

# 華東理工大學

## 模式识别课程报告

|        |                             |
|--------|-----------------------------|
| 学    院 | 信息科学与工程学院                   |
| 专    业 | 控制科学与工程                     |
| 班    级 | 控    制        2        班    |
| 学    号 | Y 3 0 1 8 0 6 6 7           |
| 姓    名 | 倪            佳            能 |
| 指导教师   | 赵            海            涛 |

|      |                  |
|------|------------------|
| 完成日期 | 2018 年 10 月 21 日 |
|------|------------------|

# 1 作业题目

Otto Group 商品识别

## 1.1 题目描述

Otto Group是世界上最大的电子商务公司之一，在全世界范围内，它每天会卖出数百万件商品。每件商品所属的类别分别是Class\_1～ Class\_9。对于这家公司的来说，货物供给和需求分析是非常重要的信息。现给定一些商品的多个特征，你需要设计一个算法模型来判断一个商品所属的类别。

## 1.2 题目提示

- (1) 本题是一个典型的多分类问题
- (2) 先从train.csv中提取每个商品的多项特征（feature）和类别(label)，使用feature和label进行模型训练
- (3) 商品的每个特征都是数值型，在使用SVM等分类器时注意将数据范围标准化
- (4) 商品的特征较多，在使用Decision Tree等树形分类器时，注意引入正则化项，防止模型过拟合
- (5) 题目要求预测商品属于Class\_1～ Class\_9的概率，使用深度学习模型，如MLP时，建议在神经网络的最后一层使用softmax函数，使得神经网络的输出与题目要求完全相符
- (6) 最后使用训练好的模型预测test.csv中每件商品的类别

# 2 解决方案简介

## 2.1 One-Hot编码数据处理

题目的train.csv文件给出49502个商品的93个特征以及商品实际类别，首先应将特征值编码。编码方式采用了One-Hot编码，它又称为一位有效编码，主要是采用位状态寄存器来对单个状态进行编码，每个状态都有它独立的寄存器位，并且在任意时候只有一位有效。

在实际的机器学习的应用任务中，特征有时候并不总是连续值，有可能是一些分类

值。对于这样的特征，通常我们需要对其进行特征数字化，如下面的例子：

性别：["male", "female"]

地区：["Europe", "America", "Asia"]

浏览器：["Firefox", "Chrome", "Safari", "Internet Explorer"]

对于某一个样本，如["male", "America", "Internet Explorer"]，我们需要将这个分类值的特征数字化。性别的属性是二维的，地区是三维的，浏览器则是四维的。采用One-Hot编码的方式对上述的样本“["male", "America", "Internet Explorer"]”编码，“male”则对应 [1, 0]，“America”对应着[0, 1, 0]，“Internet Explorer”对应着[0,0,0,1]。则完整的特征数字化的结果为：[1,0,0,1,0,0,0,0,1]。构建了一个非常稀疏的特征向量，保证了各种数据的离散性。

## 2.2 神经网络介绍

人工神经网络由神经元模型构成具有并行分布结构，每个神经元具有单一输出，能够与其它神经元连接存在许多输出连接方法，每种对应一个连接权系数。人工神经网络是一种具有下列特性的有向图：

- 对于每个节点*i*存在一个状态变量 $x_i$
- 从节点*j*至节点*i*，存在一个连接权系数 $w_{ij}$
- 对于每个节点*i*，存在一个阈值  $\theta_i$
- 对于每个节点*i*，定义一个变换函数  $f_i(x_i, w_{ij}, \theta_i), i \neq j$
- 对于最一般的情况，此函数取  $f_i(\sum_j w_{ij}x_j - \theta_i)$  形式

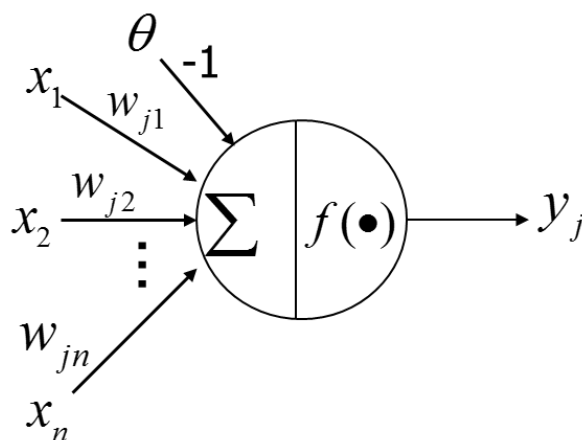


图2-1 单个神经网络示意图

将神经网络作为被辨识系统的模型，可在已知常规模型结构的情况下，估计模型的参数，它不要求建立实际系统的辨识格式，可以省去系统结构建模这一步骤。收敛速度不依赖于待辨识系统的维数，只于神经网络本身及其采用的学习算法有关。

## 2.3 Keras介绍

Keras是基于Python的深度学习库，它是一个高层神经网络API，由Python编写而成并基于Tensorflow、Theano以及CNTK后端。Keras的核心数据结构是“模型”，模型是一种组织网络层的方式。

