データの読み込み

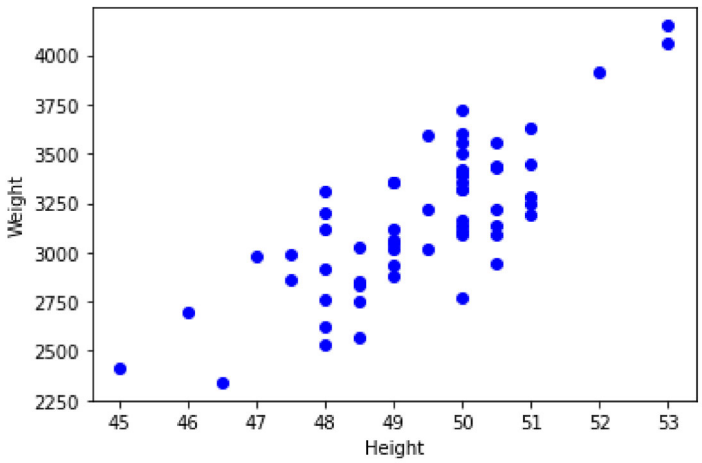
In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
#  以上3つのライブラリが「3種の神器」
```

In [2]:
```
Data=pd.read_csv(
        "D:2022_数理統計学/StatData/StatData02_1.csv",
        skiprows=1, # 1行目を飛ばす
        names=['No', 'Height', 'Weight'])
del Data['No']   # No カラム不要
Data.head()
```

Out[2]:

	Height	Weight
0	46.0	2700
1	49.5	3220
2	50.0	3360
3	50.0	3500
4	49.0	3120

In [3]:
```
# plt.figure(figsize=(6,4))  # 図の大きさ (6,4) がデフォルト
plt.scatter(Data['Height'],Data['Weight'],c='blue')      #散布図を表示
plt.xlabel('Height')
plt.ylabel('Weight')
plt.savefig('StatData02_1_HW-SP.jpeg')  # 画像ファイル(png)として保存（作業ディレクトリに）
plt.savefig('D:2022_数理統計学/PNG/StatData02_1_HW-SP.png')  # 画像ファイル(png)として保存（パス指
# 画像ファイルの形式として, png jpeg eps などが指定できる
```



回帰分析

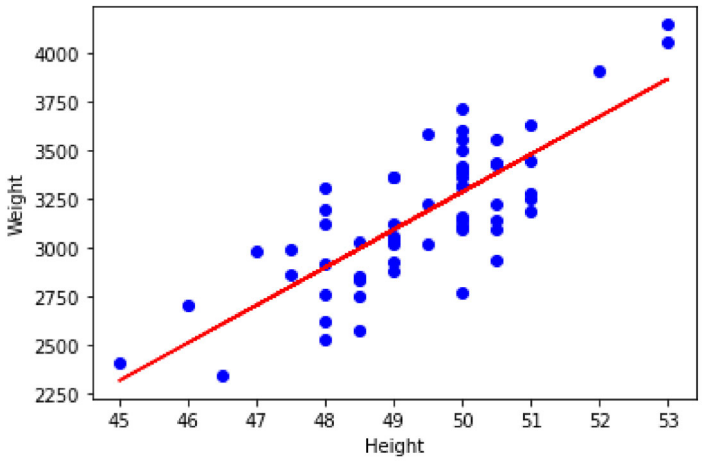
In [4]:
```
# 回帰分析用のライブラリ
from sklearn import linear_model
model=linear_model.LinearRegression()
```

In [5]:
```
x=np.c_[Data['Height']]     #変量 Height をコラムベクトルとして抽出
y=np.c_[Data['Weight']]     #変量 Weight をコラムベクトルとして抽出
model.fit(x,y)              # 回帰直線を求める
```

Out[5]:
```
LinearRegression()
```

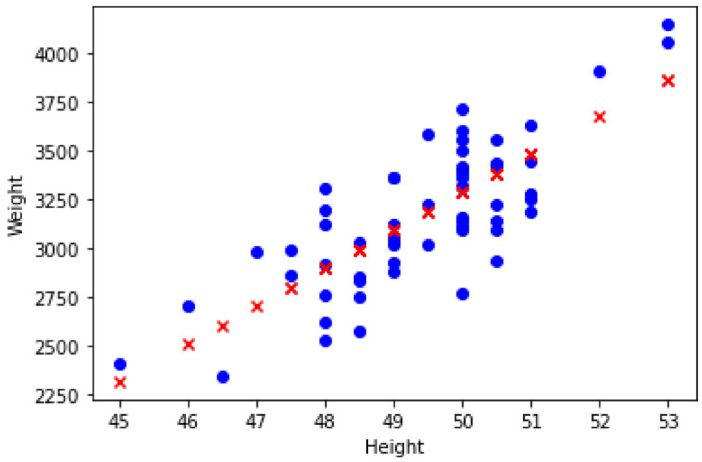
In [6]:
```
plt.scatter(x,y,color = 'blue')                 #散布図
