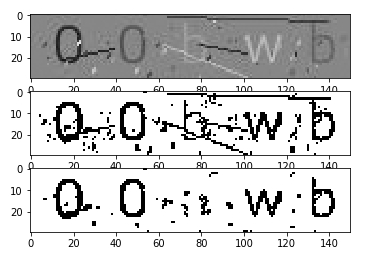
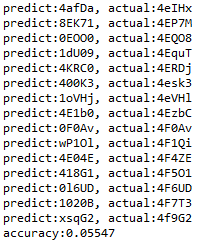
二值化，去噪





说明文档

def load\_data(file): #根据文件目录，读取文件数据

def load\_training\_set(data): #处理训练集

def load\_test\_set(data): #处理测试集

def csv\_save(answer) # 将测试集的结果保存到csv文件里

**代码流程：**

在if \_\_name\_\_ == '\_\_main\_\_'中， 先通过load\_data('train.csv')得到训练集，再通过load\_training\_set(data)处理训练集， 将训练集分成85%的训练集、 15%的验证集。用KNN模型进行训练，对近邻数n\_neighbors设多次值， 比较正确率， 选取模型训练正确率最大时的n\_neighbors，最终设定n\_neighbors=54。通过load\_data('test.csv')得到测试集，再用load\_training\_set(data)处理测试集，用刚训练好的模型对测试集进行预测，得出测试集的结果，用csv\_save() 函数将得到的结果保存到csv文件里。

**核心代码：**

核心代码是load\_data(file)函数中对数据进行预处理的部分，还有选用KNN模型进行训练的部分。

#======================数据预处理===========================#

for i in range(rows):

for j in range(cols):

if(orig\_data[i:i+1][0][j]!='?'):

orig\_data[i:i+1][0][j] = int(orig\_data[i:i+1][0][j]) #把数据从 string类型转换成int类型

else:

orig\_data[i:i+1][0][j]= np.nan #用np.nan替代数据中的‘？’，把np.nan作为缺失值处理

#用指定的方法计算数据集中的每个缺失值，然后填充数据

impute = preprocessing.Imputer(strategy='most\_frequent')

data = impute.fit\_transform(orig\_data)

#=========================训练训练集=========================#

knn = KNeighborsClassifier(n\_neighbors = 54) #选用KNN模型

knn.fit(digits\_train,labels\_train) #对数据进行训练

score = knn.score(digits\_verfct, labels\_verfct) #计算正确率

print(score) #输出训练模型的正确率

**问题解决：**

一、数据预处理问题

给出的训练集是共14列，许多列的数据不全，里面缺失的数据用“？”来表示，且缺失数据的列其中的数据都是string类型的，其他数据完整的列是int类型的。所以要填充缺失值，并把所有数据转换成相同的类型。

我选择先把所有数据遍历一遍，在遍历的过程中把除了“？”之外的数据类型都变为int类型，并用np.nan代替“？”，以方便下面用imputer类填充缺失值。我选择用缺失值所在的列中的众数来填充缺失值。

二、模型选择问题

选用什么模型来进行训练是一个问题。我最开始用softmax回归进行了训练，经过多次选取不同的循环次数和学习率，最终得到的循环次数为500、学习率为0.00001时的正确率最高，为0.332。后来我尝试了决策树、KNN、神经网络等模型。用决策树模型训练得到的正确率是0.292592，用神经网络模型训练得到的正确率是0.349074，因为KNN模型中有个不确定的参数，即最大近邻数n\_neighbors，所以我设了多个不同的n\_neighbors值来进行训练，最终当n\_neighbors=54时得到的正确率最高，为0.35064。所以最后定下的是KNN模型，其中的n\_neighbors参数取值为54。