Multinomial Naive贝叶斯

特征选择：所有的单词

数据处理：label coding：将11种文本分类转换成数字。如下表：

|  |  |
| --- | --- |
| Category name | Category code |
| ARTS CULTURE ENTERTAINMENT | 0 |
| BIOGRAPHIES PERSONALITIES PEOPLE | 1 |
| DEFENCE | 2 |
| DOMESTIC MARKETS | 3 |
| FOREX MARKETS | 4 |
| HEALTH | 5 |
| MONEY MARKETS | 6 |
| SCIENCE AND TECHNOLOGY | 7 |
| SHARE LISTINGS | 8 |
| SPORTS | 9 |
| IRRELEVANT | 10 |

方法和原因:

所有的单词都算作feature，用整个training set训练，没有做cross validation（这里我比较不确定，因为不知道要不要做，我不做的原因是已经决定要multinomial bayes了并且该模型并没有特定参数要进行比较取舍）。

具体代码实现用sklearn 包里的MultinomialNB() 来fit 训练集，然后对于test集合进行预测，预测正确率为72.8%。

由于推荐10个文章对每个读者，这里用的方法是用predict\_proba函数计算test集合里每个article对于每种topic的概率，然后对于每个topic，挑选出概率最大的10个，推荐给相应的读者。我们发现整个test集合里那些总共文章数很少topic，推荐的10个文章效果很差，例如对于ARTS CULTURE ENTERTAINMENT类型，整个test集合只有3个，用贝叶斯模型训练test也只有推荐出5个该类型。这种情况下如果为了凑齐10篇推荐文章而选择其他不相关文章，读者可能会觉得烦躁而停止使用这个app；同时用模型测试test集合时，发现有些对于有些topic，通过训练出的模型竟然没有推荐的文章，比如SCIENCE AND TECHNOLOGY，实际test集合里该类型有三篇(分别是9604、9722、9929)，这样的话如果没有任何推荐文章也达不到读者的要求。

所以为了用户的良好体验，这里选用的方法是：如果用模型训练出的推荐该类型数目小于5个，这时候推荐出该用户偏爱topic里概率最大的5个；推荐该类型数目大于等于5个、小于等于7个的时候，那就全部推荐这些文章，不再推荐其他文章，防止过多不相关文章。如果用模型训练出的推荐该类型文章数目大于等于8个，那么可以选用该用户偏爱topic里概率最大的10个。(8个或9个的情况，可以混入一两篇非该分类文章，同时模型挑选出的该分类文章也基本都出现在概率前十里，所以可以使用直接使用模型计算的概率前十最大的文章来近似模型预测的文章)。输出结果顺序按照概率从小到大输出。

Performance measurement of the whole test set:

|  |  |  |  |
| --- | --- | --- | --- |
| Topic name | Precision | Recall | F1 |
| ARTS CULTURE ENTERTAINMENT | 0.50 | 0.33 | 0.40 |
| BIOGRAPHIES PERSONALITIES PEOPLE | 0.75 | 0.20 | 0.32 |
| DEFENCE | 0.88 | 0.54 | 0.67 |
| DOMESTIC MARKETS | 0.00 | 0.00 | 0.00 |
| FOREX MARKETS | 0.29 | 0.08 | 0.13 |
| HEALTH | 0.86 | 0.43 | 0.57 |
| MONEY MARKETS | 0.44 | 0.87 | 0.58 |
| SCIENCE AND TECHNOLOGY | 0.00 | 0.00 | 0.00 |
| SHARE LISTINGS | 1.00 | 0.14 | 0.25 |
| SPORTS | 0.94 | 1.00 | 0.97 |

Performance measurement of recommendation set:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Topic name | Suggested articles | Precision | Recall | F1 |
| ARTS CULTURE ENTERTAINMENT | 9830,9952,9789,9703, 9604 | 0.40 | 0.67 | 0.50 |
| BIOGRAPHIES PERSONALITIES PEOPLE | 9758,9940,9854,9933,9988,  9526,9896 | 0.86 | 0.40 | 0.55 |
| DEFENCE | 9770,9987,9783,9773,9576,  9559,9616,9607,9579,9842. | 0.70 | 0.54 | 0.61 |
| DOMESTIC MARKETS | 9762,9591,9989,9640,9969 | 0.00 | 0.00 | 0.00 |
| FOREX MARKETS | 9572,9693,9584,9711,9823,  9704,9727,9743,9551, 9577 | 0.50 | 0.10 | 0.17 |
| HEALTH | 9621,9982,9661,9929,9807,  9873,9947,9617,9735,9937. | 0.80 | 0.57 | 0.67 |
| MONEY MARKETS | 9901,9985,9769,9967,9829,  9765,9586,9939,9863,9755. | 0.80 | 0.12 | 0.20 |
| SCIENCE AND TECHNOLOGY | 9722,9978,9617,9981,9644. | 0.20 | 0.33 | 0.25 |
| SHARE LISTINGS | 9601,9666,9518,9972,9867. | 0.60 | 0.43 | 0.50 |
| SPORTS | 9821,9653,9754,9836,9757, 9832,9760,9536,9568,9573. | 0.90 | 0.15 | 0.26 |

结果评估

我们重点关注Precision因为这代表了推荐给用户的文章符合预期topic的比率。

测验的结果对于test集合里文章数量较少的topic效果比较差，例如SCIENCE AND TECHNOLOGY，test里面只有3个(9604、9722、9929)，模型测试预测前五分别是9644、9617、9978、9981、9722，只有9722预测对了。更差的是DOMESTIC MARKETS, 预测正确率为0，在这种情况下，读者可能发现搜索结果和预期相差很远而不喜欢用这款app。需要更大的test集合来正确地挑选出拥有此类型的文章。对于test set里面拥有很比较多文章的topic 预测效果很好，例如sports，预测的10篇里正确率达到90 percent，还有一些用在whole test set 效果一般的topic，但用了推荐算法后Precision得到提升，例如MONEY MARKETS，从0.44到0.80。对于Precision有显著下降的SHARE LISTINGS（1.0->0.6）,模型预测只得到了唯一的结果9601，虽然是正确的预测，但只有一个预测显然也达不到读者的预期，增加到5个推荐后，Precision为0.6，说明有3个为SHARE LISTINGS类型，推荐反而更加符合读者要求。

总的来说，这种推荐没有数学依据，但比较符合人之常情(在最多推荐10个情况下比较平衡的考虑了读者的感受和文章的推荐数目)，训练的效果也相对不错。提升的话需要更大的test集合，这样推荐出的文章会更多并且更加符合读者的期望。

引用：

<https://towardsdatascience.com/text-classification-in-python-dd95d264c802>

<https://towardsdatascience.com/hyperparameter-tuning-c5619e7e6624>