plt.plot(x,model.predict(x),color = 'red')      #回帰直線
plt.xlabel('Height')
plt.ylabel('Weight')
```

Out[6]:
```
Text(0, 0.5, 'Weight')
```



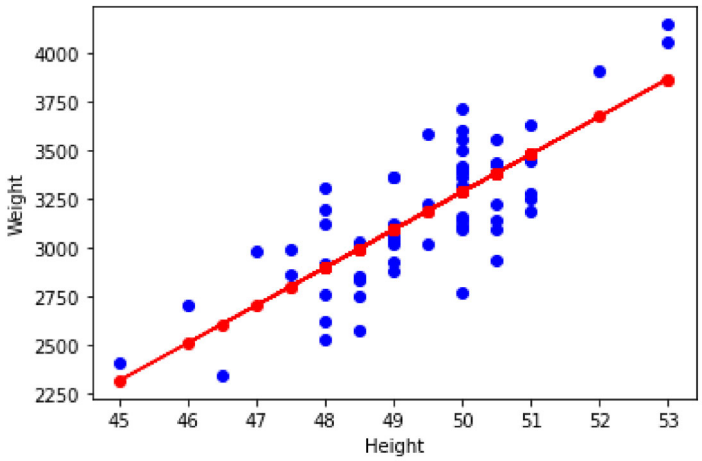
In [7]:
```
plt.scatter(x,y,color='blue')                   # 散布図
plt.scatter(x,model.predict(x),color='red', marker= 'x')    # 各xに対するyの予想値(fitted value)をプ
plt.xlabel('Height')
plt.ylabel('Weight')
```

Out[7]:
```
Text(0, 0.5, 'Weight')
```



In [8]:
```
plt.scatter(x,y,c='blue')                                    #散布図
plt.plot(x,model.predict(x),c='red')                         #回帰直線
plt.scatter(x,model.predict(x),c='red', marker='o')   #fitted values を赤oで表示
plt.xlabel('Height')
plt.ylabel('Weight')
```

Out[8]:
```
Text(0, 0.5, 'Weight')
```



In [9]:
```
p=model.coef_              # 回帰直線 y=px+q の傾き
q=model.intercept_         # 回帰直線 y=px+q の切片
p,q
```

Out[9]:
```
(array([[194.18308964]]), array([-6423.04767959]))
```

In [10]:
```
# p, q ともアレイになって出力されるので、数値だけ取り出す
p=p[0][0]  # [0] はアレイの第0番目の要素を抽出する
q=q[0]
print(p,q)
```

```
194.18308963763505 -6423.047679593132
```

In [11]:
```
# 回帰直線の方程式
print('y=px+q', 'p='+str(np.round(p,2)), 'q='+str(np.round(q,2)))
```

```
y=px+q p=194.18 q=-6423.05
```

In [ ]: