HW 09. Linear Regression

We continue to work with the same SOCR data source as the last time.

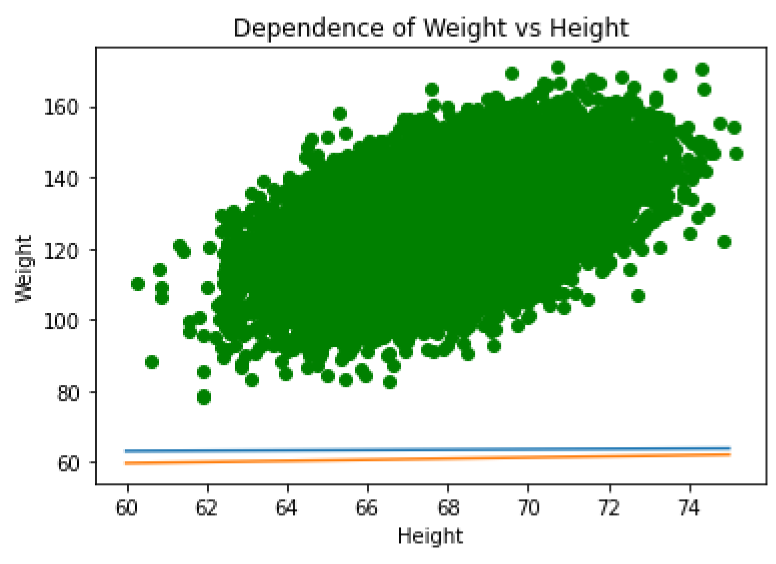
In [1]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.optimize import minimize_scalar
from scipy.optimize import minimize
%matplotlib inline
```

Import the data to a dataframe:

In [2]:
```python
data = pd.read_csv('teenagers.csv', index_col='Index')
```

Task 1: Use the method plot() from the module matplotlib.pyplot to plot the dependence of the weight on the height from the dataset. Add the plots of the two linear regressions to the graph: one with the coefficients $(w_0, w_1) = (60, 0.05)$ and the other with $(w_0, w_1) = (50, 0.16)$. Do the regressions approximate the data well? Don't forget to label the axis and add a title to the figure.

In [16]:
```python
fig = plt.figure()
ax = plt.axes()
plt.plot(data['Height'], data["Weight"], 'o', color = 'green')
xx = np.linspace(60, 75, 200)
w0_1, w1_1 = 60, 0.05
w0_2, w1_2 = 50, 0.16
plt.plot(xx, w1_1 * xx + w0_1)
plt.plot(xx, w1_2 * xx + w0_2)
ax.set(xlabel = "Height", ylabel = "Weight", title = "Dependence of Weight vs Height")
plt.show()

print("The regressions do not appear to be good approximations.")
```




```
The regressions do not appear to be good approximations.
```

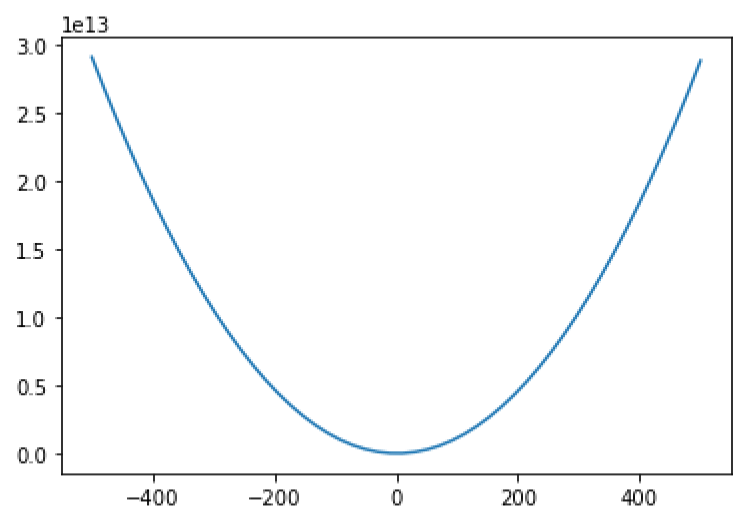
Task 2: Write a python fucntion error(X, Y, w0, w1), which calculates the Loss Function $Q(w_0, w_1) = \sum_{i=1}^{n}(y_i - (w_0 + w_1 * x_i))^2$

In [17]:
```python
def error(x , y ,w0, w1):
    retVal = 0
    errorArray = (y - (w0 + w1 * x)) ** 2
    retVal = sum(errorArray)
    return retVal
```

Task 3: Plot the loss function vs w_1 with a fixed value of $w_0 = 50$ for your dataset. Where approximately does the function reaches its minimum?

In [36]:
```python
fig2 = plt.figure()
ax2 = plt.axes()
errors_len = 50
ww1 = np.linspace(-500, 500, errors_len)
ww0 = np.zeros(errors_len) + 50
errors_arr = np.zeros(errors_len)
for i in range(0, errors_len):
    errors_arr[i] = error(data["Height"], data["Weight"], ww0[i], ww1[i])
ax2.plot(ww1, errors_arr)

print("The minimum is approximatrely at x = 0.")
```

```
The minimum is approximatrely at x = 0.
```



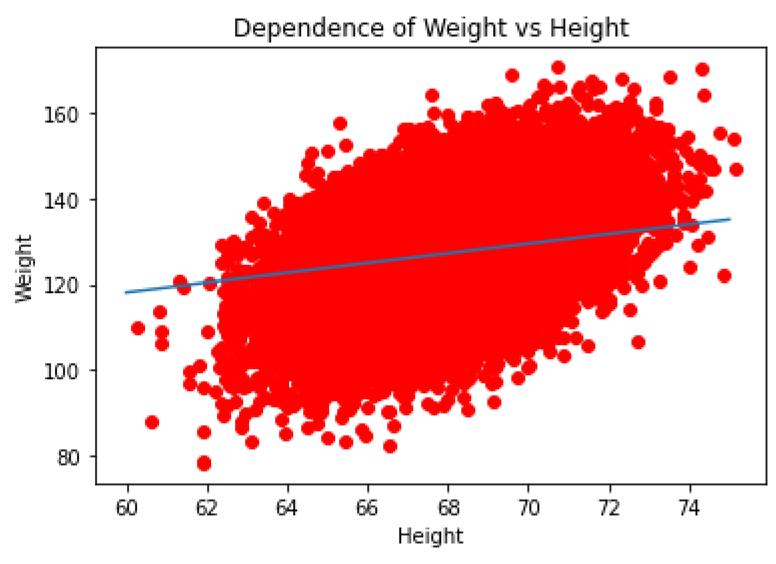
Task 4: Use the method *minimize_scalar* from the module *scipy.optimize* to find the minimum of your Loss function for a fixed value of $w_0 = 50$. Make the method to look for the minimum in the range $-5 \leq w_1 \leq 5$.

In [150…
```python
def error_fun_50(w1):
    return error(data["Height"], data["Weight"], 5, w1)

w1_opt = minimize_scalar(error_fun_50, bounds = (-100, 300))
print( "The minimumum is located at x =", w1_opt.x)
```

```
The minimumum is located at x = 1.7964741480102366
```

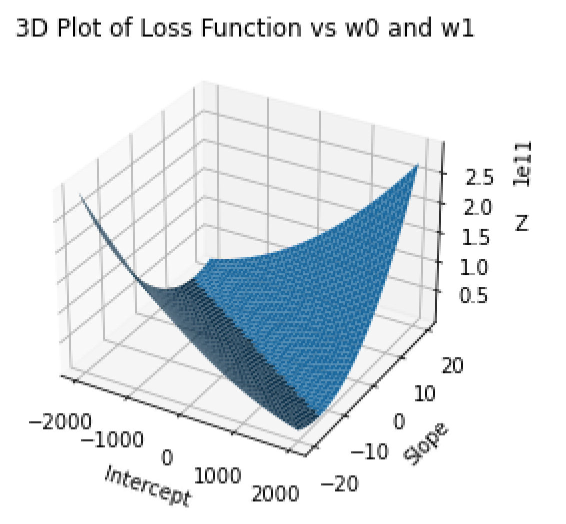
Task 5: Use the method plot() from the module matplotlib.pyplot to plot the dependence of the weight on the height from the dataset. Add to the graph a plot of the linear regression with $w_0 = 50$, and $w_0 = \text{the value you found in Task 4}$. Does this linear regression model the data better than the one from the Task 1?

In [43]:
```python
fig = plt.figure()
ax = plt.axes()
plt.plot(data['Height'], data["Weight"], 'o', color = 'red')
xx = np.linspace(60, 75, 200)
w0_1, w1_1 = 50, 1.1351597092091665
plt.plot(xx, w1_1 * xx + w0_1)
ax.set(xlabel = "Height", ylabel = "Weight", title = "Dependence of Weight vs Height")
plt.show()
```



Task 6: Make a 3D plot of the Loss function vs both w_0 and w_1. Label the axis and add a title to the plot. According to the plot, where should we look for the miminum of the loss function?

In [93]:
```python
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure()
ax = fig. gca(projection = '3d')

X = np.linspace(-2000, 2000, 40)
Y = np. linspace(-20, 20, 40)

Z = np.zeros((len(X), len(Y)))

def error_2D(w0, w1):
    return error(data["Height"], data["Height"], w0, w1)

for i in range(0, len(X)):
    for j in range(0, len(Y)):
        Z[i, j] = error_2D(X[i], Y[j])

X, Y, = np.meshgrid(X, Y)

surf = ax.plot_surface(X, Y, Z)
ax.set_xlabel("Intercept")
ax.set_ylabel("Slope")
ax.set_zlabel("Z")
ax.set(title = "3D Plot of Loss Function vs w0 and w1")

plt.show()
```



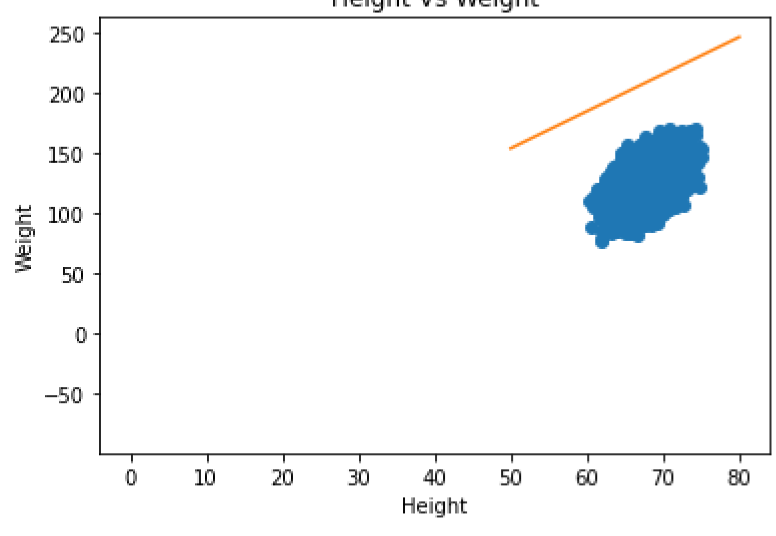
Task 7: Use the method *minimize* from the module *scipy.optimize* to find the minimum of the loss function. Make the method to look for the minimum inside the range $-100 \leq w_0 \leq 100$, $-5 \leq w_1 \leq 5$. Use the argument *method='L-BFGS-B'*

In [170…
```python
def error_2D(w):
    return error(data["Height"], data["Weight"], w[0], w[1])

res = minimize(error_2D, (0, 0), method = "L-BFGS-B")
res.x
res
```

Out[170…
```
      fun: 2539713.3146102373
 hess_inv: <2x2 LbfgsInvHessProduct with dtype=float64>
      jac: array([  -7.8696807 , -449.40971111])
  message: b'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 105
      nit: 20
     njev: 35
   status: 0
  success: True
        x: array([-82.57717703,   3.08349557])
```

Task 8: Use the method plot() from the module matplotlib.pyplot to plot the dependence of the weight on the height from the dataset. Add to the graph a plot of the linear regression with $w_0 = w_0 = \text{the value you found in Task 7}$, and $w_0 = \text{the value you found in Task 7}$.

In [171…
```python
fig = plt.figure()
ax = plt.axes()
ax.plot(data["Height"], data["Weight"], 'o')
xx = np.linspace(50, 80, 200)
w0_1, w1_1 = res.x[0], res.x[1]
ax.plot(xx, w1_1 * xx, + w0_1)
ax.set(xlabel = "Height", ylabel = "Weight", title = "Height Vs Weight")
plt.show()
```



In [ ]: