Supplementary Table 1. The TensorFlow code to predict MCI

```
import tensorflow as tf
import numpy as np

mcidata = np.loadtxt('C:\Machine_learning\MCITRAIN1.csv', delimiter=',', dtype=np.float32)

x_data = mcidata[:, 3:5]
y_data = mcidata[: ,[0]]
x_data_std = (mcidata[:, 3:5] - mcidata[:, 3:5].mean()) / mcidata[:, 3:5].std()

X = tf.placeholder(tf.float32, shape=[None, 2])
Y = tf.placeholder(tf.float32, shape=[None, 1])

W = tf.Variable(tf.random_normal([2,1]))
b = tf.Variable(tf.random_normal([1]))

H = tf.sigmoid(tf.matmul(X, W) + b)
cost = -tf.reduce_mean(Y * tf.log(H)+ (1-Y) * tf.log(1-H))
a = tf.Variable(0.5)
optimizer = tf.train.GradientDescentOptimizer(a)
train = optimizer.minimize(cost)

prediction = tf.cast(H > 0.5, dtype=tf.float32)
is_correct = tf.equal(prediction, Y)
accuracy = tf.reduce_mean(tf.cast(is_correct, tf.float32))

init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)

for i in range (10001):
    sess.run(train, feed_dict={X: x_data_std, Y: y_data})
    if i % 1000 == 0:
        print(i, sess.run(cost, feed_dict={X: x_data_std, Y: y_data}))

mcidata = np.loadtxt('C:\Machine_learning\MCITEST1.csv', delimiter=',', dtype=np.float32)


x_data = mcidata[:, 3:5]
y_data = mcidata[: ,[0]]
x_data_std = (mcidata[:, 3:5] - mcidata[:, 3:5].mean()) / mcidata[:, 3:5].std()

h, c, a = sess.run([H, prediction,    accuracy], feed_dict={X: x_data_std, Y: y_data})
print("\nH:", h, "\nCorrect:", c, "\nAccuracy: ", a)
```