

Q1: What happens to the result if you repeat the program?

값이 달라진다. 그 이유는 랜덤으로 들어가기 때문이다.

Q2: What is the impact of train_tot?

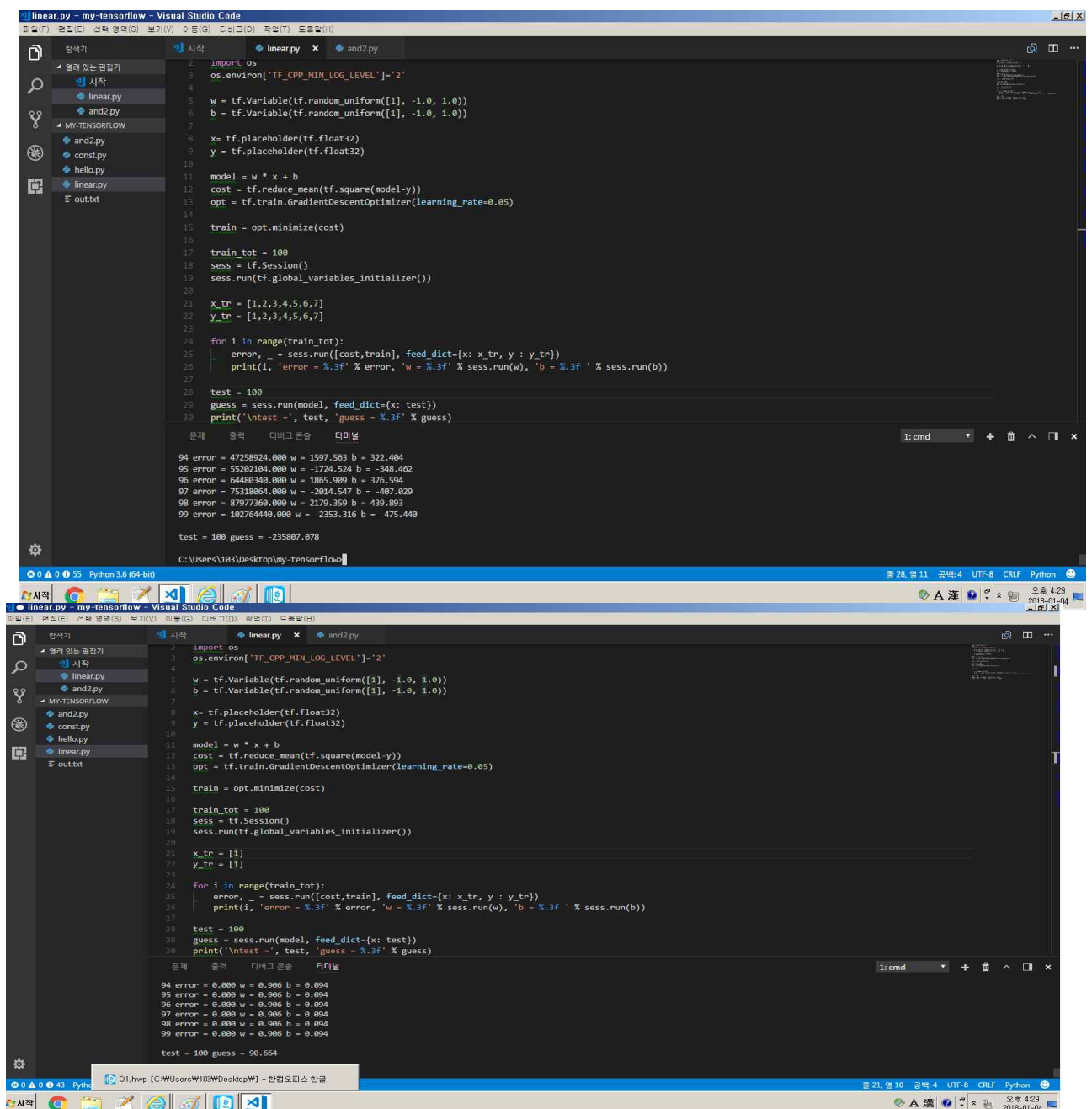
반복횟수를 지정해 주는 것

Q3: What is the impact of learning_rate?

오차 값과 추측 값을 더 좋은 상위 값으로 수렴하게 한다.

Q4: What happens if you drop
(or add) more training data?

drop하면 오차 값이 부정확해지고
add하면 오차 값이 커진다.



```
linear.py - my-tensorflow - Visual Studio Code
2 import os
3 os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
4
5 w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
6 b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
7
8 x = tf.placeholder(tf.float32)
9 y = tf.placeholder(tf.float32)
10
11 model = w * x + b
12 cost = tf.reduce_mean(tf.square(model-y))
13 opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
14
15 train = opt.minimize(cost)
16
17 train_tot = 100
18 sess = tf.Session()
19 sess.run(tf.global_variables_initializer())
20
21 x_tr = [1,2,3,4,5,6,7]
22 y_tr = [1,2,3,4,5,6,7]
23
24 for i in range(train_tot):
25     error, _ = sess.run([cost,train], feed_dict={x: x_tr, y: y_tr})
26     print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))
27
28 test = 100
29 guess = sess.run(model, feed_dict={x: test})
30 print('\ntest =', test, 'guess = %.3f' % guess)

94 error = 47258924.000 w = 1597.563 b = 322.484
95 error = 55282104.000 w = -1724.524 b = -348.462
96 error = 64480340.000 w = 1865.909 b = 376.594
97 error = 75318064.000 w = -2014.547 b = -407.029
98 error = 8797360.000 w = 2179.359 b = 439.893
99 error = 102764440.000 w = -2353.316 b = -475.440

test = 100 guess = -235807.078
C:\Users\103\Desktop\my-tensorflow

Python 3.6 (64-bit)
21. 오후 4:29
2018-01-04
```

```
linear.py - my-tensorflow - Visual Studio Code
2 import os
3 os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
4
5 w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
6 b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
7
8 x = tf.placeholder(tf.float32)
9 y = tf.placeholder(tf.float32)
10
11 model = w * x + b
12 cost = tf.reduce_mean(tf.square(model-y))
13 opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
14
15 train = opt.minimize(cost)
16
17 train_tot = 100
18 sess = tf.Session()
19 sess.run(tf.global_variables_initializer())
20
21 x_tr = [1]
22 y_tr = [1]
23
24 for i in range(train_tot):
25     error, _ = sess.run([cost,train], feed_dict={x: x_tr, y: y_tr})
26     print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))
27
28 test = 100
29 guess = sess.run(model, feed_dict={x: test})
30 print('\ntest =', test, 'guess = %.3f' % guess)

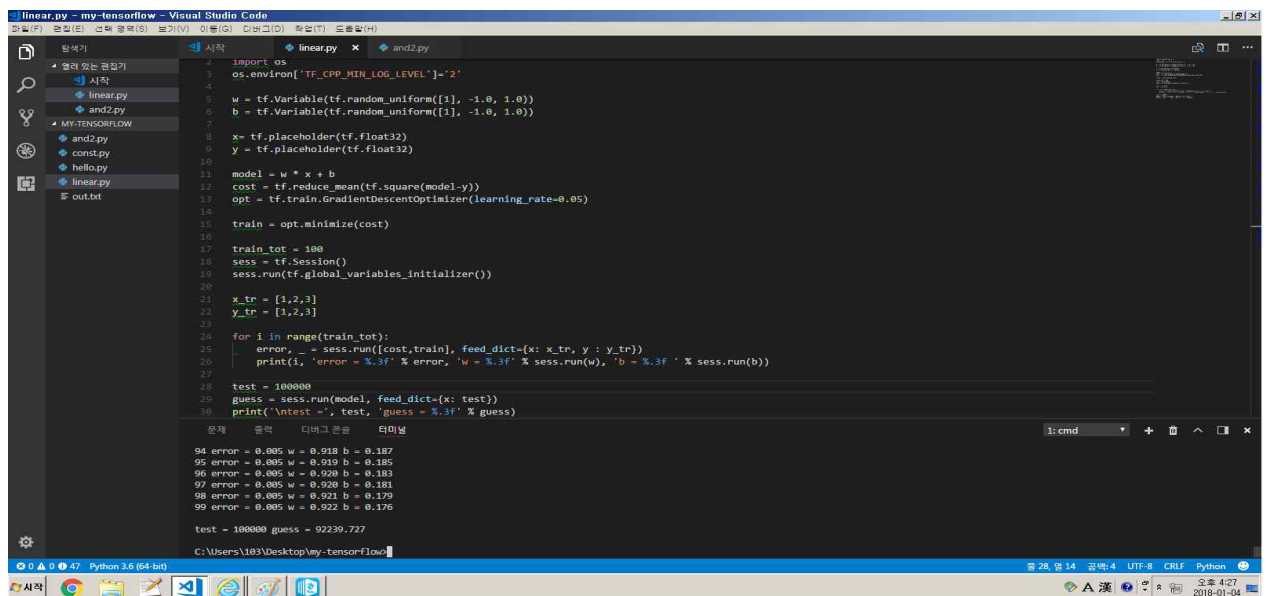
94 error = 0.000 w = 0.906 b = 0.094
95 error = 0.000 w = 0.906 b = 0.094
96 error = 0.000 w = 0.906 b = 0.094
97 error = 0.000 w = 0.906 b = 0.094
98 error = 0.000 w = 0.906 b = 0.094
99 error = 0.000 w = 0.906 b = 0.094

test = 100 guess = 90.664
C:\Users\103\Desktop\my-tensorflow

Python 3.6 (64-bit)
21. 오후 4:29
2018-01-04
```

