**ASSIGNMENT NO 8**

**Q1. Simple LR**

import pandas as pd

import numpy as np

from sklearn.metrics import r2\_score,mean\_squared\_error

train = pd.read\_csv('train\_dataset1.csv')

test = pd.read\_csv('test\_dataset1.csv')

train\_x = train.iloc[:,1]

train\_y = train.iloc[:,-1]

test\_x = test.iloc[:,1]

test\_y = test.iloc[:,-1]

a = []

def MSE(y\_actual,y\_predict) :

    mse = sum([i\*i for i in list(np.array(y\_actual)-np.array(y\_predict))])/len(y\_actual)

    return mse

def R2\_Score(y\_actual,y\_predict) :

    ssr = 0

    sst = 0

    y\_mean = np.mean(y\_predict)

    for i in range(len(y\_actual)) :

        ssr = ssr + (y\_actual[i]-y\_predict[i])\*\*2

        sst = sst + (y\_actual[i]-y\_mean)\*\*2

    r2 = 1-(ssr/sst)

    return r2

def train\_data() :

    xy = 0

    for i in range(len(train)) :

        xy = xy + train\_x[i]\*train\_y[i]

    xy\_bar = xy/len(train)

    x\_bar = np.mean(train\_x)

    y\_bar = np.mean(train\_y)

    x\_bar\_y\_bar = x\_bar\*y\_bar

    x\_square\_bar = sum(i\*i for i in train\_x)/len(train)

    x\_bar\_square = x\_bar\*\*2

    a1 = (xy\_bar-x\_bar\_y\_bar)/(x\_square\_bar-x\_bar\_square)

    a0 = train\_y[i] - a1\*train\_x[i]

    a.append(a0)

    a.append(a1)

    print("Equaltion of line is y = {} + {}x".format(a0,a1))

def test\_data() :

    predicted\_test\_y = []

    for x in test\_x :

        y = a[0] + a[1]\*x

        predicted\_test\_y.append(y)

    print(test\_y)

    print(predicted\_test\_y)

    print("Mean square error (MSE) Score = ",mean\_squared\_error(test\_y, predicted\_test\_y))

    print("Mean square error (MSE) Score = ",MSE(test\_y, predicted\_test\_y))

    print("R2 Score = ",r2\_score(test\_y, predicted\_test\_y))

    print("R2 Score = ",R2\_Score(test\_y, predicted\_test\_y))

train\_data()

test\_data()

**Q2. Multilinear Regression**

import pandas as pd

import numpy as np

from sklearn.metrics import r2\_score,mean\_squared\_error

train = pd.read\_csv('train\_dataset2.csv')

test = pd.read\_csv('test\_dataset2.csv')

train\_x = train.iloc[:,:-1]

train\_y = train.iloc[:,-1]

test\_x = test.iloc[:,:-1]

test\_y = test.iloc[:,-1]

a = []

def MSE(y\_actual,y\_predict) :

    mse = sum([i\*i for i in list(np.array(y\_actual)-np.array(y\_predict))])/len(y\_actual)

    return mse

def R2\_Score(y\_actual,y\_predict) :

    ssr = 0

    sst = 0

    y\_mean = np.mean(y\_predict)

    for i in range(len(y\_actual)) :

        ssr = ssr + (y\_actual[i]-y\_predict[i])\*\*2

        sst = sst + (y\_actual[i]-y\_mean)\*\*2

    r2 = 1-(ssr/sst)

    return r2

def train\_data() :

    # print(train\_x)

    # print(train\_y)

    global a

    n = ((np.array(train\_x)).shape)[0]

    print(n)

    ones = np.ones([n,1],dtype=int)

    x = np.hstack((ones,(np.array(train\_x))))

    # print(x)

    x\_transpose = np.transpose(x)

    x\_x\_transpose = np.dot(x\_transpose, x)

    x\_x\_transpose\_inverse = np.linalg.inv(x\_x\_transpose)

    transpose = np.dot(x\_x\_transpose\_inverse,x\_transpose)

    a = np.array(np.dot(transpose,train\_y))

    # print(a)

def test\_data() :

    # print(test\_x)

    # print(test\_y)

    print(a)

    y\_predicted = []

    for i in range(len(test\_x)) :

        x = np.array(test\_x.iloc[i,:])

        y = a[0]

        for j in range(len(x)) :

            y = y + a[j+1]\*x[j]

        y\_predicted.append(y)

    print(test\_y)

    print(y\_predicted)

    print("Mean square error (MSE) Score = ",mean\_squared\_error(test\_y, y\_predicted))

    print("Mean square error (MSE) Score = ",MSE(test\_y, y\_predicted))

    print("R2 Score = ",r2\_score(test\_y, y\_predicted))

    print("R2 Score = ",R2\_Score(test\_y, y\_predicted))

    pass

train\_data()

test\_data()

**Q3. Polynomial Regression**

import pandas as pd

import numpy as np

from sklearn.metrics import r2\_score,mean\_squared\_error

train = pd.read\_csv('train\_dataset3.csv')

test = pd.read\_csv('test\_dataset3.csv')

train\_x = train.iloc[:,:-1]

train\_y = train.iloc[:,-1]

test\_x = test.iloc[:,:-1]

test\_y = test.iloc[:,-1]

a = []

def MSE(y\_actual,y\_predict) :

    mse = sum([i\*i for i in list(np.array(y\_actual)-np.array(y\_predict))])/len(y\_actual)

    return mse

def R2\_Score(y\_actual,y\_predict) :

    ssr = 0

    sst = 0

    y\_mean = np.mean(y\_predict)

    for i in range(len(y\_actual)) :

        ssr = ssr + (y\_actual[i]-y\_predict[i])\*\*2

        sst = sst + (y\_actual[i]-y\_mean)\*\*2

    r2 = 1-(ssr/sst)

    return r2

def train\_data() :

    # print(train\_x)

    # print(train\_y)

    global a

    n = 3

    x = []

    for i in range(n) :

        xi = []

        for j in range(n) :

            xij = ([l\*\*(i+j) for l in np.array(train\_x)])

            s = sum(k for k in xij)

            xi.append(s[0])

        x.append(xi)

    y = []

    for i in range(n) :

        xi = [k\*\*i for k in np.array(train\_x)]

        # print(xi)

        xiy = []

        for j in range(len(xi)) :

            xiy.append(xi[j]\*train\_y[j])

        # print(xiy)

        y.append(sum(xiy)[0])

    print(x)

    print(y)

    x = np.array(x)

    print(np.shape(y))

    y = np.resize(y, (3,1))

    print(np.shape(x))

    print(np.shape(y))

    x\_inverse = np.linalg.inv(x)

    ans = np.dot(x\_inverse,y)

    a.append(ans[0][0])

    a.append(ans[1][0])

    a.append(ans[2][0])

    print(a)

def test\_data() :

    y\_predicted = []

    for i in range(len(test\_x)) :

        x = np.array(test\_x.iloc[i,:])[0]

        # print(x)

        y = a[0] + a[1]\*x + a[2]\*x\*x

        y\_predicted.append(y)

    print(test\_y)

    print(y\_predicted)

    print("Mean square error (MSE) Score = ",mean\_squared\_error(test\_y, y\_predicted))

    print("Mean square error (MSE) Score = ",MSE(test\_y, y\_predicted))

    print("R2 Score = ",r2\_score(test\_y, y\_predicted))

    print("R2 Score = ",R2\_Score(test\_y, y\_predicted))

    pass

train\_data()

test\_data()

**Q4. Logistic LR**

import pandas as pd

import numpy as np

from sklearn.metrics import r2\_score,mean\_squared\_error,accuracy\_score

import math

train = pd.read\_csv('train\_dataset4.csv')

test = pd.read\_csv('test\_dataset4.csv')

train\_x = train.iloc[:,0]

train\_y = train.iloc[:,-1]

test\_x = test.iloc[:,0]

test\_y = test.iloc[:,-1]

a = []

def sigmoid(x):

    return 1./(1.+np.exp(-x))

def MSE(y\_actual,y\_predict) :

    mse = sum([i\*i for i in list(np.array(y\_actual)-np.array(y\_predict))])/len(y\_actual)

    return mse

def R2\_Score(y\_actual,y\_predict) :

    ssr = 0

    sst = 0

    y\_mean = np.mean(y\_predict)

    for i in range(len(y\_actual)) :

        ssr = ssr + (y\_actual[i]-y\_predict[i])\*\*2

        sst = sst + (y\_actual[i]-y\_mean)\*\*2

    r2 = 1-(ssr/sst)

    return r2

def train\_data() :

    xy = 0

    for i in range(len(train)) :

        xy = xy + train\_x[i]\*train\_y[i]

    xy\_bar = xy/len(train)

    x\_bar = np.mean(train\_x)

    y\_bar = np.mean(train\_y)

    x\_bar\_y\_bar = x\_bar\*y\_bar

    x\_square\_bar = sum(i\*i for i in train\_x)/len(train)

    x\_bar\_square = x\_bar\*\*2

    a1 = (xy\_bar-x\_bar\_y\_bar)/(x\_square\_bar-x\_bar\_square)

    a0 = train\_y[i] - a1\*train\_x[i]

    a.append(a0)

    a.append(a1)

    print("Equaltion of line is y = {} + {}x".format(a0,a1))

def test\_data() :

    y\_predict = []

    for i in range(len(test\_x)) :

        x = np.array(test\_x)[i]

        # print(x)

        y = a[0]+a[1]\*x

        print(test\_y[i],y)

        p = sigmoid(y)

        print(p)

        if p < 0.6 :

            y\_predict.append(0)

        else  :

            y\_predict.append(1)

    print(test\_y)

    print(y\_predict)

    print("Mean square error (MSE) Score = ",mean\_squared\_error(test\_y, y\_predict))

    print("Mean square error (MSE) Score = ",MSE(test\_y, y\_predict))

    print("R2 Score = ",r2\_score(test\_y, y\_predict))

    print("R2 Score = ",R2\_Score(test\_y, y\_predict))

    print("Acccuracy Score = ",accuracy\_score(test\_y, y\_predict))

train\_data()

test\_data()