Building linear regression model

In [82]:

```
#importing libraries that we need
import pandas as pd
import math
#importing library and later on we will import dataset from sklearn.datasets
from sklearn import datasets
```

In [83]:

```
#dataset
dataset = datasets.load_diabetes()
```

In [84]:

```
#printing description of dataset
print(dataset.DESCR)
```

```
.. _diabetes_dataset:

Diabetes dataset
----------------

Ten baseline variables, age, sex, body mass index, average blood
pressure, and six blood serum measurements were obtained for each of n =
442 diabetes patients, as well as the response of interest, a
quantitative measure of disease progression one year after baseline.

**Data Set Characteristics:**

  :Number of Instances: 442

  :Number of Attributes: First 10 columns are numeric predictive values

  :Target: Column 11 is a quantitative measure of disease progression one year after baseline

  :Attribute Information:
      - age     age in years
      - sex
      - bmi     body mass index
      - bp      average blood pressure
      - s1      tc, T-Cells (a type of white blood cells)
      - s2      ldl, low-density lipoproteins
      - s3      hdl, high-density lipoproteins
      - s4      tch, thyroid stimulating hormone
      - s5      ltg, lamotrigine
      - s6      glu, blood sugar level

Note: Each of these 10 feature variables have been mean centered and scaled by the standard deviat
ion times `n_samples` (i.e. the sum of squares of each column totals 1).

Source URL:
https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html

For more information see:
Bradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least Angle Regressio
n," Annals of Statistics (with discussion), 407-499.
(https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)
```

Feature Names

In [85]:

```
print(dataset.feature_names)
```

```
['age', 'sex', 'bmi', 'bp', 's1', 's2', 's3', 's4', 's5', 's6']
```

Creating X and Y data matrices

In [86]:

```
X = dataset.data
Y = dataset.target
```

In [87]:

```
print(X.shape)
print(Y.shape)
```

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

Data Split

In [88]:

```
from sklearn.model_selection import train_test_split
```

Dividing dataset into 80/20 for training and testing

In [89]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2)
```

Dimension for Training and Testing data

In [90]:

```
X_train.shape, Y_train.shape
```

Out[90]:

```
((353, 10), (353,))
```

In [91]:

```
X_test.shape, Y_test.shape
```

Out[91]:

```
((89, 10), (89,))
```

Building a Linear Regression Model

In [92]:

```
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

In [93]:

```
model = linear_model.LinearRegression()
```

Training model

In [94]:

```
model.fit(X_train, Y_train)
```

Out[94]:

```
LinearRegression()
```

Predicting

In [95]:

```
Y_pred = model.predict(X_test)
```

Prediction Results

Print model performance

In [96]:

```
print('Coefficients: ', model.coef_)
print("Intercept: ", model.intercept_)
print("MSE: %.2f" %  mean_squared_error(Y_test, Y_pred))
print("Root Mean Squared Error: ", math.sqrt(mean_squared_error(Y_test, Y_pred)))
print("Coeff of determination: %.2f" % r2_score(Y_test, Y_pred))
```

```
Coefficients:  [   7.29633299 -291.28686925  524.86170535  311.43214306 -853.27464515
  526.4381822   177.46028143  236.51017074  788.08205474   85.3919959 ]
Intercept:  153.77535135709772
MSE: 2635.94
Root Mean Squared Error:  51.34137756544958
Coeff of determination: 0.43
```

In [97]:

```
print(Y_test)
```

```
[ 55. 104. 109. 158. 128.  55. 150.  59. 233.  88. 208.  48. 310.  96.
 217.  85. 246. 103. 246. 274. 121. 310. 132.  95. 181. 164.  42.  77.
 150. 261. 257. 135.  68. 245.  93. 150. 132.  97. 123. 206.  83.  97.
 210. 202. 107. 288. 115. 196.  84. 144.  94. 108.  75. 145. 100. 144.
 140.  61. 148. 142.  71. 121. 128.  66. 197. 141. 109.  84.  52. 259.
 122. 242.  57. 163.  43.  99. 178.  91. 109. 206. 142.  89. 167. 198.
  71.  49. 232. 270. 172.]
```

In [98]:

```
print(Y_pred)
```

```
[152.08892515  34.62562271 169.84049258  68.99442219  99.95076326
  75.21172991 125.68112628  71.33111401 198.2944626  149.29042909
 228.76395517 196.75530165 261.01735895  87.64791123 180.73130155
  57.71989001 155.16301338 138.42548581 237.52873237 243.96207173
 171.842769   256.94323272 250.51619527 153.27152472 173.91931541
 181.93489088  79.93009759  62.88835391 149.09872274 240.33677159
 189.69227352 101.65326274 121.88340161 162.94618495 142.8321951
 211.31035065 118.25169182 123.44322286 198.96921936 170.18426569
 143.22429396 115.37756299 152.45333564 144.91019532 177.40841779
 211.35978676 147.50255804 161.2083162  182.8319508  123.01967195
 104.44713246 105.13386212  70.81139064 119.18016203 154.83597106
 159.89463934 176.81665687 114.54434614 138.61692226 141.35904069
 112.21159629 219.67927048 174.84315776 179.98078315 194.65306398
 167.6702523  162.00961391  95.65497034  67.24518607 159.84979703
 198.99092362 175.11084172  64.38867381 215.81508468  55.7592632
  54.76996121 189.34151291 154.16583631 108.74935804 168.80576706
 107.48806992  79.63061749 187.82550222 212.11973487  79.80160374
 101.52203216 228.81212289 240.09577225 150.90666564]
```

Scatter Plots

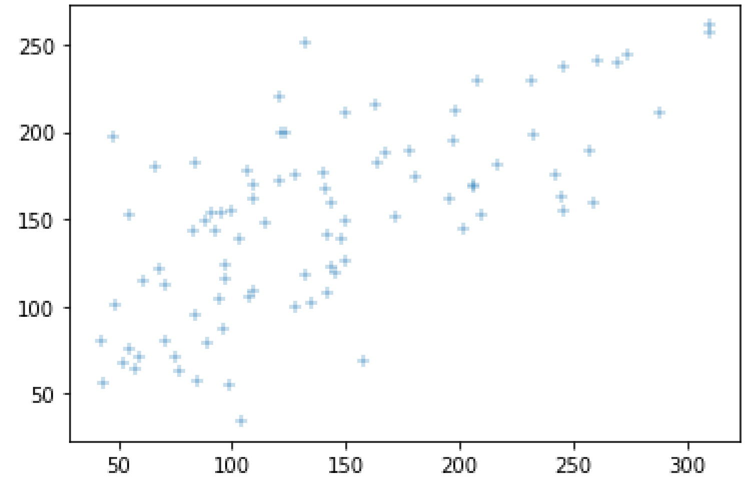
In [99]:

```
import seaborn as sns
sns.scatterplot(Y_test,Y_pred, marker = '+')
```

```
C:\Users\nbrar\anaconda3\envs\tensorflow_env\lib\site-packages\seaborn\_decorators.py:43: FutureWa
rning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid posit
ional argument will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```

Out[99]:

```
<AxesSubplot:>
```



Inference

As we can see that this model has very low accuracy as the training data is not big enough