Linear Regression

OBJECTIVE: Understand and practice linear regression.

- Very important !

In [14]:

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
```

X and Y data

In [2]:

```
x_train = [1, 2, 3, 4, 5]

#y_train = [2, 4, 6, 8, 10]
y_train = [2+0.1+3, 4-0.3+3, 6+0.15+3, 8+0.1+3, 10-0.12+3]#  Add some noise
```

Initialization

In [3]:

```
#W = tf.Variable(tf.random_normal([1]), name='weight')
#b = tf.Variable(tf.random_normal([1]), name='bias')
w0 = 7.0;
b0 = 5.0;

W = tf.Variable(w0*tf.ones([1]), name='weight')
b = tf.Variable(b0*tf.ones([1]), name='bias')
```

Our hypothesis XW+b

In [4]:

```
hypothesis = x_train * W + b
```

cost/loss function

In [5]:

```
cost = tf.reduce_mean(tf.square(hypothesis - y_train))
```

Optimizer

In [6]:

```
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.01)
train = optimizer.minimize(cost)
```

Launch the graph in a session

In [7]:

```
sess = tf.Session()
```

Initializes global variables in the graph.

In [8]:

```
sess.run(tf.global_variables_initializer())
```

In [9]:

```
for step in range(20):
    if step %3 == 1:
        print(step)
```

```
1
4
7
10
13
16
19
```

In [10]:

```
vw=[]
vb=[]
for step in range(2001):
    sess.run(train)
    w1 = sess.run(W)[0] # slope
    b1 = sess.run(b)[0] # bias
    vw.append(w1)
    vb.append(b1)

    if step % 100 == 0:
        print(step, sess.run(cost), w1, b1)
```

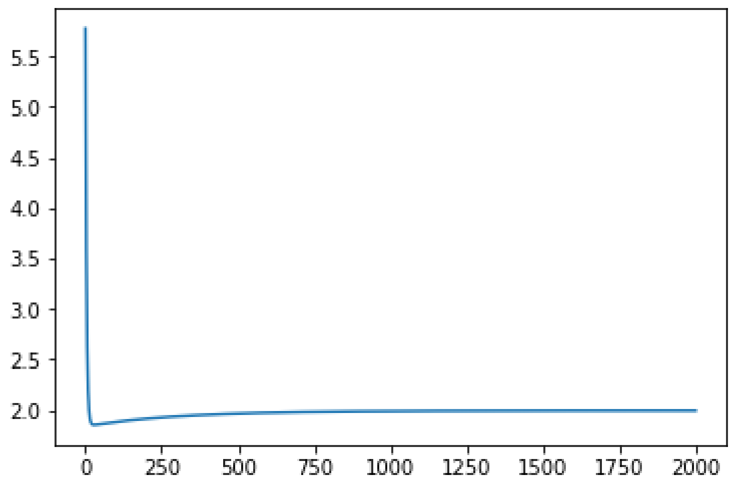
```
0 197.93013 5.779 4.65972
100 0.059207488 1.8834463 3.4043543
200 0.044419073 1.9157813 3.2876153
300 0.03690723 1.9388266 3.2044141
400 0.03309139 1.9552515 3.1451154
500 0.031153206 1.9669574 3.1028528
600 0.03016856 1.9753007 3.0727308
700 0.02966839 1.9812474 3.0512617
800 0.029414233 1.9854856 3.0359604
900 0.029285207 1.9885062 3.025055
1000 0.029219672 1.9906595 3.017281
1100 0.029186413 1.9921937 3.0117414
1200 0.02916945 1.9932873 3.0077934
1300 0.02916089 1.9940666 3.00498
1400 0.029156584 1.994622 3.0029747
1500 0.02915427 1.9950179 3.0015454
1600 0.029153157 1.9952999 3.0005271
1700 0.02915262 1.9955009 2.9998014
1800 0.029152293 1.9956442 2.9992845
1900 0.029152233 1.9957463 2.9989161
2000 0.029152084 1.9958189 2.998654
```

In [11]:

```
plt.plot(vw)
```

Out[11]:

```
[<matplotlib.lines.Line2D at 0x7fc4bcabb048>]
```



Complete training

In [12]:

```
w1 = sess.run(W)[0] # slope
b1 = sess.run(b)[0] # bias
str1 = 'y = ' + str(w1) +'x + ' + str(b1)
print(w1, b1)
print(str1)
```

```
1.9958189 2.998654
y = 1.9958189x + 2.998654
```

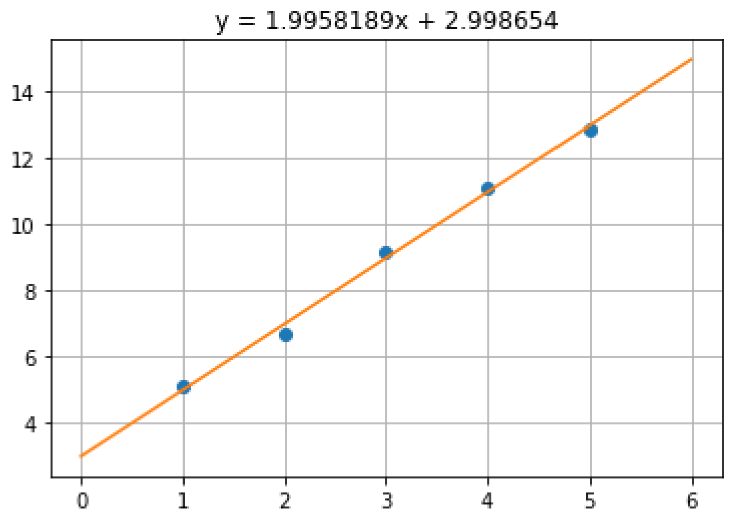
In [13]:

```
plt.figure(1)
plt.plot(x_train, y_train,'o')

x1 = np.linspace(np.min(x_train)-1, np.max(x_train)+1)
y1 = w1*x1 + b1
plt.plot(x1, y1)
plt.grid()
plt.title(str1)
```

Out[13]:

```
Text(0.5,1,'y = 1.9958189x + 2.998654')
```



In [ ]:

```

```