Keras中主要的模型是Sequential模型，Sequential是一系列网络层按顺序构成的栈。其中，add()函数可以将一些网络层堆叠起来，构一个模型。此程序选取了93→128→64→32→16→9六层神经网络，中间层使用relu激活函数，根据题目提示，最后一层用softmax将预测值转化为每个分类的概率。

relu激活函数为：

$$\text{Relu}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \quad (2-1)$$

Softmax激活函数为：

$$S_i = \frac{e^i}{\sum_j e^j} \quad (2-2)$$

relu函数其实是分段线性函数，把所有的负值都变为0，而正值不变，这种操作被成为单侧抑制，可使得神经网络中的神经元具有了稀疏激活性。尤其体现在深度神经网络模型(如CNN)中。softmax函数则用于多分类过程中，它将多个神经元的输出，映射到(0,1)区间，可以将它看作“概率”的概念。

完成模型的搭建后，需要使用compile()函数来编译模型，compile接收三个参数：

**1. 优化器optimizer:** 可以在调用model.compile()之前初始化一个优化器对象，然后传入该函数（如上所示），也可以在调用model.compile()时传递一个预定义优化器名，如：SGD（随机梯度下降法）、RMSprop、Adagrad（自适应梯度算法）、Adadelat等。此程序选用Adadelat优化器。

**2. 损失函数loss:** 该参数为模型试图最小化的目标函数，它可为预定义的损失函数名，如mean\_squared\_error、mean\_absolute\_error、mean\_squared\_logarithmic\_error、hinge

等。根据题目评价的要求，此程序选用mean\_squared\_logarithmic\_error。

3. 指标列表metrics: 对分类问题，一般将该列表设置为metrics=['accuracy']。

### 3 程序实现

本程序基于图3-1的流程编写。下面具体分析程序实现过程。

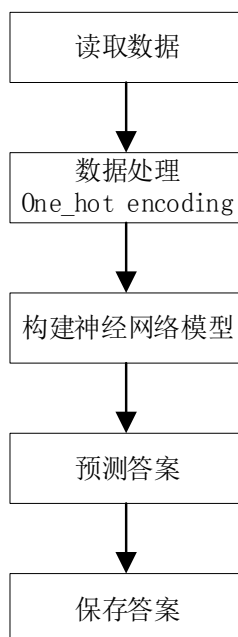


图3-1 程序流程图

#### 3.1 读取数据与数据处理

题目的train.csv文件给出49502个商品的93个特征以及商品实际类别，要完成的任务是根据93个特征预测这些商品是哪类。首先读取数据并存放，代码如下：

```
18 # ----- 读取数据 ----- #
19 train_data = pd.read_csv(open("train.csv")) # 读取train.csv的数据
20 test_data = pd.read_csv(open("test.csv")) # 读取test.csv的数据
21 ##print(train_data.info()) # 显示train.csv数据的信息
22 train_y_raw = train_data["target"] # 把商品类别的数据存入train_y_raw, 49502个商品的类别
23 ##print(train_y_raw)
24 x_label = []
25 for i in range(1, 94):
26     x_label.append("feat_{}".format(i)) # x_label存入feat数据(feat_1, feat_2, ..., feat_93)
27 train_x = np.array(train_data[x_label]) # train_x存入train.csv文件中所有商品的feat信息
28 test_x = np.array(test_data[x_label]) # test_x存入test.csv文件中所有商品的feat信息
```

其次，使用one\_hot encoding方法将每样商品的类别编码，同时存入train\_y变量。如1号商品为class\_7，则其转换后的one\_hot向量为[0 0 0 0 0 1 0 0]。代码如下：

```

32 # ----- 将train_y的数据转换成one_hot向量(9维) ----- #
33 train_y = np.zeros([len(train_y_raw), 9]) # 构建train_y矩阵(49502*9)
34 for i in range(len(train_y_raw)):
35     label_data = int(train_y_raw[i][-1]) # label_data存入了49502个商品的类别号
36     train_y[i, label_data-1] = 1 # train_y存入class的one_hot向量(class=7=000000100)
37 #print(train_x.shape, train_y.shape, test_x.shape)# (49502, 93) (49502, 9) (12376, 93)

```

在调试过程中得到train\_x变量存放49502个商品的93个特征，数据为49502×93，train\_y变量存放所有商品的类别数据（编码后），数据为49502×9（一共9种类别），test\_x变量存放未知类别的12376个商品的93个特征，用于后期预测类别的样本数据，数据为12376×93。

## 3.2 构建模型与模型训练

此程序使用Keras框架构建神经网络模型。首先创建Sequential模型，通过add()方法一个个的将layer加入模型中。由于输入的是93维数据（feature），要转换成9类的分布概率（class），这里就选取了93→128→64→32→16→9的神经网络，中间层使用relu激活函数，最后一层根据题目提示用softmax函数将预测值转化为每个分类的概率。

在训练模型之前，先通过compile来对学习过程进行配置。其中，根据Litcode的评价标准，loss（损失函数）使用 multi-class logarithmic loss, optimizer（优化器）选择'adadelata'，metrics（指标列表）中，对于分类问题，一般将该列表设置为metrics=['accuracy']。

Keras以Numpy数组作为输入数据和标签的数据类型。训练模型一般使用fit()函数，此程序epoch取250，即迭代250轮。代码如下：

```

41 # ----- 构建模型与模型训练, 93-128-64-32-16-9神经网络结构 ----- #
42 model = Sequential() # 序贯(Sequential)模型(多个网络层的线性堆叠)
43 model.add(Dense(128, input_shape=(93,), activation="relu"))
44 model.add(Dense(64, activation="relu"))
45 model.add(Dense(32, activation="relu"))
46 model.add(Dense(16, activation="relu"))
47 model.add(Dense(9))
48 model.add(Activation('softmax')) # 前几层用relu激活函数,最后一层使用softmax激活函数
49 #model.summary()
50 model.compile(loss='mean_squared_logarithmic_error',
51               optimizer='adadelata', metrics=['accuracy'])
52 model.fit(x = train_x, y = train_y, batch_size = 2048, nb_epoch = 250) # 训练模型,分批迭代样本的数据

```

## 3.3 预测答案与答案保存

模型经过train.csv文件的数据训练后就可用来预测未知类别的商品的类别。使用.predict函数，将经过处理的test数据输入模型。并且通过sampleSubmission.csv文件的格式将预测的每个商品是某个类别的概率数据存入submission.csv文件。代码如下：

```

56 # ----- 预测答案 ----- #
57 test_y = model.predict(test_x)
58 print(test_y.shape)
59 answer = pd.read_csv(open("sampleSubmission.csv"))
60 class_list = ["Class_1", "Class_2", "Class_3", "Class_4",
61 |             | "Class_5", "Class_6", "Class_7", "Class_8", "Class_9"]
62 answer[class_list] = answer[class_list].astype(float)
63
64
65
66 # ----- 答案存入submission.csv文件 ----- #
67 j = 0
68 for class_name in class_list:
69     answer[class_name] = test_y[:, j]
70     j += 1
71 answer.to_csv("submission.csv", index=False) # 不要保存索引列

```

### 3.4 测试结果

预测数据共在lintcode上提交9次，最低成绩0.75375，最高成绩0.67582，排名14/27。其中，主要调整神经网络的迭代次数，迭代次数设置在200以下时，迭代轮数越多分数越高，但其高于300之后会出现过拟合问题，因此迭代次数不宜过高。

| 文件及描述  | 状态 | 分数      |
|--|----|---------|
| <a href="#">predictions.csv</a> 2 小时前<br>250 | 成功 | 0.67582 |
| <a href="#">predictions.csv</a> 2 小时前        | 成功 | 0.69435 |
| <a href="#">predictions.csv</a> 2 小时前        | 成功 | 0.75375 |
| <a href="#">predictions.csv</a> 2 小时前        | 成功 | 0.67970 |
| <a href="#">predictions.csv</a> 2 小时前        | 成功 | 0.70602 |
| <a href="#">predictions.csv</a> 2 小时前        | 成功 | 0.70051 |

图3-2 提交分数










|    |            |         |   |    |        |
|----|------------|---------|---|----|--------|
| 6  | jxt        | 0.00070 |  | 1  | 10 个月前 |
| 7  | HaoliZhang | 0.00846 |  | 4  | 10 个月前 |
| 8  | eric074    | 0.01768 |  | 7  | 10 个月前 |
| 9  | bingege    | 0.09836 |  | 19 | 10 个月前 |
| 10 | LightGHLi  | 0.24349 |  | 3  | 10 个月前 |
| 11 | lza11111   | 0.60259 |  | 1  | 10 个月前 |
| 12 | AlphaKitty | 0.66751 |  | 14 | 15 天前  |
| 13 | HJULTX     | 0.66869 |  | 2  | 3 个月前  |
| 14 | cannonni   | 0.67582 |  | 9  | 2 小时前  |

图3-3 分数排名

代码详见 github 网站：【<https://github.com/Canonni/Python-Examples-for-Pattern-Recognition/blob/master/5.Otto%20Group%20Product%20Identification.py>】

## 4 心得体会

从初学Python到能运用Python编写模式识别课程上的例子已经经历了近一个半月，在这段时间里我认真学习模式识别课程，课后勤加练习，利用Python完成了最小二乘法示例、BP神经网络示例、SVD图像压缩、PCA数据分析等几个小实验。课上的学习加上课后的练习使我的编程水平大有进步，但在理论分析方面仍有欠缺。

此次大作业选择了Lintcode上的“Otto Group 商品识别”题目，运用深度学习包——Keras中的序贯结构的神经网络实现，程序简洁明了，稍作调试后能取得良好的预测结果。虽然在调试迭代轮数的过程中产生了神经网络过拟合现象，但是经过不断尝试还是找到了一个结果较好的迭代次数的数据。

这次的大作业让我进一步熟悉了Python，并且提高了对模式识别相关的兴趣，短短半天的编程使原来抽象的算法与理论变得清晰客观。感谢赵海涛老师的教学帮助以及同学们的交流指导！