In [35]:

```
from sklearn.datasets import load_diabetes
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
import time
```

In [2]:

```
X,y = load_diabetes(return_X_y=True)
```

In [3]:

```
print(X.shape)
print(y.shape)
```

```
(442, 10)
(442,)
```

In [4]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=
```

In [5]:

```
reg = LinearRegression()
reg.fit(X_train,y_train)
```

Out[5]:

```
LinearRegression()
```

In [6]:

```
print(reg.coef_)
print(reg.intercept_)
```

```
[  -9.16088483 -205.46225988  516.68462383  340.62734108 -895.54360867
  561.21453306  153.88478595  126.73431596  861.12139955   52.4198283
6]
151.88334520854633
```

In [7]:

```
y_pred = reg.predict(X_test)
r2_score(y_test,y_pred)
```

Out[7]:

```
0.4399387660024644
```

In [8]:

```
X_train.shape
```

Out[8]:

```
(353, 10)
```

In [25]:

```
class SGDRegressor:
    def __init__ (self,learning_rate=0.01,epochs=100):
        
        self.coef_ = None
        self.intercept_ = None
        self.lr = learning_rate
        self.epochs = epochs
        
    def fit (self,X_train,y_train):
        # init ypur coef
        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_
                
                intercept_der  = -2 * (y_train[idx] - y_hat)
                
                self.intercept_ = self.intercept_ - (self.lr * intercept_der)
                
                coef_der = -2 * np.dot((y_train[idx] - y_hat),X_train[idx])
                self.coef_ = self.coef_ - (self.lr * coef_der)
        print(self.coef_,self.intercept_)
        
    def predict (self,X_test):
        return np.dot(X_test,self.coef_) + self.intercept_
```

In [36]:

```
sgd = SGDRegressor(learning_rate=0.01,epochs=40)
```

In [37]:

```
start = time.time()
sgd.fit(X_train,y_train)
print("Time taken is ",time.time()-start)
```

```
[  66.21572333  -53.01738948  313.35545407  219.51176897   31.20338125
  -10.09962766 -156.30198002  129.4828785   285.07616187  128.5234913
6] 155.00769608567956
Time taken is  0.16379976272583008
```

In [38]:

```
y_pred = sgd.predict(X_test)
```

In [39]:

```
r2_score(y_test,y_pred)
```

Out[39]:

```
0.4196427381847576
```

In [40]:

```
# using sklearn model
from sklearn.linear_model import SGDRegressor
```

In [41]:

```
reg = SGDRegressor(max_iter=100,learning_rate='constant',eta0=0.01)
```

In [42]:

```
reg.fit(X_train,y_train)
```

Out[42]:

```
SGDRegressor(learning_rate='constant', max_iter=100)
```

In [43]:

```
y_pred = reg.predict(X_test)
```

In [44]:

```
r2_score(y_test,y_pred)
```

Out[44]:

```
0.426623172525575
```

In [ ]:

```

```