```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
df = pd.read_csv("https://raw.githubusercontent.com/EnggQasim/UIT/master/Deep_Learning/Ch
```

```
df.sample(5)
```

| | Gender | Height | Weight |
|---|---|---|---|
| **7253** | Female | 64.715292 | 143.939717 |
| **1319** | Male | 70.404436 | 193.856769 |
| **5868** | Female | 58.376812 | 96.579212 |
| **6634** | Female | 63.806045 | 145.429650 |
| **9884** | Female | 61.808434 | 121.339602 |

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Gender  10000 non-null  object 
 1   Height  10000 non-null  float64
 2   Weight  10000 non-null  float64
dtypes: float64(2), object(1)
memory usage: 234.5+ KB
```

```
df.describe()
```

```
                 Height       Weight
```

```
X = df.Height.values # input, data
y_true = df.Weight.values #output, labels, Asnwers

print(X[:5], y_true[:5], sep='\n\n')
```

```
[73.84701702 68.78190405 74.11010539 71.7309784  69.88179586]

[241.89356318 162.31047252 212.74085556 220.0424703  206.34980062]

  50%      66.318070    161.212928
```

```
#line Equation
def line(x, w=0, b=0):
    return x * w + b

#Cost Functions
def mean_squared_error(y_true, y_pred):
    s = (y_true - y_pred) ** 2
    return np.sqrt(s.mean())
```

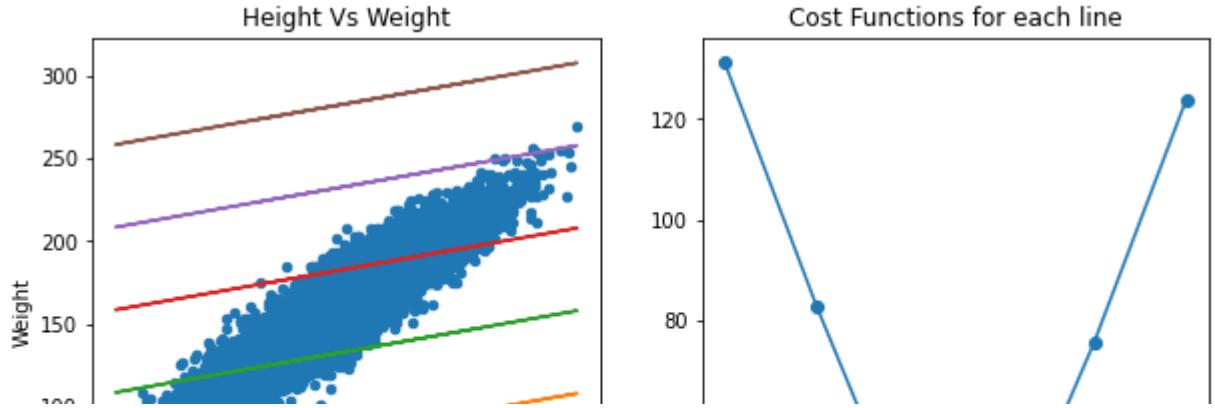
```
plt.figure(figsize=(10,5))
ax1 = plt.subplot(121) # rows,columns, select column 121
df.plot(kind='scatter',
       x="Height",
       y="Weight",
       title="Height Vs Weight", ax=ax1)


bs = [-100, -50, 0, 50, 100, 150]

mses = []
for b in bs:
    y_pred = line(X, w=2, b=b) # predict line
    mse = mean_squared_error(y_true, y_pred)
    mses.append(mse)
    plt.plot(X, y_pred)


ax2 = plt.subplot(122)
plt.plot(bs, mses, 'o-')
plt.title("Cost Functions for each line")
plt.xlabel("b")
```

```
Text(0.5, 0, 'b')
```




```
print(bs)
print(mses)
```

```
[-100, -50, 0, 50, 100, 150]
[131.1417931672291, 82.62957208407575, 38.172273320810234, 32.96359686320369, 75.604728
```

# Try to find w and b (weights) with Deep Learning

```
from tensorflow.keras.models import Sequential #sequential, Functional API
from tensorflow.keras.layers import Dense #each nodes connected with others nodes
from tensorflow.keras.optimizers import Adam, SGD
```

```
import tensorflow
tensorflow.__version__
```

```
'2.8.2'
```

```
print(X[:2],
      y_pred[:2], sep="\n")
```

```
[73.84701702 68.78190405]
[297.69403404 287.56380809]
```

```
model = Sequential()
model.add(Dense(1,input_shape=(1,)))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 1)                 2
```

```
=================================================================
Total params: 2
Trainable params: 2
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(Adam(lr=0.8),loss="mean_squared_error")
```

```
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The
  super(Adam, self).__init__(name, **kwargs)
```

```
model.fit(X,y_true,epochs=50)
```

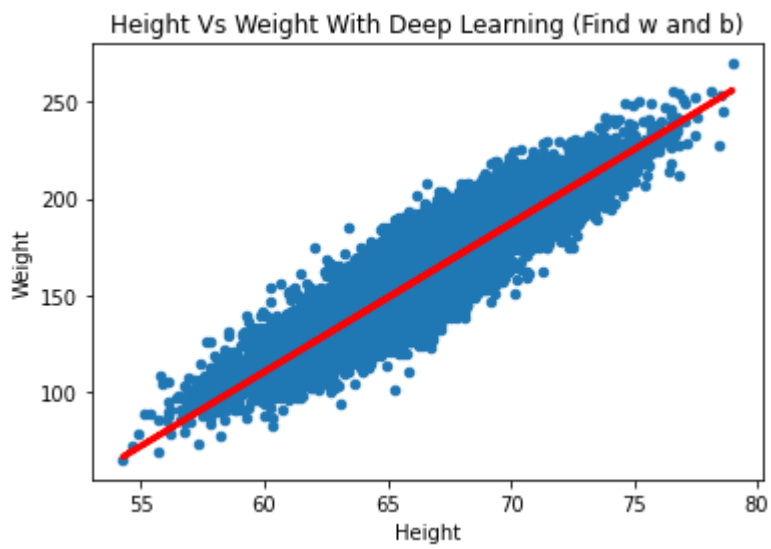
```
313/313 [==============================] - 1s 2ms/step - loss: 195.3859
Epoch 23/50
313/313 [==============================] - 1s 2ms/step - loss: 169.7487
Epoch 24/50
313/313 [==============================] - 1s 2ms/step - loss: 172.9342
Epoch 25/50
313/313 [==============================] - 1s 2ms/step - loss: 180.0589
Epoch 26/50
313/313 [==============================] - 1s 2ms/step - loss: 177.7890
Epoch 27/50
313/313 [==============================] - 0s 1ms/step - loss: 181.3946
Epoch 28/50
313/313 [==============================] - 0s 1ms/step - loss: 176.7419
Epoch 29/50
313/313 [==============================] - 0s 1ms/step - loss: 176.7515
Epoch 30/50
313/313 [==============================] - 0s 1ms/step - loss: 197.0234
Epoch 31/50
313/313 [==============================] - 0s 1ms/step - loss: 178.1048
Epoch 32/50
313/313 [==============================] - 0s 1ms/step - loss: 170.2527
Epoch 33/50
313/313 [==============================] - 0s 1ms/step - loss: 182.2901
Epoch 34/50
313/313 [==============================] - 0s 1ms/step - loss: 175.3139
Epoch 35/50
313/313 [==============================] - 0s 1ms/step - loss: 171.3165
Epoch 36/50
313/313 [==============================] - 0s 1ms/step - loss: 197.2961
Epoch 37/50
313/313 [==============================] - 0s 1ms/step - loss: 170.9256
Epoch 38/50
313/313 [==============================] - 0s 1ms/step - loss: 172.2102
Epoch 39/50
313/313 [==============================] - 0s 1ms/step - loss: 181.2404
Epoch 40/50
313/313 [==============================] - 0s 1ms/step - loss: 166.1049
Epoch 41/50
313/313 [==============================] - 0s 1ms/step - loss: 186.8298
Epoch 42/50
313/313 [==============================] - 0s 1ms/step - loss: 174.7204
```

```
313/313 [==============================] - 0s 1ms/step - loss: 174.7204
Epoch 43/50
313/313 [==============================] - 0s 1ms/step - loss: 174.5288
Epoch 44/50
313/313 [==============================] - 0s 1ms/step - loss: 175.1559
Epoch 45/50
313/313 [==============================] - 0s 1ms/step - loss: 181.3003
Epoch 46/50
313/313 [==============================] - 0s 1ms/step - loss: 182.0315
Epoch 47/50
313/313 [==============================] - 0s 1ms/step - loss: 179.7730
Epoch 48/50
313/313 [==============================] - 0s 1ms/step - loss: 185.1462
Epoch 49/50
313/313 [==============================] - 0s 1ms/step - loss: 183.1624
Epoch 50/50
313/313 [==============================] - 0s 1ms/step - loss: 177.4733
<keras.callbacks.History at 0x7f83a6d35090>
```

```
y_pred = model.predict(X)
```

```
df.plot(kind='scatter',
       x='Height',
       y="Weight",
       title="Height Vs Weight With Deep Learning (Find w and b)")
plt.plot(X, y_pred, color='red', linewidth=3)
```

```
[<matplotlib.lines.Line2D at 0x7f83a5c3e610>]
```




```
model.get_weights()
```

```
[array([[7.6817446]], dtype=float32), array([-350.65726], dtype=float32)]
```

✓ 0s completed at 10:59 AM