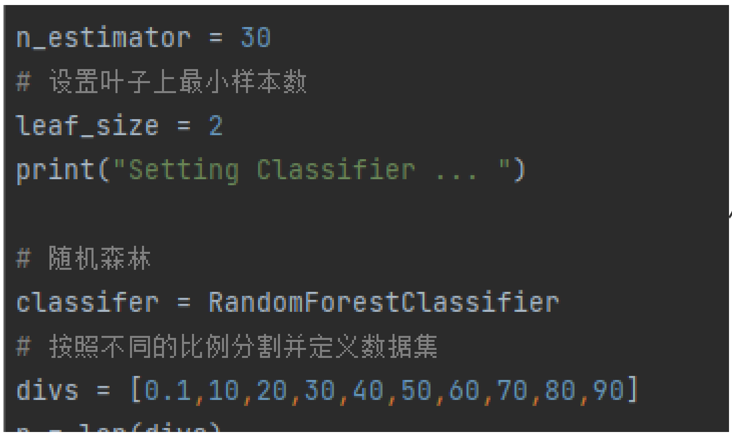
机器学习作业（HW9）

伊辛模型

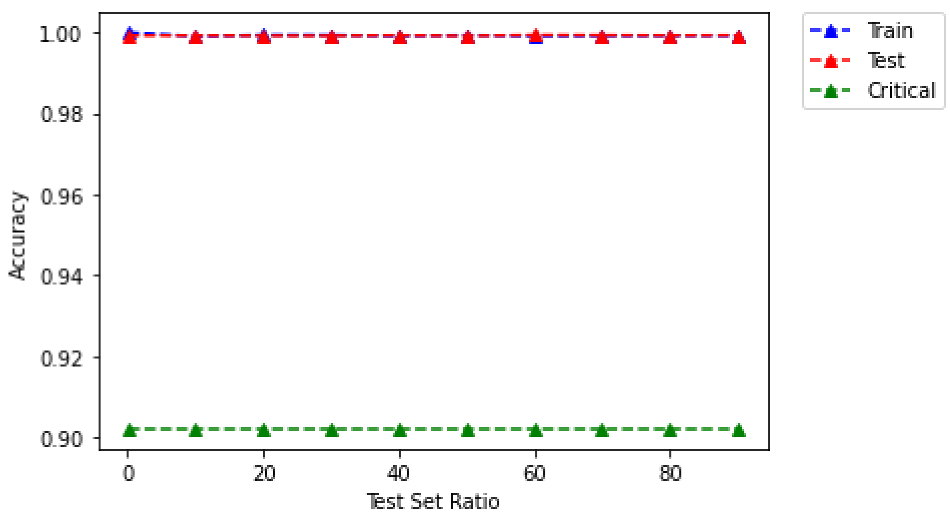
1. 代码见Ising_1.py。取树木棵树为30，树木深度参数设为None，叶子上最小样本数分别取2和1000，对样本比例百分之[0.1,10,20,30,40,50,60,70,80,90]进行遍历，



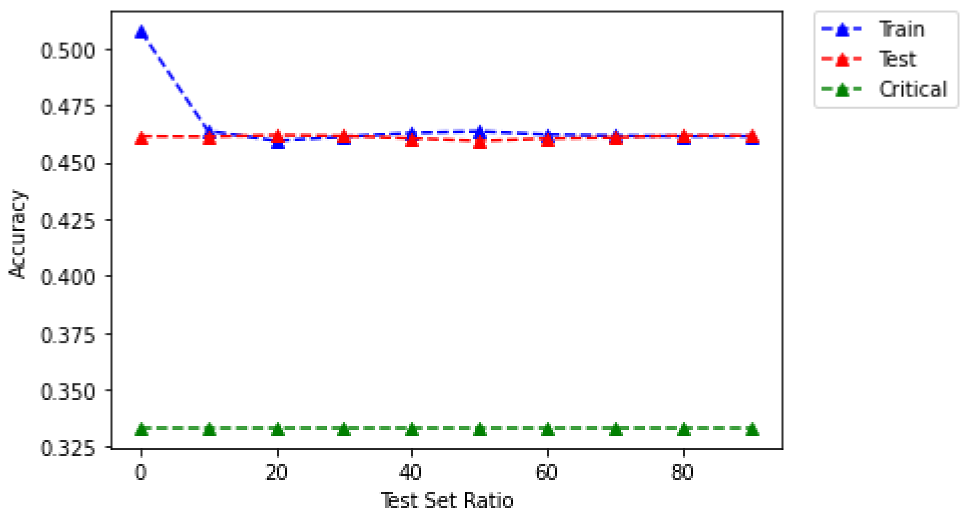
```
n_estimator = 30
# 设置叶子上最小样本数
leaf_size = 2
print("Setting Classifier ... ")

# 随机森林
classifer = RandomForestClassifier
# 按照不同的比例分割并定义数据集
divs = [0.1,10,20,30,40,50,60,70,80,90]
```

作图如下

最小样本数2:



最小样本数10000:



有此可知，训练样本比例对精度影响较小。在最小样本数为10000时Train accuracy在testset比例较低时随testset占比增大而减小。

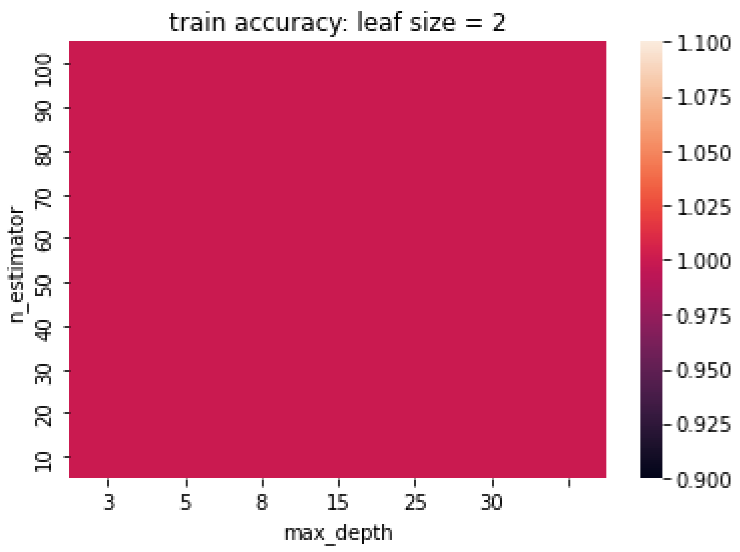
2. 取样本比例0.9:0.1，用OOB_accuracy衡量超参数优劣

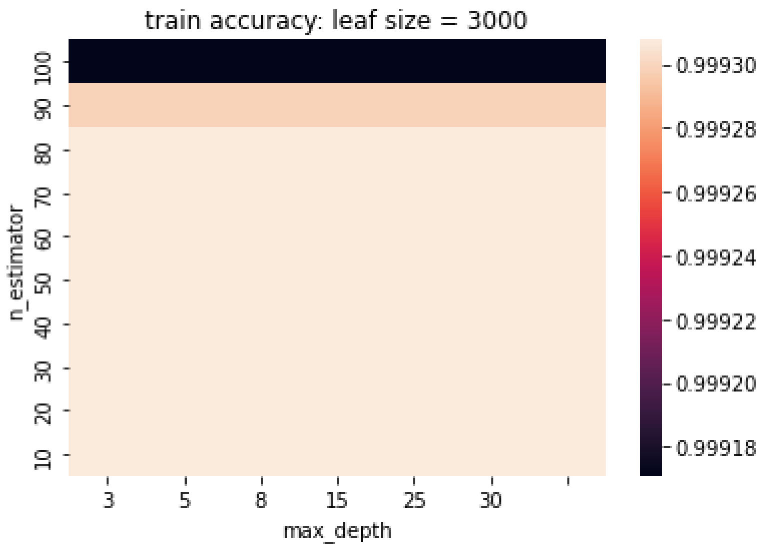
   叶子上最小样本数：[2, 3000, 7000, 10000]

