

**Introduction to Data Analysis**

**(DATA 1200)**

**Assignment - 2**

**Submitted by: Kartik Sojitra**

**(100723768)**

**To Professor: Sam Plati**

1. **Using Python develop the Neural Network Algorithm script.**

**Step 1: Load libraries**

*#Load Libraries*

**import** **numpy** **as** **np**

**import** **pandas** **as** **pd**

**import** **matplotlib.pyplot** **as** **plt**

%**matplotlib** inline

**Step 2: Load Data**

*#Load Dataset*

data=pd.read\_csv('./Concrete\_Data1.csv')

data.head()

**Step 3: Show key statistics**

*#Show Key Statistics*

data.describe()

**Step 4: Define X and Y variable**

*#Define x and y variable*

x = data.drop('CMS',axis=1).values

y = data['CMS'].values

**Step 5: Load Library for Training Dataset**

*#Load Library for Training Dataset*

**from** **sklearn.model\_selection** **import** train\_test\_split x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=100)

**Step 6: Scale the Data**

*#Scale the Data*

**from** **sklearn.preprocessing** **import** StandardScaler

sc = StandardScaler()

x\_train2 = sc.fit\_transform(x\_train)

x\_test2 = sc.fit\_transform(x\_test)

**Step 7: Script for Linear Regression**

*#Script for Linear Regression*

**from** **sklearn.linear\_model** **import** LinearRegression

**from** **sklearn** **import** metrics

**for** name,method **in** [('Linear Regression', LinearRegression())]: method.fit(x\_train2,y\_train)

predict = method.predict(x\_test2) print('Method: **{}**'.format(name))

*#Coefficents*

print('**\n**Intercept: **{:0.2f}**'.format(float(method.intercept\_))) coeff\_table=pd.DataFrame(np.transpose(method.coef\_),data.drop('CMS',axis=1).columns,columns=['Coefficients'])

print('**\n**')

print(coeff\_table)

*#MAE,MSE and RMSE*

print('**\n**R2: **{:0.2f}**'.format(metrics.r2\_score(y\_test, predict))) print('Mean Absolute Error: **{:0.2f}**'.format(metrics.mean\_absolute\_error(y\_test, predict)))

print('Mean Squared Error: **{:0.2f}**'.format(metrics.mean\_squared\_error(y\_test, predict)))

print('Root Mean Squared Error: **{:0.2f}**'.format(np.sqrt(metrics.mean\_squared\_error(y\_test, predict))))

**Step 8: Forecast Table**

*#Forecast Table*

predict2 = predict.T

diff = predict2-y\_test FcstTble=pd.DataFrame({'Actual':y\_test,'Predicted':predict2.round(1),'Difference':diff.round(1)})

print('**\n**Forecast Table')

FcstTble.head()

(Plati, MLR-Tutorial, 2019)

1. **Summery of key findings:**

**CMS = 35.95 + 13.02\*(Cement) + 8.95\*(Blast) + 5.95\*(Fly Ash) – 2.84\*(Water) + 1.73(Superplasticizer) + 1.59\*(CA) + 2.03\*(FA) + 7.21\*(Age)**

* In this model, each number is treated as a weight and the sign (+ or -) is the average effect on CMS.
* here, for certain factors, the gap between the mean and the standard deviation is less, for example, blast, fly ash, superplasticizer, and age which implies that their information is less spreader.
* On the opposite side, for the variable such as cement, water, CA, FA is extremely less than implies that their information is profoundly spreader.
* For those factors who’s mean and, the standard deviation is high they influence the CMS to affect most as compares to a variable whose gap is little.
* The coefficient of cement is 13.027 states that the increase in the quantity of cement content should increase the CMS value on the other hand coefficient of water is -2.84 which means it water reduced the concrete compressive strength if the water content of the concrete is increased.
* For those factors who’s mean and, the standard deviation is high they influence the CMS to affect most as compares to a variable whose gap is little.

(Plati, RFM and Linear Regression, 2019)

1. **Conclusion and Next step:**

The value of root mean squared error is 10.66, which is more than 10% of the mean value of the percentages of CMS (i.e. 3.50) which implies our algorithm was not exceptionally exact but can still make a sensibly great prediction.

         This model clarifies the division of the fluctuation between the quantities anticipated by the model and the value as opposed to the mean of the actual. This value is between 0 and 1. This model value is 62% which means the model can explain more than 62% of the variation. The linear regression model suggests that the connection between CMS and the independent variable can be expressed in a straight line.

To improve the algorithm, we can add one more independent variable aggregate which highly affects concrete compressive strength and use low water to cement ratio. Aggregate has a high coefficient value as compare to other independent variables such as blast, fly ash, superplasticizer, and age which can help to improve CMS value.

(Plati, RFM and Linear Regression, 2019)

# References:

Plati, S. (2019). MLR-Tutorial.

Plati, S. (2019). RFM and Linear Regression.