贝叶斯 聚类

tf.Variable 通过创建Variable实例向graph中添加变量

Variable需要初始值。

更改值通过assign方法

改变形状，使用 assign+valiable\_shape=False

global\_variables\_initializer() 添加一个op到graph。 这个op可以初始化所有变量。在launch图后，运行这个op

init\_op=tf.global\_variables\_initializer()

with tf.Session() as sess:

sess.run(init\_op)

所有加法、减、除、乘（按元素相乘）、取余等矩阵的算术运算都要求两个张量矩阵是相同的数据类型，否则就会产生错误。可以使用 tf.cast() 将张量从一种数据类型转换为另一种数据类型。

**拓展阅读**

如果在整数张量之间进行除法，最好使用 tf.truediv(a，b)，因为它首先将整数张量转换为浮点类，然后再执行按位相除。



1. **tf.multiply（）两个矩阵中对应元素各自相乘**
2. **tf.matmul（）将矩阵a乘以矩阵b，生成a \* b。**
3. tf.zeros([1]) 【0】 tf.zeros([2])【0 0】
4. tf.random\_uniform([1, 2], -1.0, 1.0) 1行2列的 在【-1,1】之间，均匀分布的值。
5. tf.reduce\_mean 函数用于计算张量tensor沿着指定的数轴（tensor的某一维度）上的的平均值，主要用作降维或者计算tensor（图像）的平均值。

第一个参数input\_tensor： 输入的待降维的tensor;

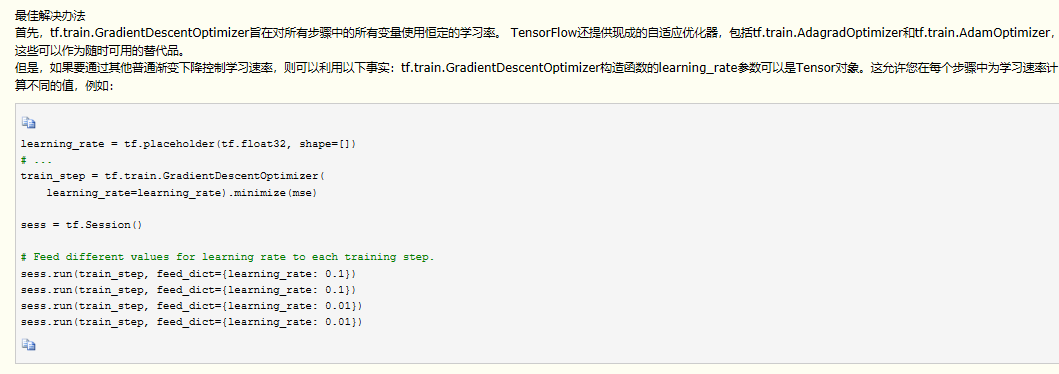
第二个参数axis： 指定的轴，如果不指定，则计算所有元素的均值;

第三个参数keep\_dims：是否降维度，设置为True，输出的结果保持输入tensor的形状，设置为False，输出结果会降低维度;

第四个参数name： 操作的名称;

第五个参数 reduction\_indices：在以前版本中用来指定轴，已弃用;

**首先，tf.train.GradientDescentOptimizer旨在对所有步骤中的所有变量使用恒定的学习率**



train = optimizer.minimize(loss)

其中有三个参数需要注意：

（1）loss：即最小化的目标变量，一般就是训练的目标函数，均方差或者交叉熵；

（2）global\_step：梯度下降一次加1，一般用于记录迭代优化的次数，主要用于参数输出和保存；

（3）var\_list 每次要迭代更新的参数集合。

Tf.cast() 数据转换。

import input\_data

mnist = input\_data.read\_data\_sets("MNIST\_data/", one\_hot=True)

在 TensorFlow 中可以通过三种方式读取数据：

1. 通过feed\_dict传递数据；
2. 从文件中读取数据；
3. 使用预加载的数据；当数据集很小时可以使用，可以在内存中完全加载

优化器的使用：

GradientDecentOptimizer

Optimizer=tf.train.GradientDescentOptimizer(learning\_rate)

Train=Optimizer.minimize(loss)

MomentumOptimizer

Optimizer=tf.train.MomentumOptimizer(learning\_rate=0.01,momentum=0.5).minimize(loss)

AdadeltaOptimizer

Optimizer=tf.train.AdadeltaOptimizer(learning\_rate=0.8,rho=0.95).minimize(loss)

多线程：

两个类Coordinator和tf.QueueRunner 一起用管理多线程

Coordinator:可以同时停止多个线程，并且向等待线程结束的工作线程报告异常

Tf.train.Coordinator() 创建一个线程管理器。

QueueRunner：启动tensor的入队线程，可以用来启动多个工作线程同时将多个tensor（训练数据）推送入文件名称队列中，具体函数：tf.train.start\_queue\_runners

Tf.train.slice\_input\_producer(tensor\_list,num\_epochs=None)

Tensor生成器，每次从tensor 列表中按顺序或随机抽取出一个tensor放入文件名队列。

Tf.train.start\_queue\_runners(sess,coord) 启动队列填充

Tf.train.batch 从文件名称队列中读取文件 准备放入文件队列。

string\_input\_producer每次放出一个文件名。slice\_input\_producer可以既可以同时放出文件名和它对应的label，也可以只放出一个文件名

神经网络分类模型：

导入CSV格式的数据集

建立神经网络分类模型

用训练数据集训练模型

评价模型的准确率

对新样本数据进行分类

matplotlib.pyplot绘制图形