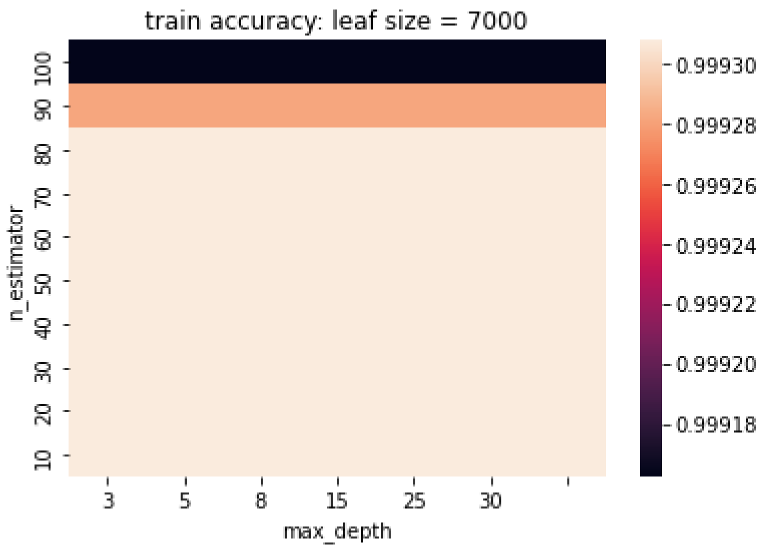
   取树木棵树：n_estimator_range=[10,20,30,40,50,60,70,80,90,100]

   树木深度：max_depth_range = [3,5,8,15,25,30,None]

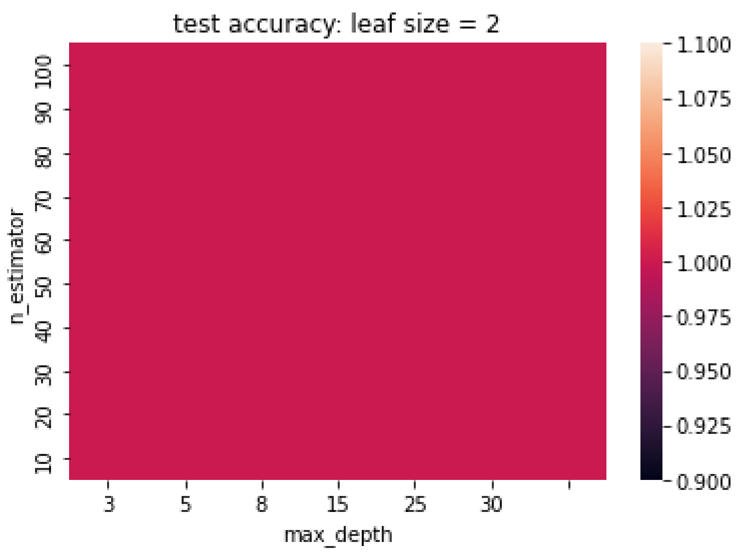
   train accuracy:

