# 05-02-22 -Linear Regression

Saturday, February 5, 2022 2:03 PM

## Simple

$$y = bx + a$$

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \; ; \; a = \bar{y} - b\bar{x}$$

Code:

```
x = [1, 2, 3, 4, 5]
y = [12, 15, 13, 14, 16]
n = len(x)
sumx = 0
sumy = 0
for i in x:
    sumx += i
for i in y:
    sumy += i
meanx = sumx/n
meany = sumy/n
numer = 0
denomer = 0
for i in range(n):
    numer += (x[i] - meanx)*(y[i] - meany)
    denomer += (x[i] - meanx)**2
b = numer/denomer
```

```
a = meany - b*meanx
SS=0
print (b, a)
for i in range (n):
    predy = b*x[i] + a
    print (y, predy)
    SS+ = (y[i] - predy)**2
print(ss)
```

Multiple

$$\beta = (X^T X)^{-1} X^T y$$

$$\hat{y} = X\beta = X (X^T X)^{-1} X^T y$$

Code: import numpy as np
from numpy.linalg import inv.

```
a = [   ]
b = [   ]
y = [   ]
X = []
for i in range (len(a)):
    X.append ([])
    X[i].append(1)        # B0 + B1 x1 + B2 x2 ...
    X[i].append(a[i])
    X[i].append(b[i])
print (X)
matrix = np.matrix (X)
```

```
print(x)
matrx = np.array(X)
print(matrx)
temp = inv((matrx.T).dot(matrx))
temp2 = temp.dot(matrx.T)
print(temp2)
beta = temp2.dot(np.array(y))
print(beta)
print(matrx.dot(beta))
```

Using Scikit learn : { Same for Simple & multiple }

```
from pandas import DataFrame
from sklearn import linear_model
d2 = {
    } # dictionary data
df = DataFrame(d2, columns=['y', 'm', 'i', 'u', 'output'])
X = df[['i', 'u']]   # Like previous example in class
                     # pass a 2D list only
y = df['output']
regr = linear_model.LinearRegression()  # regression object
                                        → does
#fit() # predict()
regr.fit(X, y) → regr.fit(X.values, y)
print(regr.intercept_)   # a
print(regr.coef_)        # b
print(regr.predict([[2.75], [7.3]]))  # predict for   ← 2D list
```

```
print(regr.coef_)
print(regr.predict([[2.75,3]]))  # predict for one random eg:
```

for simple & multiple : both → 2D list