What’s cooking 数据挖掘报告

杨展富　14301033　计算机科学与技术

用户名：DM14301033 成绩：81.627 名次：27 名(1388 人)

27	↑310	DM14301033	0.81627	21	Sun, 20 Dec 2015 08:29:31

---截图时间：2016/1/8　22:22

一，问题描述：

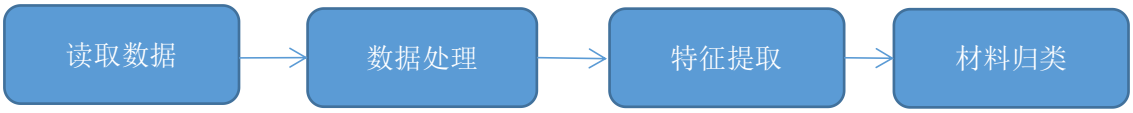
What’s cooking 是 kaggle 网站中发布的一个练习性（无奖金）竞赛，在竞赛中，kaggle 希望我们根据配方配料来分类烹饪，并给出我们训练集(train.json)与测试集(test.json)，希望我们根据训练集已有的配料与对应的国家名称的数据，来训练出测试集对应每项配方和配料的对应国家名称。

对应于我们所学过的数据挖掘课程，即是类似要求我们通过传统的训练集进行训练，然后根据训练集的特征，为测试集进行分类。

二，解题思路：

总体思路：我考虑了我们得到了的这个数据集，分为对应的国家与配料.既然有了一个国家名和对应的材料，那么，我们就可以提取出每个国家对应材料搭配等特征，利用一定的方法进行特征提取，那么，我们就可以对新有材料集合进行划分了。其中应该涉及有（1）对材料和国家数据进行读取（2）对材料数据进行相关处理（3）对训练集中每个国家的材料特征进行提取（4）对测试集中每个材料集合进行归类。

总体流程：



三，具体步骤：

（1）对材料和国家数据进行读取：

友好的是，python 对于 json 格式的数据处理是支持的，我们只要 import json 就可以对 json 进行读取了本流程中采用 json.load 进行读取操作，由于读入 json 格式的数组与一般的数组不一样，所以，python 在利用该数组查询菜式和国家时，分别需要对[‘cuisine’]和[‘ingredients’]进行访问。

（2）对材料数据进行相关处理：

由于原始材料数据是存储在数组中的字符串，所以，我们需要对其特征提取之前，需要对其进行必要的处理，使之向量化，成为我们可以操作的数字数据。而在向量化之前，我们可以有几种方法对数据进行优化，分别是化数组为大字符串类型，就如本身的[‘hot milk’],[‘salt’]化为[‘hot milk salt‘](直接通过空格连接字符串)或者[‘hot_milk salt’](同个数组的字符串用”_”进行连接，不同数组的字符串用空格连接),这样就方便于我们进行 TF-IDF 等向量化的处理。

再者，对于提取数据的优化，我们可以剔除掉很多无关的干扰项，例如，我们可以在训练集中的原料集合剔除掉测试集中没有的原料。并且，也可以在测试集中剔除掉训练集中没有的原料，以优化我们提取的数据质量。

最后在用几种方法训练得出的结果进行对比后，我还发现，每个国家菜中若有对应国家(地区，人种)的出现，会极大影响它的最终结果，所以我也尝试过先遍历直接将含上述信息的菜进行分类，结果也有微小变化，不过变动不多，只在± 0.02波动，虽然变化不明显，也能对结果有一定提高。

（3）对训练集中每个国家的材料特征进行提取：

我们使用 TF-IDF 等向量化的方法对我们提取的优化为字符串后的原料集合进行处理。处理后我们每一个国家都对应着一个向量。接下来，我们就可以使用我们已知的数据挖掘算法对我们向量化后的数据进行特征提取。

Logistic_Regression

一开始，我是用的是 logistic_regression.采用的是无序分类，对于无序分类的 logit 模型，其分析结果是以其中一类作为参照，其余各类均与参照类比较。

在Logistic regression中，所学习的系统的程为：

$$h_\theta(x) = \frac{1}{1+\exp(-\theta^T x)},$$

其对应的损失函数为：

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))\right]$$

但是在多分类模型中，logistic_regression 要进行相关的改变：

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)}=1|x^{(i)};\theta) \\ p(y^{(i)}=2|x^{(i)};\theta) \\ \vdots \\ p(y^{(i)}=k|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

对应的损失函数就改变为：

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)}=j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

起初，我在实现 logistic 时是调用 python 的 pandas 和 sklearn 包，而 logistic 等模型中，最重要的是调参数问题，一开始我并没有考虑太多，直接手动调节参数，通过 3 次迭代求得特征。

后来，我使用了 grid_search 对 logistic 的参数进行了优化，优化后成功率得到了提升。

并且，我自己要实现了一个纯手动写的 logistic_regression,可惜，效果没有那么理想，只有 77 成功率。

Grid_search:

利用网格法扫描搜索模型各待定参数值。从给定的区间取得该区间内最小目标值及其对应的最佳待定参数值。这种估值方法可保证所得的搜索解基本是全局最优解，避免了重大误差。并且在实现的时间花费上是可以接受的。

Linear_SVC：

SVC 代表支持向量分类器。作为支持向量机的一个扩展，Linear_SVC 也是一个比较有效的数据挖掘模型，类似于以'linear'为核函数的 SVC 模型，但不同于传统 SVM，它在惩罚函数与损失函数的选择中更加灵活，所以对于大规模数据更加适用。

在本项目中，我调用的是 python 中 sklearn 的 Linear_SVC()函数，并以 grid_search 进行参数自动调整，以实现对模型的回归。

Random_Forest:

随机森林法，简单来说，随机森林就是 Bagging+决策树的组合（此处一般使用 CART 树）。即由很多独立的决策树组成的一个森林，因为每棵树 之间相互独立，故而在最终模型组合时，每棵树的权重相等，即通过投票的方式决定最终的分类结果。

在本程序中，我所调用的是 sklearn 中的随机森林算法，不过效果并不理想，即使在优化后也没过 70%正确率。

Naive Bayes classifier：

朴素贝叶斯，根据经验，我认为本数据是适合贝叶斯进行处理的。根据知识，朴素贝叶斯概率模型分类器是一个条件概率模型：

$$p(C|F_1,\dots,F_n)$$

贝叶斯有以下式子：

$$p(C|F_1,\dots,F_n) = \frac{p(C)\ p(F_1,\dots,F_n|C)}{p(F_1,\dots,F_n)}.$$

而最大后验概率（MAP）决策准则。相应的分类器便是如下定义的：

$$\text{classify}(f_1,\dots,f_n) = \operatorname*{argmax}_c\ p(C=c)\prod_{i=1}^{n} p(F_i=f_i|C=c).$$

在本项目中，我尝试过自己写算法，不过没上 70%，后来采用 sklearn 自带的 MultinomialNB,则取到了比较好的效果。

Xgboost：

xgboost 是一种极限梯度上升方法，它能最优化随机森林，决策树等算法。在本项目中，我是用的网上开源 xgboost 的 library.并且，在配置环境后，参考了一些论坛中人们写过的一些类似的代码，经过筛选，修改，我最终借鉴了一个效果比较好的并行化 xgboost 的优化随机森林算法进行特征提取，以及原料分类。过程是先提取 ingredient 特征，再训练特征模型，之后再将特征模型整合进菜系模型以形成比较好的效果。

Vote：（important）

一般而言，弱分类器或多或少存在过拟合现象，即部分属性能够很好区分没部分属性则被忽略，导致这些弱分类器往往只能正确处理一部分样本，而解决不了另一部分样本。那么，解决的办法就是把这些弱分类器通过某种机制组合起来，比如投票。通过投票，单一弱分类器由于过拟合等原因无法处理特定属性的缺陷就在一定程度上被弥补了。投票的机制也有很多，最简单的就是把大家的结果放在一起做个平均。

在我操作过程中，我有好几次在 excel 中使用到了 vote 对于结果进行优化，大致分为两类：（如图 5）

1，四种模型结合：A2>B2>C2>D2

=IF(A2=B2,A2,if(A2=C2,A2,if(A2=D2,A2,if(D2=C2,C2,B2))))

2，三种模型结合：A2>B2>C2

=IF(B2=C2,B2,A2)

5	mexican	mexican	0	mexican	0	0	mexican	0
7	indian	indian	0	indian	0	0	indian	0
11	vietnames	vietnames	0	thai	1	0	vietnames	0
12	italian	french	1	southern_	1	1	italian	0
13	southern_	southern_	0	southern_	0	0	southern_	0
17	italian	italian	0	italian	0	0	italian	0
18	italian	italian	0	italian	0	0	italian	0
23	spanish	spanish	0	italian	1	0	spanish	0
26	cajun_cre	cajun_cre	0	southern_	1	0	cajun_cre	0
28	southern_	southern_	0	southern_	0	0	southern_	0

图 5

（4）对测试集中每个材料集合进行归类：

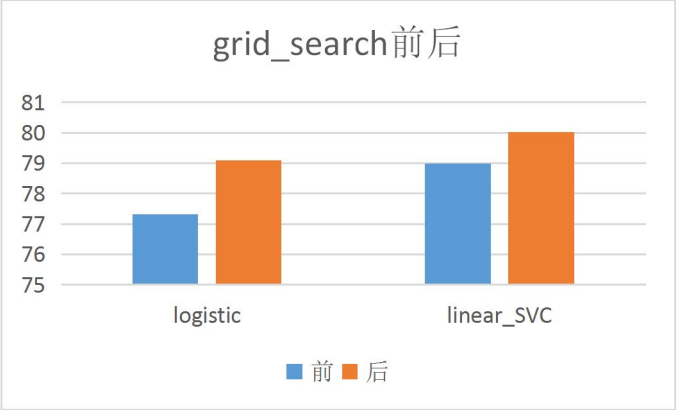
在成功从训练集中提取出了相关的特征后，我们可以开始对测试集进行分类。

类似的，我们对处理后的测试集进行简单的 function_fit 通过不同模型相同的正向计算计算出对应每个原料集合所最匹配的国家分类，最后得出结果。

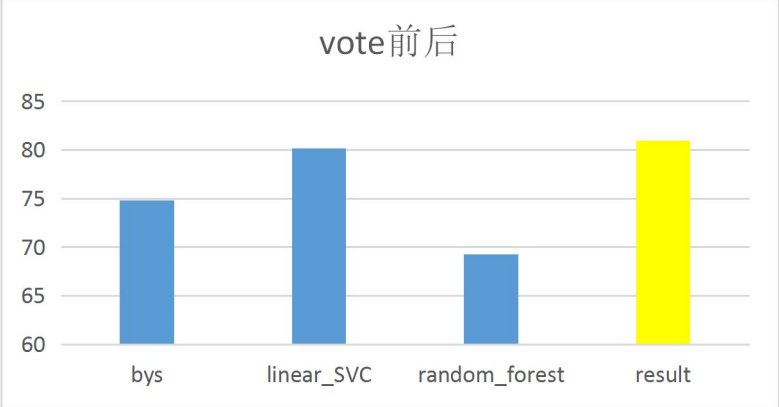
四，提升得分：

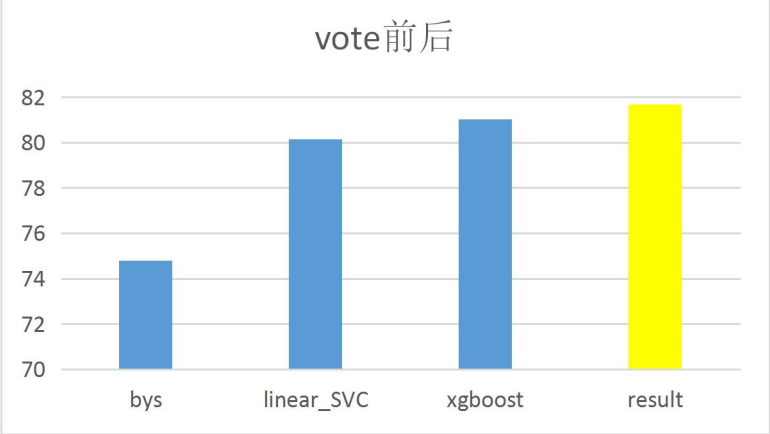
在整个项目过程我一直有优化，改变模型方法以实现对 cooking 挖掘准确性的提升，尝试过很多种模型与组合方法，得到的结果也是良莠不齐，从一开始全选 italian 的 19%，random 的 30%...logistic 掉包的 77%，不超过 70%的自我实现的朴素贝叶斯和掉包的 random_forest,到 linear_SVC 以及 grid_search 的优化，xgboost 的引入,直到最终的 vote 我才实现了当下的准确率。

Grid_search 前后：

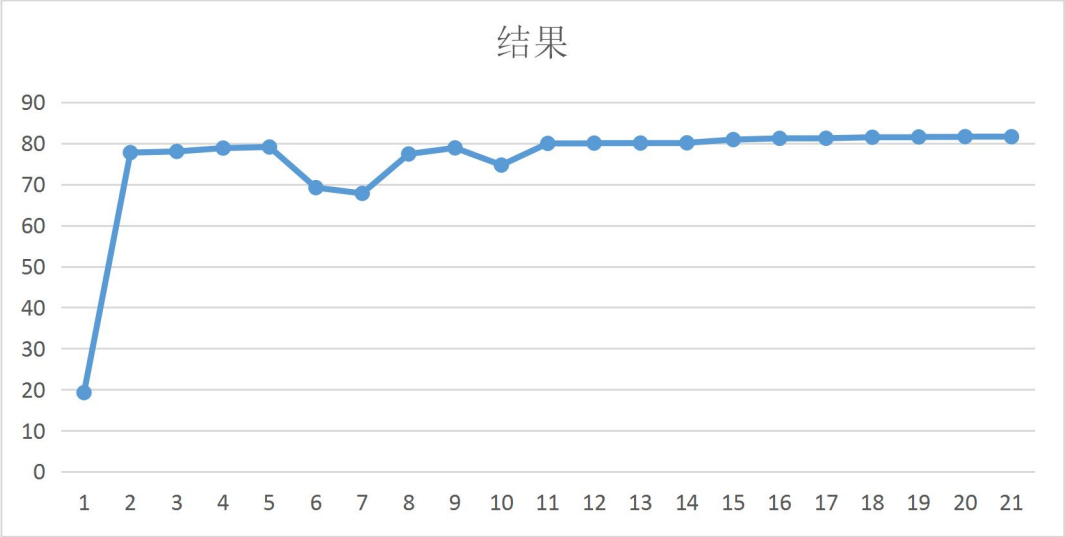


vote 前后：





总提交正确率波动：



五，最终结果：

在经过上述各种方法尝试与优化过后，我输出结果的准确率逐步提高，最终还是发现组合模型在各种模型并不高且差异比较大的情况下对正确率的提高比较大，最终在 Vote 的帮助下实现了 81.627 的成绩。

六，反思与总结：

在这次的数据挖掘项目中，我深深地发现了数据挖掘的实用性与有趣性，对于一些看似杂乱无章的数据，我们根据一些规则与方法，提取出其中的特征，规律与精髓，并发现其中的价值，这是一件多么有意义的事情啊，并且，在项目进行的过程中也让我更加了解了对数据和处理方法建立科学模型的重要性。无论在处理什么格式的数据时，都有一种方法可以从当前格式联系到我们已知的格式，在进行转化后我们就可以更加轻易地对数据特征进行提取和概括。

当然，整个过程中最重要的还是处理方法的确定。众所周知，一个高效快捷的算法是我们人人都想要的，但现实总是残酷的，往往我们在对数据进行特征提取和归类时，总是不可避免的需要以时间（空间）换取效率或者以微小效率换取更少的时间或空间。在经过我们反复权衡下，我们会选择一个最实用的模型进行求解，以减少不必要时间或者空间的浪费或者防止正确率太低。

其次，过拟合的问题也是不可忽视的，无论在使用什么方法的过程中，我们都有一个迭代个数的确定，而这，往往确定了你的数据是否过拟合或者采样不够，对于这种迭代个数的确定，我们可以使用一部分训练集进行预测试，以减低生成误差与经验误差。

总而言之，数据挖掘是一门有趣的学科，在这门学科中，可以说，我只是刚刚入门，对于今后可能遇到更加多的挑战，我希望我能够保持赤子之心，就如这次完成项目般循序渐进，不骄不躁，实现更大的进步。