

```
In [44]: import pandas as pd
import numpy as np
```


```
In [262... #concrete data set
# df=pd.read_csv(r"D:\16_MACHINE_LEARNING\02_MACHINE_LEARNING_USING_PYTHON\concrete
df=pd.read_csv(r"D:\fake\concrete_data_100k.csv")

df.head()
X=df.iloc[:,0:8]
y=df.iloc[:, -1]
print("Ddimension of Dataset",df.shape)
df.head()
```

Ddimension of Dataset (100000, 9)

Out[262...

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	Strength
0	314.0	0.0	113.0	170.0	10.0	925.0	783.0	28	38.46
1	475.0	118.8	0.0	181.1	8.9	852.1	781.5	28	68.30
2	190.3	0.0	125.2	166.6	9.9	1079.0	798.9	100	33.56
3	246.8	0.0	125.1	143.3	12.0	1086.8	800.9	14	42.22
4	286.3	200.9	0.0	144.7	11.2	1004.6	803.7	3	24.40




```
In [230... #synthetic dataset
df=pd.read_csv(r"D:\fake\Multiple_Linear_Regression_Dataset.csv")
X=df.iloc[:,0:10]
y=df.iloc[:, -1]
print("Ddimension of Dataset",df.shape)
df.head()
```

Ddimension of Dataset (10000, 11)

Out[230...

	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5	Feature_6	Feature_7	Feature_8
0	37.454012	95.071431	73.199394	59.865848	15.601864	15.599452	5.808361	86.617615
1	2.058449	96.990985	83.244264	21.233911	18.182497	18.340451	30.424224	52.475643
2	61.185289	13.949386	29.214465	36.636184	45.606998	78.517596	19.967378	51.423444
3	60.754485	17.052412	6.505159	94.888554	96.563203	80.839735	30.461377	9.767211
4	12.203823	49.517691	3.438852	90.932040	25.877998	66.252228	31.171108	52.006802



```
In [232... #student Performance Dataset
df=pd.read_csv(r"D:\fake\Student_Performance_1M.csv")
X=df.iloc[:,0:4]
```

```
y=df.iloc[:, -1]
print("Ddimension of Dataset",df.shape)
df.head()
```

Ddimension of Dataset (1000000, 5)

Out[232...]

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	9	62	9	6	65
1	6	97	8	7	92
2	3	65	4	9	48
3	2	56	4	5	28
4	2	42	7	8	21

```
In [45]: df=pd.read_csv(r"D:\16_MACHINE_LEARNING\02_MACHINE_LEARNING_USING_PYTHON\concrete_d
X=df.iloc[:,0:8]
y=df.iloc[:, -1]
print("Ddimension of Dataset",df.shape)
df.head()
```

Ddimension of Dataset (1030, 9)

Out[45]:

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	Strength
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30

```
In [46]: from sklearn.model_selection import train_test_split
```

```
In [47]: X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=2,test_size=0.2)
```

USING SCIKIT LEARN

```
In [38]: from sklearn.linear_model import LinearRegression
```

```
In [39]: lr=LinearRegression()
```

```
In [40]: lr.fit(X_train,y_train)
```

```
Out[40]: ▼ LinearRegression ⓘ ?  
LinearRegression()
```

```
In [41]: y_pred=lr.predict(X_test)
```

```
In [42]: from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error,root_m  
print("R2 SCORE :",r2_score(y_test,y_pred))  
print("MEAN ABSOLUTE ERROR :",mean_absolute_error(y_test,y_pred))  
print("MEAN SQUARED ERROR :",mean_squared_error(y_test,y_pred))  
print("ROOT MEAN SQUARED ERROR :",root_mean_squared_error(y_test,y_pred))
```

```
R2 SCORE : 0.570114265275778  
MEAN ABSOLUTE ERROR : 8.226419967037103  
MEAN SQUARED ERROR : 105.76432225737778  
ROOT MEAN SQUARED ERROR : 10.284178249008415
```

```
In [10]: lr.coef_
```

```
Out[10]: array([ 0.11975878,  0.10084644,  0.08639338, -0.1742025 ,  0.26416309,  
                0.01266003,  0.01466698,  0.10931811])
```

```
In [11]: lr.intercept_
```

```
Out[11]: np.float64(-8.724323713693096)
```

```
In [ ]: lr.predict(X_test.loc[[788110]]) # Double brackets to keep it as a DataFrame
```

```
In [ ]:
```

CUSTOM MULTIPLE LR CLASS

```
In [48]: class MeraLR:  
  
    def __init__(self):  
        self.coef_ = None  
        self.intercept_ = None  
  
    def fit(self,X_train,y_train):  
        X_train = np.insert(X_train,0,1,axis=1)  
  
        # calculate the coeffs  
        betas = np.linalg.inv(np.dot(X_train.T,X_train)).dot(X_train.T).dot(y_train)  
        self.intercept_ = betas[0]  
        self.coef_ = betas[1:]  
  
    def predict(self,X_test):  
        y_pred = np.dot(X_test,self.coef_) + self.intercept_  
        return y_pred
```

```
In [49]: Clr=MeraLR()
```

```
In [50]: Clr.fit(X_train,y_train)
```

```
In [51]: y_pred=Clr.predict(X_test)
```

```
In [ ]: Clr.predict(X_test.loc[[788110]]) # Double brackets to keep it as a DataFrame
```

```
In [261... print("Intercept:", Clr.intercept_) # Access the intercept
print("Coefficients:", Clr.coef_)
```

Intercept: -26.970956487999274

Coefficients: [0.12077798 0.1054747 0.08960371 -0.14571446 0.28704713 0.01956395
0.0213306 0.11535533]

REGRESSION METRICES USING SCIKIT LEARN

```
In [43]: from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error,root_me
print("R2 SCORE :",r2_score(y_test,y_pred))
print("MEAN ABSOLUTE ERROR :",mean_absolute_error(y_test,y_pred))
print("MEAN SQUARED ERROR :",mean_squared_error(y_test,y_pred))
print("ROOT MEAN SQUARED ERROR :",root_mean_squared_error(y_test,y_pred))
```

R2 SCORE : 0.570114265275778

MEAN ABSOLUTE ERROR : 8.226419967037103

MEAN SQUARED ERROR : 105.76432225737778

ROOT MEAN SQUARED ERROR : 10.284178249008415

```
In [ ]:
```

REGRESSION METRICES USING CUSTOM CODE

```
In [52]: num=sum((y_test-y_pred)**2)
den=sum((y_test-y_test.mean())**2)
R2score=1-(num/den)
MAE=((sum(abs(y_test-y_pred)))/len(y_test))
MSE=(sum((y_test-y_pred)**2))/(len(y_test))
RMSE=np.sqrt((sum((y_test-y_pred)**2)/len(y_test)))

print("R2 SCORE :",R2score)
print("MEAN ABSOLUTE ERROR :",MAE)
print("MEAN SQUARED ERROR :",MSE)
print("ROOT MEAN SQUARED ERROR :",RMSE)
```

R2 SCORE : 0.5701142652758168

MEAN ABSOLUTE ERROR : 8.226419967035437

MEAN SQUARED ERROR : 105.76432225736826

ROOT MEAN SQUARED ERROR : 10.284178249007953

In []: trh