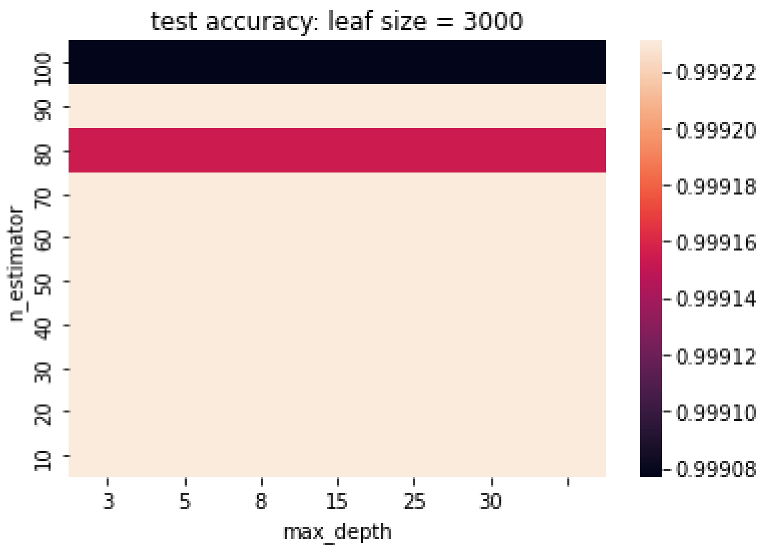


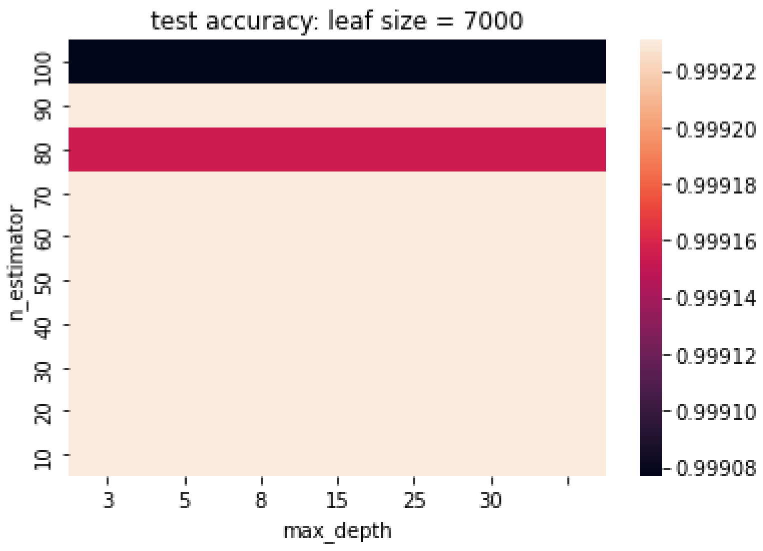


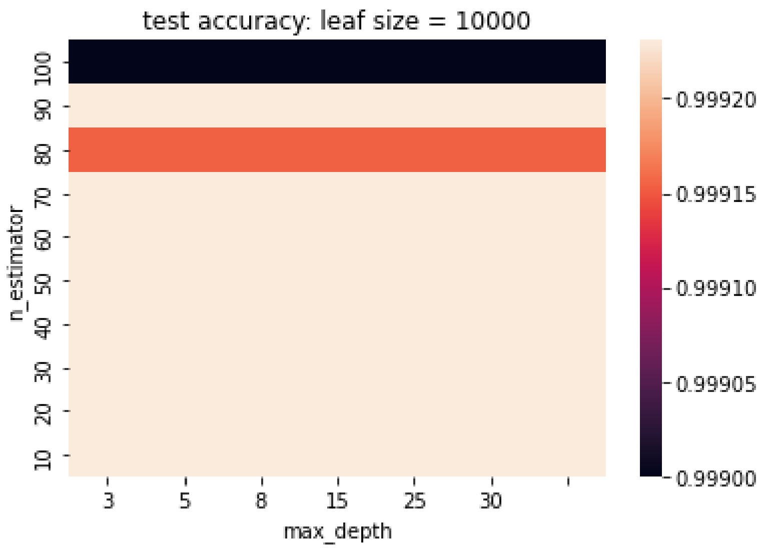


test accuracy:

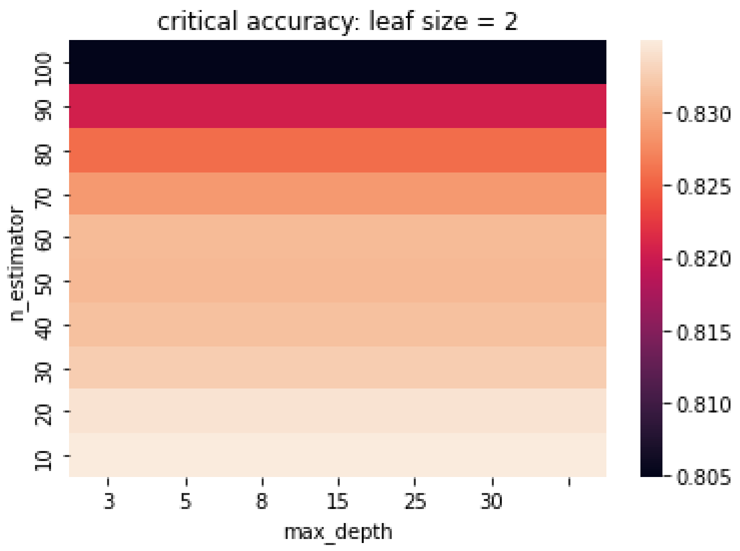


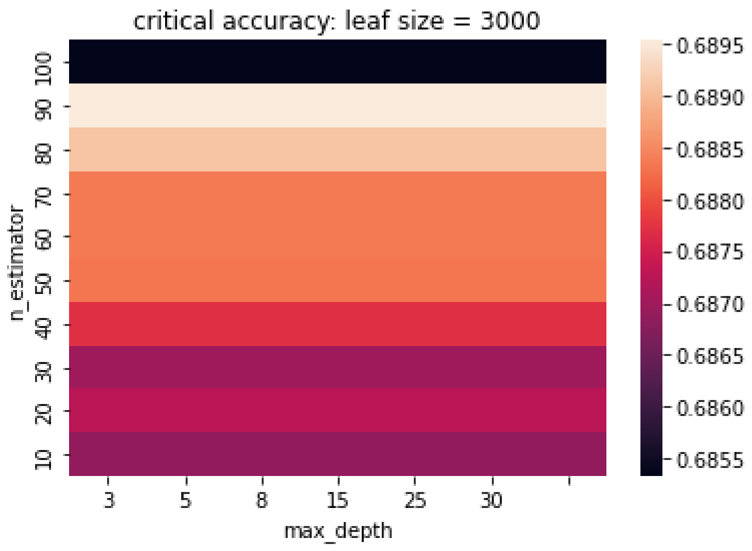


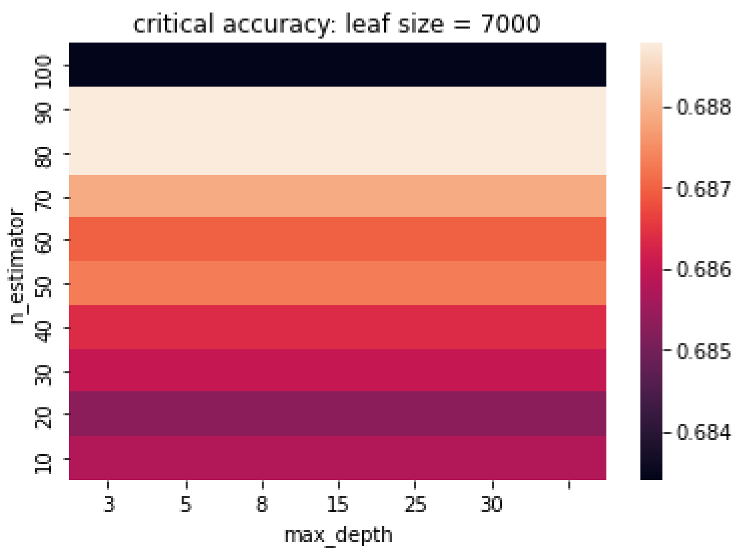


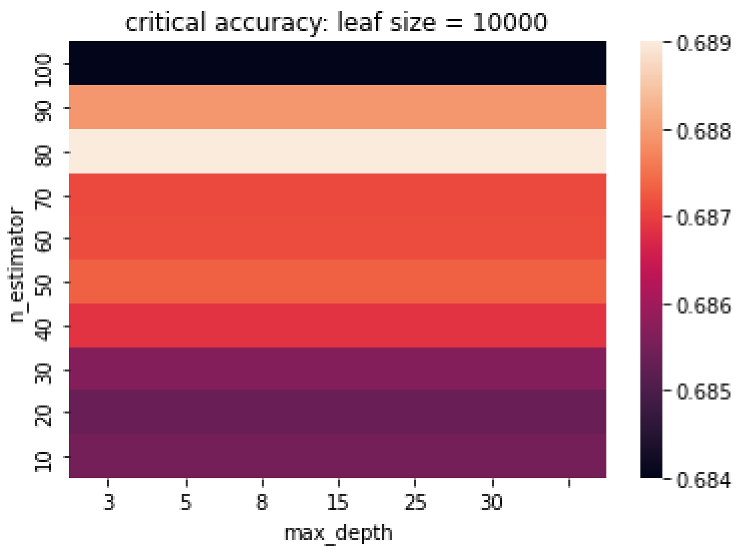


critical accuracy:









由此可知随着树木棵树的增大，准确率总体呈增大趋势；随着叶子上最小样本数的增大，准确率总体呈减小趋势；准确率不受树木深度的影响。

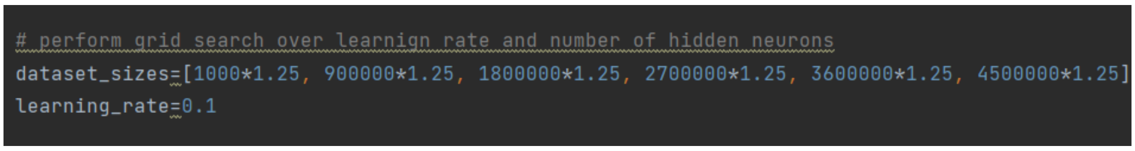
寻找OOB accuracy最大的参数：

leaf_size: 2, n_estimators: 90, max_depth: 3
leaf_size: 2, n_estimators: 90, max_depth: 5
leaf_size: 2, n_estimators: 90, max_depth: 8
leaf_size: 2, n_estimators: 90, max_depth: 15
leaf_size: 2, n_estimators: 90, max_depth: 25
leaf_size: 2, n_estimators: 90, max_depth: 30
leaf_size: 2, n_estimators: 90, max_depth: None
leaf_size: 2, n_estimators: 100, max_depth: 3
leaf_size: 2, n_estimators: 100, max_depth: 5
leaf_size: 2, n_estimators: 100, max_depth: 8
leaf_size: 2, n_estimators: 100, max_depth: 15
leaf_size: 2, n_estimators: 100, max_depth: 25
leaf_size: 2, n_estimators: 100, max_depth: 30
leaf_size: 2, n_estimators: 100, max_depth: None

用OOB准确率来选择最佳超参数，则为以上参数。即树木深度小，树木棵树大，深度任意

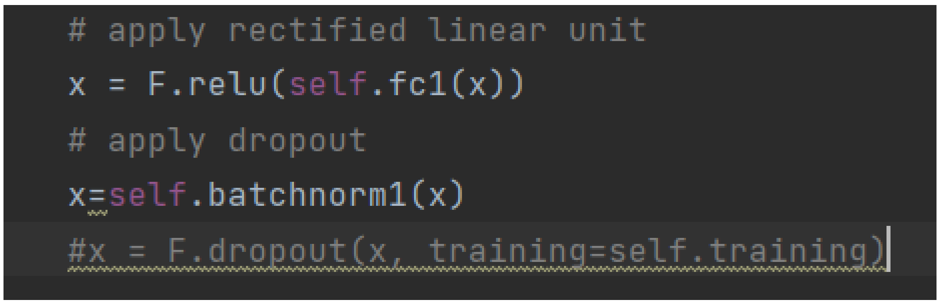
超对称

1. 取train_size = [1000, 900000, 1800000, 2700000, 3600000, 4500000]

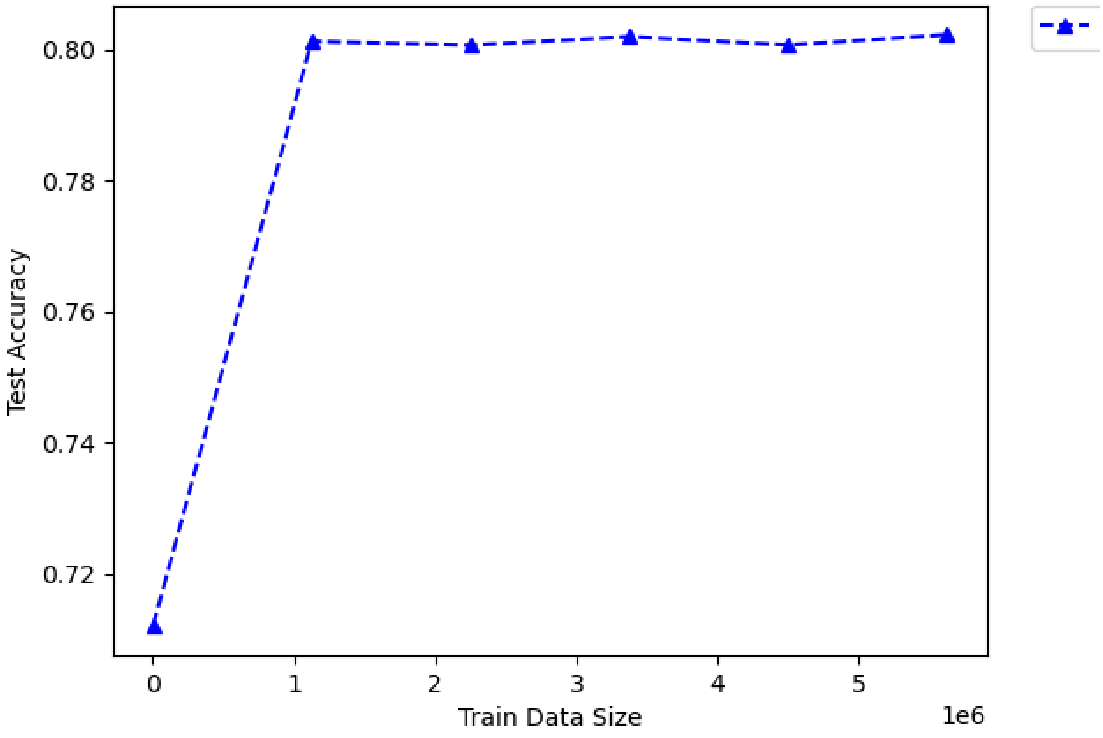


```
# perform grid search over learnign rate and number of hidden neurons
dataset_sizes=[1000*1.25, 900000*1.25, 1800000*1.25, 2700000*1.25, 3600000*1.25, 4500000*1.25]
learning_rate=0.1
```

单层隐藏层包含1000个神经元，用batchnorm防止过拟合，learning rate取0.1

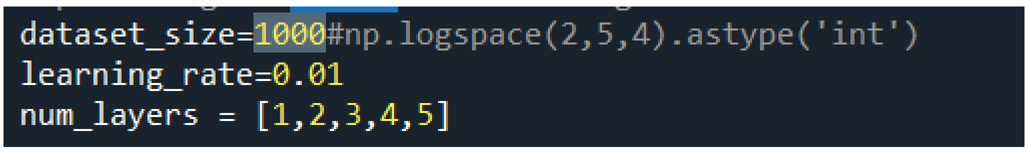


```
# apply rectified linear unit
x = F.relu(self.fc1(x))
# apply dropout
x=self.batchnorm1(x)
#x = F.dropout(x, training=self.training)
```



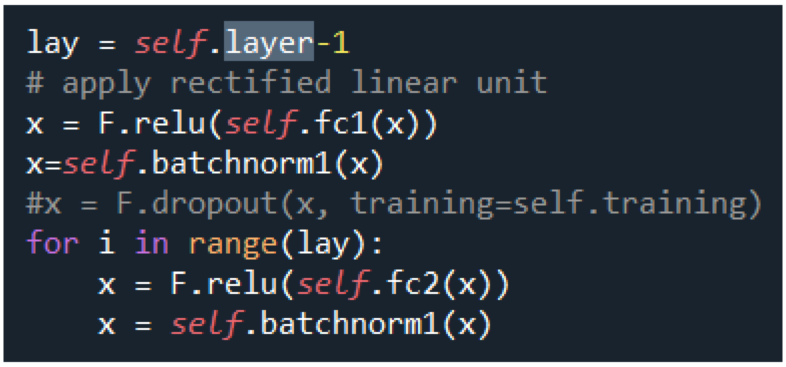
精度随datasize的增大先增大后趋于稳定

2. 取测试集与训练集样本总是为1000，learning rate取0.01



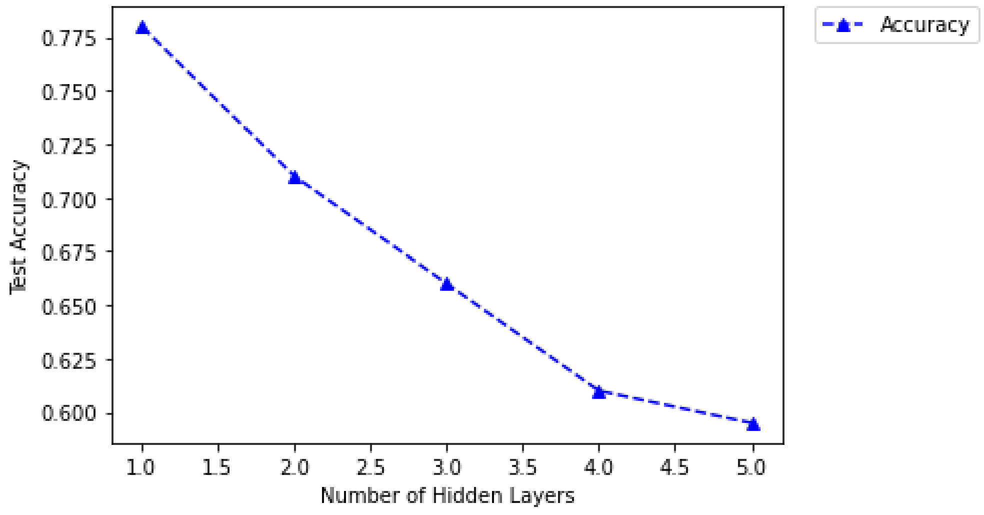
```
dataset_size=1000#np.logspace(2,5,4).astype('int')
learning_rate=0.01
num_layers = [1,2,3,4,5]
```

取层数1到5，作为参数传入model类，按照层数增加隐藏层



```python
lay = self.layer-1
# apply rectified linear unit
x = F.relu(self.fc1(x))
x=self.batchnorm1(x)
#x = F.dropout(x, training=self.training)
for i in range(lay):
    x = F.relu(self.fc2(x))
    x = self.batchnorm1(x)
```

训练结果作图如下



准确率随层数的增加而减小。