In [1]:
```
import numpy as np
import pandas as pd
```

In [74]:
```
#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")
X=df.iloc[:,0:8]
y=df.iloc[:,-1]
print("Ddimension of Dataset",df.shape)
df.head()
```

```
Ddimension of Dataset (100000, 9)
```

Out[74]:

	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 [2]:
```
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[2]:

	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 [3]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=2,test_size=0.2)
```

In [4]:
```
import numpy as np
import pandas as pd
```

```
from sklearn.preprocessing import StandardScaler

class SGDRegressor:

    def __init__(self, learning_rate=0.001, epochs=100):
        self.coef_ = None
        self.intercept_ = None
        self.lr = learning_rate
        self.epochs = epochs
        self.scaler = StandardScaler()  # Add feature scaling

    def fit(self, X_train, y_train):
        # Handle Pandas DataFrame
        if isinstance(X_train, pd.DataFrame):
            X_train = X_train.to_numpy()
        if isinstance(y_train, pd.Series):
            y_train = y_train.to_numpy()

        # Scale the features
        X_train = self.scaler.fit_transform(X_train)

        # Initialize coefficients
        self.intercept_ = 0
        self.coef_ = np.ones(X_train.shape[1])

        for i in range(self.epochs):
            for j in range(X_train.shape[0]):
                idx = np.random.randint(0, X_train.shape[0])

                y_hat = np.dot(X_train[idx], self.coef_) + self.intercept_

                # Compute gradients
                intercept_der = -2 * (y_train[idx] - y_hat)
                self.intercept_ -= self.lr * intercept_der

                coef_der = -2 * np.dot((y_train[idx] - y_hat), X_train[idx])
                self.coef_ -= self.lr * coef_der

        print("Final Intercept:", self.intercept_)
        print("Final Coefficients:", self.coef_)

    def predict(self, X_test):
        X_test = self.scaler.transform(X_test)  # Apply same scaling
        return np.dot(X_test, self.coef_) + self.intercept_
```

In [5]:
```
sgd = SGDRegressor(learning_rate=0.001, epochs=1000)
```

In [6]:
```
sgd.fit(X_train,y_train)
```

```
Final Intercept: 36.2690609578484
Final Coefficients: [13.1255909   8.74070632  5.23094057 -4.21300816  1.73661968  1.
07329783
  0.84438508  6.42167596]
```

In [7]:
```
y_pred = sgd.predict(X_test)
```

```
C:\Users\bhush\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\uti
ls\validation.py:2732: UserWarning: X has feature names, but StandardScaler was fitt
ed without feature names
  warnings.warn(
```

In [9]:
```
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 : 0.546460418859912
MEAN ABSOLUTE ERROR : 8.371917366324283
MEAN SQUARED ERROR  : 111.5838525019882
ROOT MEAN SQUARED ERROR  : 10.563325825798815
```

In [ ]: