Linear Regression

OBJECTIVE: Understand and practice linear regression.

- Very important !

In [12]:

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

X and Y data

In [2]:

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

y_train = [2+0.1, 4-0.3, 6+0.15] #  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(2001):
    sess.run(train)

    if step % 100 == 0:
        w1 = sess.run(W)[0] # slope
        b1 = sess.run(b)[0] # bias
        print(step, sess.run(cost), w1, b1)
```

```
0 191.49959 6.333 4.6996665
100 0.5531761 1.1931436 1.8244524
200 0.3571646 1.3710523 1.4199102
300 0.23604143 1.5109376 1.101918
400 0.1611947 1.6209002 0.8519473
500 0.11494404 1.7073406 0.6554478
600 0.086363904 1.7752907 0.50098115
700 0.06870318 1.8287058 0.37955633
800 0.057789847 1.870695 0.28410515
900 0.051046133 1.9037024 0.20907182
1000 0.046878953 1.9296488 0.15008886
1100 0.044303745 1.9500452 0.10372298
1200 0.042712525 1.9660789 0.067275114
1300 0.041729197 1.9786826 0.03862365
1400 0.041121576 1.9885905 0.016101135
1500 0.040746186 1.9963787 -0.0016036469
1600 0.040514156 2.002501 -0.015521128
1700 0.040370733 2.007314 -0.02646201
1800 0.040282138 2.0110972 -0.035062194
1900 0.040227447 2.0140712 -0.04182288
2000 0.040193584 2.016409 -0.047137175
```

Complete training

In [10]:

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

```
2.016409 -0.047137175
y = 2.016409x + -0.047137175
```

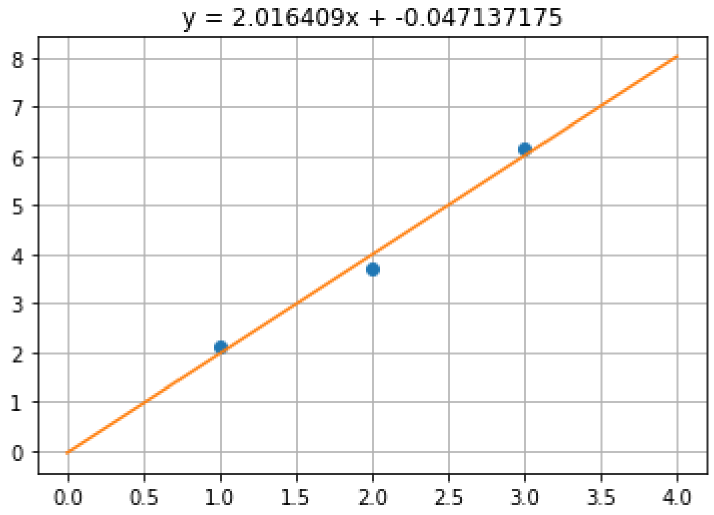
In [11]:

```
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[11]:

```
Text(0.5,1,'y = 2.016409x + -0.047137175')
```



In [ ]: