the machine leraning library, we will use heavily in this course
```

In [6]:
```python
#ok, enough of theory, let's the game begin!!
```

In [7]:
```python
#let's first import required packages/libraries to read, manipulate the data and train the model
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split        #for splitting our data to training and te
sting
from sklearn.linear_model import LinearRegression           #to fit the linear model to the data
```

In [8]:
```python
#let's work on simple boston house-prices dataset
from sklearn import datasets
data = datasets.load_boston()
```

In [9]:

```
#let's look at the data we have loaded
data
```

Out[9]:
```
{'data': array([[6.3200e-03, 1.8000e+01, 2.3100e+00, ..., 1.5300e+01, 3.9690e+02,
         4.9800e+00],
        [2.7310e-02, 0.0000e+00, 7.0700e+00, ..., 1.7800e+01, 3.9690e+02,
         9.1400e+00],
        [2.7290e-02, 0.0000e+00, 7.0700e+00, ..., 1.7800e+01, 3.9283e+02,
         4.0300e+00],
        ...,
        [6.0760e-02, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9690e+02,
         5.6400e+00],
        [1.0959e-01, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9345e+02,
         6.4800e+00],
        [4.7410e-02, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9690e+02,
         7.8800e+00]]),
 'target': array([24. , 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9, 15. ,
        18.9, 21.7, 20.4, 18.2, 19.9, 23.1, 17.5, 20.2, 18.2, 13.6, 19.6,
        15.2, 14.5, 15.6, 13.9, 16.6, 14.8, 18.4, 21. , 12.7, 14.5, 13.2,
        13.1, 13.5, 18.9, 20. , 21. , 24.7, 30.8, 34.9, 26.6, 25.3, 24.7,
        21.2, 19.3, 20. , 16.6, 14.4, 19.4, 19.7, 20.5, 25. , 23.4, 18.9,
        35.4, 24.7, 31.6, 23.3, 19.6, 18.7, 16. , 22.2, 25. , 33. , 23.5,
        19.4, 22. , 17.4, 20.9, 24.2, 21.7, 22.8, 23.4, 24.1, 21.4, 20. ,
        20.8, 21.2, 20.3, 28. , 23.9, 24.8, 22.9, 23.9, 26.6, 22.5, 22.2,
        23.6, 28.7, 22.6, 22. , 22.9, 25. , 20.6, 28.4, 21.4, 38.7, 43.8,
        33.2, 27.5, 26.5, 18.6, 19.3, 20.1, 19.5, 19.5, 20.4, 19.8, 19.4,
        21.7, 22.8, 18.8, 18.7, 18.5, 18.3, 21.2, 19.2, 20.4, 19.3, 22. ,
        20.3, 20.5, 17.3, 18.8, 21.4, 15.7, 16.2, 18. , 14.3, 19.2, 19.6,
        23. , 18.4, 15.6, 18.1, 17.4, 17.1, 13.3, 17.8, 14. , 14.4, 13.4,
        15.6, 11.8, 13.8, 15.6, 14.6, 17.8, 15.4, 21.5, 19.6, 15.3, 19.4,
        17. , 15.6, 13.1, 41.3, 24.3, 23.3, 27. , 50. , 50. , 50. , 22.7,
        25. , 50. , 23.8, 23.8, 22.3, 17.4, 19.1, 23.1, 23.6, 22.6, 29.4,
        23.2, 24.6, 29.9, 37.2, 39.8, 36.2, 37.9, 32.5, 26.4, 29.6, 50. ,
        32. , 29.8, 34.9, 37. , 30.5, 36.4, 31.1, 29.1, 50. , 33.3, 30.3,
        34.6, 34.9, 32.9, 24.1, 42.3, 48.5, 50. , 22.6, 24.4, 22.5, 24.4,
        20. , 21.7, 19.3, 22.4, 28.1, 23.7, 25. , 23.3, 28.7, 21.5, 23. ,
        26.7, 21.7, 27.5, 30.1, 44.8, 50. , 37.6, 31.6, 46.7, 31.5, 24.3,
        31.7, 41.7, 48.3, 29. , 24. , 25.1, 31.5, 23.7, 23.3, 22. , 20.1,
        22.2, 23.7, 17.6, 18.5, 24.3, 20.5, 24.5, 26.2, 24.4, 24.8, 29.6,
        42.8, 21.9, 20.9, 44. , 50. , 36. , 30.1, 33.8, 43.1, 48.8, 31. ,
        36.5, 22.8, 30.7, 50. , 43.5, 20.7, 21.1, 25.2, 24.4, 35.2, 32.4,
        32. , 33.2, 33.1, 29.1, 35.1, 45.4, 35.4, 46. , 50. , 32.2, 22. ,
        20.1, 23.2, 22.3, 24.8, 28.5, 37.3, 27.9, 23.9, 21.7, 28.6, 27.1,
        20.3, 22.5, 29. , 24.8, 22. , 26.4, 33.1, 36.1, 28.4, 33.4, 28.2,
        22.8, 20.3, 16.1, 22.1, 19.4, 21.6, 23.8, 16.2, 17.8, 19.8, 23.1,
        21. , 23.8, 23.1, 20.4, 18.5, 25. , 24.6, 23. , 22.2, 19.3, 22.6,
        19.8, 17.1, 19.4, 22.2, 20.7, 21.1, 19.5, 18.5, 20.6, 19. , 18.7,
        32.7, 16.5, 23.9, 31.2, 17.5, 17.2, 23.1, 24.5, 26.6, 22.9, 24.1,
        18.6, 30.1, 18.2, 20.6, 17.8, 21.7, 22.7, 22.6, 25. , 19.9, 20.8,
        16.8, 21.9, 27.5, 21.9, 23.1, 50. , 50. , 50. , 50. , 50. , 13.8,
        13.8, 15. , 13.9, 13.3, 13.1, 10.2, 10.4, 10.9, 11.3, 12.3,  8.8,
         7.2, 10.5,  7.4, 10.2, 11.5, 15.1, 23.2,  9.7, 13.8, 12.7, 13.1,
        12.5,  8.5,  5. ,  6.3,  5.6,  7.2, 12.1,  8.3,  8.5,  5. , 11.9,
        27.9, 17.2, 27.5, 15. , 17.2, 17.9, 16.3,  7. ,  7.2,  7.5, 10.4,
         8.8,  8.4, 16.7, 14.2, 20.8, 13.4, 11.7,  8.3, 10.2, 10.9, 11. ,
         9.5, 14.5, 14.1, 16.1, 14.3, 11.7, 13.4,  9.6,  8.7,  8.4, 12.8,
        10.5, 17.1, 18.4, 15.4, 10.8, 11.8, 14.9, 12.6, 14.1, 13. , 13.4,
        15.2, 16.1, 17.8, 14.9, 14.1, 12.7, 13.5, 14.9, 20. , 16.4, 17.7,
        19.5, 20.2, 21.4, 19.9, 19. , 19.1, 19.1, 20.1, 19.9, 19.6, 23.2,
        29.8, 13.8, 13.3, 16.7, 12. , 14.6, 21.4, 23. , 23.7, 25. , 21.8,
        20.6, 21.2, 19.1, 20.6, 15.2,  7. ,  8.1, 13.6, 20.1, 21.8, 24.5,
        23.1, 19.7, 18.3, 21.2, 17.5, 16.8, 22.4, 20.6, 23.9, 22. , 11.9]),
 'feature_names': array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',
        'TAX', 'PTRATIO', 'B', 'LSTAT'], dtype='<U7'),
 'DESCR': ".. _boston_dataset:\n\nBoston house prices dataset\n---------------------------\n\n**Data
Set Characteristics:**  \n\n    :Number of Instances: 506 \n\n    :Number of Attributes: 13 numeric/c
ategorical predictive. Median Value (attribute 14) is usually the target.\n\n    :Attribute Informati
on (in order):\n        - CRIM     per capita crime rate by town\n        - ZN       proportion of re
sidential land zoned for lots over 25,000 sq.ft.\n        - INDUS    proportion of non-retail busines
s acres per town\n        - CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 other
wise)\n        - NOX      nitric oxides concentration (parts per 10 million)\n        - RM       aver
age number of rooms per dwelling\n        - AGE      proportion of owner-occupied units built prior t
o 1940\n        - DIS      weighted distances to five Boston employment centres\n        - RAD      i
ndex of accessibility to radial highways\n        - TAX      full-value property-tax rate per $10,000
```

```
\n        - PTRATIO  pupil-teacher ratio by town\n        - B        1000(Bk - 0.63)^2 where Bk is th
e proportion of blacks by town\n        - LSTAT    % lower status of the population\n        - MEDV
Median value of owner-occupied homes in $1000's\n\n    :Missing Attribute Values: None\n\n    :Creato
r: Harrison, D. and Rubinfeld, D.L.\n\nThis is a copy of UCI ML housing dataset.\nhttps://archive.ic
s.uci.edu/ml/machine-learning-databases/housing/\n\n\nThis dataset was taken from the StatLib library
which is maintained at Carnegie Mellon University.\n\nThe Boston house-price data of Harrison, D. and
Rubinfeld, D.L. 'Hedonic\nprices and the demand for clean air', J. Environ. Economics & Management,\n
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch, 'Regression diagnostics\n...', Wiley, 1980.