Q5: What happens if your test data is very big or small?

크면 근사 값이 부정확해지고 작으면 정확도가 올라간다.



```
linear.py - my-tensorflow - Visual Studio Code
import os
os.environ['TF_CPP_MIN_LOG_LEVEL']='2'

w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))

x = tf.placeholder(tf.float32)
y = tf.placeholder(tf.float32)

model = w * x + b
cost = tf.reduce_mean(tf.square(model-y))
opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
train = opt.minimize(cost)

train_tot = 100
sess = tf.Session()
sess.run(tf.global_variables_initializer())

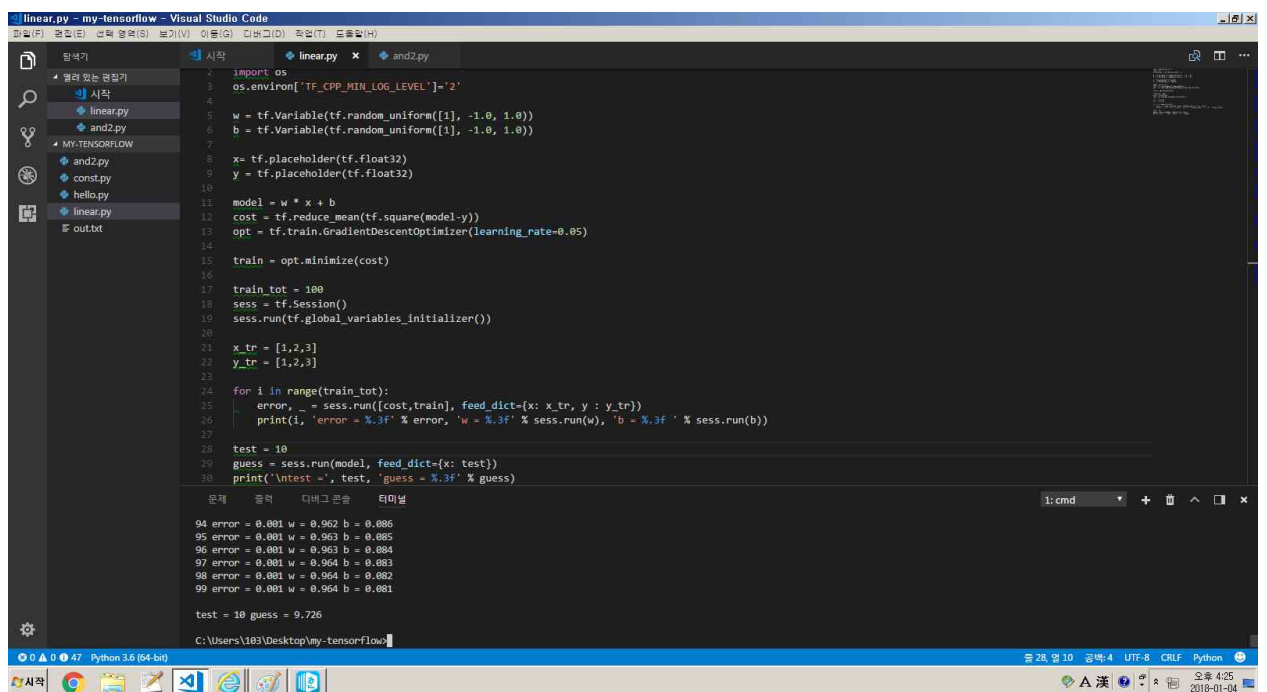
x_tr = [1,2,3]
y_tr = [1,2,3]

for i in range(train_tot):
    error, _ = sess.run([cost, train], feed_dict={x: x_tr, y: y_tr})
    print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))

test = 100000
guess = sess.run(model, feed_dict={x: test})
print('ntest = ', test, 'guess = %.3f' % guess)

94 error = 0.005 w = 0.918 b = 0.187
95 error = 0.005 w = 0.919 b = 0.185
96 error = 0.005 w = 0.920 b = 0.183
97 error = 0.005 w = 0.920 b = 0.181
98 error = 0.005 w = 0.921 b = 0.179
99 error = 0.005 w = 0.922 b = 0.176

test = 100000 guess = 92239.727
C:\Users\103\Desktop\my-tensorflow\
```



```
linear.py - my-tensorflow - Visual Studio Code
import os
os.environ['TF_CPP_MIN_LOG_LEVEL']='2'

w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))

x = tf.placeholder(tf.float32)
y = tf.placeholder(tf.float32)

model = w * x + b
cost = tf.reduce_mean(tf.square(model-y))
opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
train = opt.minimize(cost)

train_tot = 100
sess = tf.Session()
sess.run(tf.global_variables_initializer())

x_tr = [1,2,3]
y_tr = [1,2,3]

for i in range(train_tot):
    error, _ = sess.run([cost, train], feed_dict={x: x_tr, y: y_tr})
    print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))

test = 10
guess = sess.run(model, feed_dict={x: test})
print('ntest = ', test, 'guess = %.3f' % guess)

94 error = 0.001 w = 0.962 b = 0.086
95 error = 0.001 w = 0.963 b = 0.085
96 error = 0.001 w = 0.963 b = 0.084
97 error = 0.001 w = 0.964 b = 0.083
98 error = 0.001 w = 0.964 b = 0.082
99 error = 0.001 w = 0.964 b = 0.081

test = 10 guess = 9.726
C:\Users\103\Desktop\my-tensorflow\
```

Q6: What happens if there is a outlier in the training data?

오차 값이 커지고 추측값이 크게 벗어나간다.

```
1 import os
2 os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
3
4 w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
5 b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
6
7
8 x = tf.placeholder(tf.float32)
9 y = tf.placeholder(tf.float32)
10
11 model = w * x + b
12 cost = tf.reduce_mean(tf.square(model-y))
13 opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
14
15 train = opt.minimize(cost)
16
17 train_tot = 100
18 sess = tf.Session()
19 sess.run(tf.global_variables_initializer())
20
21 x_tr = [1, 2, 3, 4]
22 y_tr = [1, 2, 3, 4]
23
24 for i in range(train_tot):
25     error, _ = sess.run([cost, train], feed_dict={x: x_tr, y: y_tr})
26     print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))
27
28 test = 100
29 guess = sess.run(model, feed_dict={x: test})
30 print('\ntest = ', test, 'guess = %.3f' % guess)
```

94 error = 45.598 w = -1.114 b = 0.878
95 error = 46.878 w = 1.901 b = 1.430
96 error = 48.194 w = -1.157 b = 0.871
97 error = 49.547 w = 1.944 b = 1.439
98 error = 50.939 w = -1.200 b = 0.865
99 error = 52.369 w = 1.908 b = 1.448
test = 100 guess = 200.225

```
1 import os
2 os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
3
4 w = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
5 b = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
6
7
8 x = tf.placeholder(tf.float32)
9 y = tf.placeholder(tf.float32)
10
11 model = w * x + b
12 cost = tf.reduce_mean(tf.square(model-y))
13 opt = tf.train.GradientDescentOptimizer(learning_rate=0.05)
14
15 train = opt.minimize(cost)
16
17 train_tot = 100
18 sess = tf.Session()
19 sess.run(tf.global_variables_initializer())
20
21 x_tr = [1, 2, 3, 5]
22 y_tr = [1, 2, 3, 10]
23
24 for i in range(train_tot):
25     error, _ = sess.run([cost, train], feed_dict={x: x_tr, y: y_tr})
26     print(i, 'error = %.3f' % error, 'w = %.3f' % sess.run(w), 'b = %.3f' % sess.run(b))
27
28 test = 100
29 guess = sess.run(model, feed_dict={x: test})
30 print('\ntest = ', test, 'guess = %.3f' % guess)
```

94 error = 1.884 w = 2.220 b = -2.856
95 error = 1.883 w = 2.221 b = -2.861
96 error = 1.883 w = 2.222 b = -2.866
97 error = 1.882 w = 2.224 b = -2.870
98 error = 1.882 w = 2.225 b = -2.875
99 error = 1.881 w = 2.226 b = -2.879
test = 100 guess = 220.534