```
import pandas as pd
```

```
data = pd.read_csv('/content/weight-height.csv')
```

```
#checking first 10 elements of dataset

data.head(10)
```

	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
5	Male	67.253016	152.212156
6	Male	68.785081	183.927889
7	Male	68.348516	167.971110
8	Male	67.018950	175.929440
9	Male	63.456494	156.399676

Next steps: Generate code with data View recommended plots

```
#checking the shape of the dataset

data.shape
```

```
(10000, 3)
```

```
#checking if there are null values in the dataset
data.isnull().sum()
```

```
Gender    0
Height    0
Weight    0
dtype: int64
```

```
#Getting a sample of the dataset
data.sample(10)
```

	Gender	Height	Weight
4637	Male	59.868078	117.803842
851	Male	66.631041	170.421149
645	Male	74.824945	220.336367
4177	Male	66.054377	162.517073
3840	Male	71.307852	202.485157
5562	Female	65.220798	151.569784
433	Male	65.334928	174.655697
1865	Male	68.819430	184.718503
3480	Male	70.288582	195.917873
5486	Female	64.241109	142.579014

```
#check bottom of the dataset
data.tail(10)
```

	Gender	Height	Weight
9990	Female	63.179498	141.266100
9991	Female	62.636675	102.853563
9992	Female	62.077832	138.691680
9993	Female	60.030434	97.687432
9994	Female	59.098250	110.529686
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

```
column_to_drop = ['Gender']
```

```
#Drop column Gender
data = data.drop(column_to_drop, axis=1)
```

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

```
#splitting the dataset
x_train, x_test, y_train, y_test = train_test_split(data.Height.values.reshape(-1, 1), data
```

```
x_train.shape
```

```
(7500, 1)
```

```
y_train.shape
```

```
(7500,)
```

```
x_test.shape
```

```
(2500, 1)
```

```
from sklearn.linear_model import LinearRegression
```

```
#The linear regression model
linear_regression = LinearRegression()
```

```
#calling function fit so as to train the model
linear_regression.fit(X= x_train,  y=y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
#linear regression co-efficient (gradient of the slope)
linear_regression.coef_
```

```
array([7.71345358])
```

```
#linear regression y-intercept (where x and y meet)
linear_regression.intercept_
```

```
-350.390277050054
```

$$equation = y = mx + c$$

$$weight = 7.71345358 * height - 350.390277050054$$

```
#calling function predict to test the model
predicted_values = linear_regression.predict(x_test)
```

```
#re-assigning the expected values
expected_values = y_test
```

```
from sklearn.metrics import mean_absolute_error
```

```
#cheking the Mean Abolute Error
print("MAE", mean_absolute_error(expected_values, predicted_values))
```

```
MAE 9.585892229098492
```

```
#lambda function to calculate different weights
predict_weight = (lambda x: linear_regression.coef_ * x + linear_regression.intercept_)
```

```
#Weight predicted for a height of 70.1047862551571
predict_weight(70.1047862551571)
```

```
array([190.35973736])
```

```
import matplotlib.pyplot as plt
```

```
weight = data['Weight']
```

```
weight
```

```
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
```

```
height = data['Height']
```

```
height
```

