**第二次竞赛**

**过程一：**首先使用softmax回归，通过改变迭代次数以及学习率，得到最大的准确率为0.30241；

**过程二：**然后使用决策树，通过改变决策树的深度等参数，准确率得到提升，为0.34137；

**过程三：**使用决策树，并用PCA对文件train.csv中的数据属性降维，但结果并不理想，得到准确率为0.31725。

**过程三：**最后使用随机森林，不对 train.csv中的数据属性降维，不断变换随机森林树个数，决策树最大深度和最小分裂个数等参数，得到最高的准确率为0.37662。

1. trainDataset = openFile(trainFilename)和testDataset = openFile(testFilename)

调用openFile函数，读取train.csv和test.csv文件并将每一行“?”转成数字5,去除“,”“\n”，再将每一行的处理后得数值放入list中并返回：

def openFile(filename):

datas = []

with open(filename) as f:

for line in f:

data = []

# 每一行“?”转成数字5,去除“,”“\n”

data.extend(map(int, list(line.replace('?', '5').replace(',', '')\

.replace('\n', ''))))

datas.append(data)

datas = list(datas)

return datas

2. trainData, trainLabel = splitTrainSet(trainDataset) ，其中，trainDataset是训练样本集**train.csv的**数据，调用splitTrainSettestData函数，将训练集的数据集trainDataset分成属性集trainData（即前13列的属性数据），和类别集trainLabel（即最后一列，取值从1到9）：

def splitTrainSet(trainDataset):

trainDataset = np.array(trainDataset)

trainData = trainDataset[:, :-1].tolist()

trainLabel = trainDataset[:, -1].tolist()

return trainData, trainLabel

3. sklearn中的随机森林是基于RandomForestClassifier类实现的，

因此通过from sklearn.ensemble import RandomForestClassifier来引入随机森林。

4.设置随机森林的参数clf = RandomForestClassifier(max\_depth=10, \

min\_samples\_split=10, min\_samples\_leaf=1, n\_estimators=50, max\_features=4)

其中max\_depth=10代表决策树的最大深度为10，min\_samples\_split=10表示每次分裂节点是最小的分裂个数为10，min\_samples\_leaf=1表示若某一次分裂时一个叶子节点上的样本数小于1，则会被剪枝，n\_estimators=50代表随机森林中树的个数为50，max\_features=4为总的特征数是4，

5. predictResult = list(predictResult) 将预测结果转成list，

m, = np.shape(predictResult)得到预测结果的个数，进而作为保存数据的行号，

data = pd.DataFrame(predictResult, index=range(1, m+1))

data.to\_csv('testPredict.csv')将得到的测试数据集predictResult的对应类别转化成DataFrame类型，并保存成csv文件 testPredict.csv'，然后打开该文件将第一行改为“id”，“y”。