Caffe
Keras
Tensorflow
Pytorch
⋮

tf.get_variable( ) is preferred

```
data = np.array([[1.0],[-20.],[100.]])
w = tf.Variable(      )   定义待定的参数
x = tf.placeholder(tf.float32,[3,1])   定义给定的数据
cost = x[0][0]*w**2 - x[0][1]*10*w + x[0][2]*25
train = tf.train.GradientDescentOptimizer(learning rate).
                                    minimize(cost)
init = tf.global_variable_initializer()
```

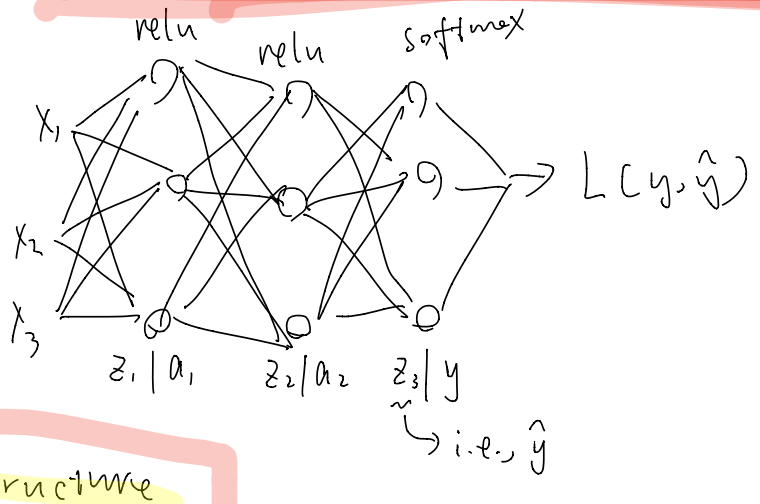
```
session = tf.Session()
session.run(init)   初始化参数
print(session.run(w))   显示当前参数值
session.run(train, feed_dict={x: data})   提供给定的数据并训练一次
print(session.run(w))   显示当前参数值
```

Logistic regression:

$$odd = \frac{p}{1-p}, \quad p = \frac{1}{1+e^{-x}}$$

$$logit(p) = Log(odd) = log\left(\frac{p}{1-p}\right) = log(e^x) = x$$

$$cross\ entropy = y \log p + (1-y)\log(1-p)$$



structure

```
def mol( X_train, Y_train, X_test, Y_test,
         learning_rate, # of epoches, mini_batch_size,
         print_or_not ):
```

定义好

```
    ops.reset_default_graph()        <- tf.set_random_seed()
                                        seed = ...
    X, Y = creat_placeholder(n_x, n_y)
```

框架，tensor

```
parameters = initialize_parameters()
Z3 = forward_propagation(X, parameters)
cost = compute_cost(Z3, Y)
optimizer = tf.train.GradientDescentOptimizer
            (learning_rate = ).minimize(cost)
init = tf.global_variable_initializer()
```

computation with 2 layer loop

```
sess = tf.session()
sess.run(init) # initialize all the parameters

for epoch in range(# of epoches):
    mini_batches = random_mini_batches
        (X_train, Y_train, mini_batch_size,
         seed)

    for mini in mini_batches:
        (mini_X, mini_Y) = mini
        _, mini_cost = sess.run([optimizer, cost],
                                feed_dict={X: mini_X, Y: mini_Y})
        print(mini_cost)
```

同时运行2个tensor，但只用记录cost，用于显示进度，optimizer的用处在于update ① parameter ② 中间量如 $Z_1, a_1, Z_2, a_2, Z_3 \cdots$

计算预测的accuracy

```
correct_prediction = tf.equal(tf.argmax(Z3),
                              tf.argmax(Y))
accuracy = tf.reduce_mean(correct_prediction)
```

预测的label

实际的label

返回模型參數

```
parameter = sess.run(parameter)
return parameter
```