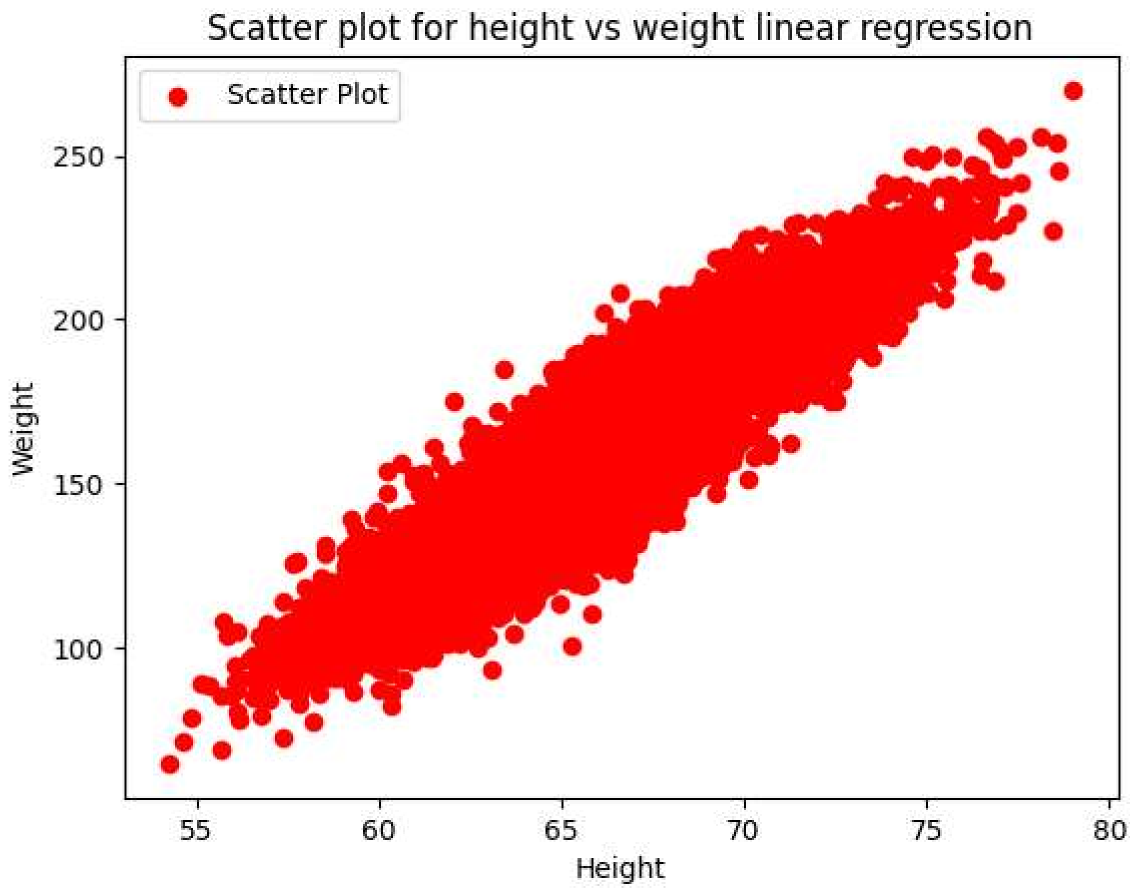
```
0       73.847017
1       68.781904
```

```
2       74.110105
3       71.730978
4       69.881796
          ...
9995    66.172652
9996    67.067155
9997    63.867992
9998    69.034243
9999    61.944246
Name: Height, Length: 10000, dtype: float64
```

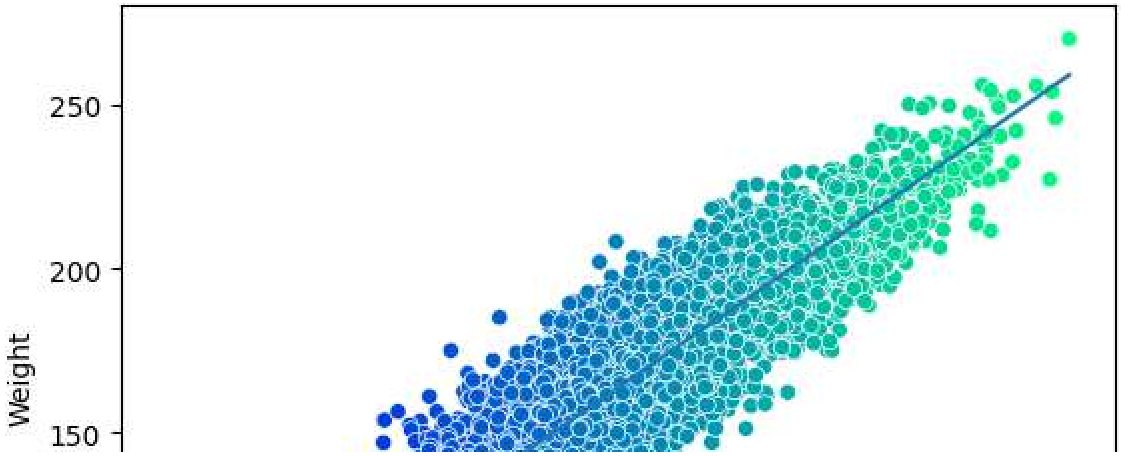
```
plt.figure(figsize=(20,20))
```

```
<Figure size 2000x2000 with 0 Axes>
<Figure size 2000x2000 with 0 Axes>
```

```
#scatter plot
#Attempt 1 at scatter plot
plt.scatter(height, weight, c='red', label='Scatter Plot')

#label for x-axis
plt.xlabel('Height')

#label for y-axis
plt.ylabel('Weight')

#Title for plot
plt.title("Scatter plot for height vs weight linear regression")
plt.legend()
plt.show()
```




```
import seaborn as sns

#seaborn scatterplot
#attempt number 2 of scatter plot
axes = sns.scatterplot(data=data, x='Height', y='Weight', hue='Height', palette='winter', l

x = np.array([min(data.Height.values), max(data.Height.values)])
y = predict(x)
line = plt.plot(x, y)
```




```
height

     0       73.847017
     1       68.781904
     2       74.110105
     3       71.730978
     4       69.881796
               ...
     9995    66.172652
     9996    67.067155
     9997    63.867992
     9998    69.034243
     9999    61.944246
     Name: Height, Length: 10000, dtype: float64
```

Start coding or generate with AI.