N.B. Various transformations are used in the table on\npages 244-261 of the latter.\n\nThe Boston hou
se-price data has been used in many machine learning papers that address regression\nproblems.   \n
\n.. topic:: References\n\n   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influenti
al Data and Sources of Collinearity', Wiley, 1980. 244-261.\n   - Quinlan,R. (1993). Combining Instan
ce-Based and Model-Based Learning. In Proceedings on the Tenth International Conference of Machine Le
arning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.\n",
 'filename': 'C:\\Users\\NKS\\Anaconda3\\lib\\site-packages\\sklearn\\datasets\\data\\boston_house_pr
ices.csv'}
```

In [10]:
```
#we can see that it's a dictionary containing the data key (features) and a key target (output)
x, y = data['data'], data['target']
```

In [11]:
```
x.shape, y.shape
```

Out[11]:
```
((506, 13), (506,))
```

In [12]:
```
#we have loaded the data and for this data, it's already cleaned, let's just go to splitting the data
 in training and testing part
x_train, x_test, y_train, y_test = train_test_split(x, y)
```

In [13]:
```
#create an object for linear regression model
alg = LinearRegression()
#let's train it on training data
alg.fit(x_train, y_train)
```

Out[13]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [14]:
```
#let's predict output for the test data
y_pred = alg.predict(x_test)
y_pred
```

Out[14]:
```
array([23.92063152, 21.70960302, 15.90137371, 25.26613301, 35.28140728,
       33.52524584, 31.5096118 , 24.7278648 , 29.78030644, 23.47682464,
       18.67509642, 16.09391054, 18.99758634, 31.0204188 , 24.4544231 ,
       33.14582508, 21.94666409, 21.50488414, 41.6312715 , 25.59723236,
       19.03071767, 15.74215299, 19.97454463,  8.22678384, 20.74759963,
       20.69842196, 23.62803176, 27.92068888, 16.77707254, 25.03078489,
       23.44771888, 32.09655476, 19.89498947, 22.31265674, 29.5992013 ,
       20.73430944, 18.21921299, 14.41814784, 33.49209035, 32.82080152,
       34.21850912, 21.03199176, 41.57949748, 31.34313733, 16.38879188,
       36.77031586, 19.86647289, 22.39533759, 20.76926633, 20.26947855,
       23.40805566, 15.02586201,  8.80571267, 30.6199747 , 12.99512848,
       23.17095429, 19.90261325, 19.67699176, 20.61715063, 21.57080798,
       40.44213951, 20.77831271, 36.60525538, 33.12336125, 28.11597573,
       19.12842547, 21.95870278, 24.69522108, 21.64426877, 35.28076106,
       23.21838962, 21.3121438 , 32.52315044, 18.32394463, 24.89186364,
       14.86180983, 12.17746263, 18.18513725, 13.05267758, 28.18491413,
       17.53579646, 23.8675469 , 35.13050216, 19.91786559, 36.54741044,
       18.54715893, 13.89860166, 11.35470543, 21.1374334 , 17.19694955,
       11.76699982, 34.04578352, 22.88080089, 32.17612864, 13.76864937,
       12.12140298, 18.56446052, 21.17790522, 27.44023073, 15.11573552,
       25.51723534, 25.83393721, 18.67099273, 14.4901367 , 11.86075032,
       36.98801211, 18.62953052, 21.58013076, 19.31129993, 33.06459861,
       23.32722355, 23.77959932, 25.93054169, 24.86666654, 17.93431748,
       15.9819414 ,  8.27879767, 31.62097212, 13.49420127, 28.91437617,
       32.55343849, 18.9068841 , 25.75440277, 20.84163537, 17.334071  ,
       33.49740441, 36.90388786])
```

In [15]:
```
#Thanks, happy Coding!
```

In [ ]:

```
#To download .ipynb notebook, right click the following link and click save as
https://ninjasfiles.s3.amazonaws.com/0000000000003220.ipynb
```