In [1]:
```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

In [2]:
```
df=pd.read_csv("C:\\Users\\JOHNSON\\Downloads\\my notes\\linear\\assignment\\As
df
```

Out[2]:

	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801
...	...	...	...
9995	Female	66.172652	136.777454
9996	Female	67.067155	170.867906
9997	Female	63.867992	128.475319
9998	Female	69.034243	163.852461
9999	Female	61.944246	113.649103

10000 rows × 3 columns

In [3]:
```
df.head
```

Out[3]:
```
<bound method NDFrame.head of       Gender     Height      Weight
0       Male  73.847017  241.893563
1       Male  68.781904  162.310473
2       Male  74.110105  212.740856
3       Male  71.730978  220.042470
4       Male  69.881796  206.349801
...      ...        ...         ...
9995  Female  66.172652  136.777454
9996  Female  67.067155  170.867906
9997  Female  63.867992  128.475319
9998  Female  69.034243  163.852461
9999  Female  61.944246  113.649103

[10000 rows x 3 columns]>
```

In [4]:
```
df.tail
```

Out[4]:
```
<bound method NDFrame.tail of       Gender     Height      Weight
0       Male  73.847017  241.893563
1       Male  68.781904  162.310473
2       Male  74.110105  212.740856
3       Male  71.730978  220.042470
4       Male  69.881796  206.349801
...      ...        ...         ...
9995  Female  66.172652  136.777454
9996  Female  67.067155  170.867906
9997  Female  63.867992  128.475319
9998  Female  69.034243  163.852461
9999  Female  61.944246  113.649103

[10000 rows x 3 columns]>
```

In [6]:
```
x=df['Height'].array.reshape(-1,1)
x
```

Out[6]:
```
<PandasArray>
[
[73.847017017515],
[68.7819040458903],
[74.1101053917849],
[71.7309784033377],
[69.8817958611153],
[67.2530156878065],
[68.7850812516616],
[68.3485155115879],
[67.018949662883],
[63.4564939783664],
[71.1953822829745],
[71.6408051192206],
[64.7663291334055],
[69.2830700967204],
[69.2437322298112],
[67.6456197004212],
[72.4183166259878],
```

In [8]:
```
y=df['Weight']
y
```

Out[8]:
```
0       241.893563
1       162.310473
2       212.740856
3       220.042470
4       206.349801
           ...
9995    136.777454
9996    170.867906
9997    128.475319
9998    163.852461
9999    113.649103
Name: Weight, Length: 10000, dtype: float64
```

In [9]:

```
#splitting the data
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=
```

In [10]:

```
model=LinearRegression()
model.fit(x_train,y_train)
```

Out[10]:

```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [11]:

```
score=model.score(x_train,y_train)
score
```

Out[11]:

```
0.8545053200432668
```

In [12]:

```
model.coef_
```

Out[12]:

```
array([7.70218561])
```

In [13]:

```
model.intercept_
```

Out[13]:

```
-349.78782058244576
```

Optimisation

In [14]:

```
#Defining parameters
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
```

In [17]:

```
param_grid={
    'copy_X':[True,False],
    'fit_intercept':[True,False],
    'n_jobs':[True,False],
    'positive':[True,False]
}
```

In [18]:

```
#performing grid with cross_validation
grid_search=GridSearchCV(model,param_grid,cv=5,scoring='neg_mean_squared_error
grid_search.fit(x_train,y_train)
```

Out[18]:

```
GridSearchCV(cv=5, estimator=LinearRegression(),
             param_grid={'copy_X': [True, False],
                         'fit_intercept': [True, False],
                         'n_jobs': [True, False], 'positive': [True, False]},
             scoring='neg_mean_squared_error')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [19]:

```
#Best model
best_model=grid_search.best_estimator_
best_model
```

Out[19]:

```
LinearRegression(n_jobs=True, positive=True)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [20]:

```
best_score=model.score(x_test,y_test)
best_score
```

Out[20]:

```
0.85773177770385
```

In [21]:

```
y_pred=best_model.predict(x_test)
y_pred
```

Out[21]:

```
array([179.25399046, 180.34848321, 161.62288801, ..., 129.20288223,
       166.78470522, 101.81227499])
```

In [23]:

```
MSE=mean_squared_error(y_test,y_pred)
MSE
```

Out[23]:

```
149.00350418448116
```

In [ ]: