文本分类思路探索

文本分类的概念

文本分类（text classification），指的是将一个文档归类到一个或多个类别的自然语言处理任务。文本分类的应用场景非常广泛，包括垃圾邮件过滤、自动打标等任何需要自动归档文本的场合。

文本分类在机器学习中属于监督学习，其流程是：人工标注文档类别、利用语料训练模型、利用模型判断文档的类别。

文本的特征提取

在向量空间模型中，表示文本的特征项可以选择文字、词、短语、甚至“概念”等多种元素，目前常用的特征提取方法有：基于文档频率的特征提取法、信息增益法、统计量法、互信息法等。

分类器的选择和训练

理论上讲，在文本特征抽取之后，就进入了常规机器学习分类模型的框架，但作为文本分类也有其特殊性，主要有以下几点：

1．自变量（词条）数量极多；

2．各自变量之间（词条）不可能完全独立；

3．大部分自变量（词条）都是干扰项，对分类没有贡献；

所以在分类模型选择上主要考虑以下几点：

1．速度–文本数据量一般比较大；

2．变量筛选能力–能够从大部分是无效变量的情况下筛选出有效变量；

3．容错性–分类模型是建立在特征抽取的基础上，特征抽取过程本身不可避免的带来部分信息差错；

4．共线容忍度–词条之间不可能相互独立，很多模型都有变量的独立性假设。

基于上面两个方面的考虑，文本分类的模型通常使用朴素贝叶斯、svm两个模型。

标准化评测

评测指标通常是四项：

$P = \frac{TP}{TP+FP}$：精确率

$R = \frac{TP}{TP+FN}$：召回率

$F_1 = \frac{2*P*R}{P+R}$：调和平均

running_time：训练时间

核心代码实现：

1. 准备文本分类语料库

```
from pyhanlp import *
corpus_path = r'/Users/kitty/Work/Projects/text_mining/data/xxxx'
MemoryDataSet = JClass('com.hankcs.hanlp.classification.corpus.MemoryDataSet')
dataSet = MemoryDataSet()
dataSet.load(corpus_path)
allClasses = dataSet.getCatalog().getCategories()
print("标注集：%s" % (allClasses))
for document in dataSet.iterator():
    print("第一篇文档的类别：" + allClasses.get(document.category))
    break
```

2. 准备分词器

采用的是2016年An Efficient Chinese Text Classifier这篇论文结果，这篇论文结论是对于中文，不分词反而取得更好的分类效率，代价是增加了模型训练的复杂度

```
BigramTokenizer = JClass('com.hankcs.hanlp.classification.tokenizers.BigramTokenizer')
HanLPTokenizer = JClass('com.hankcs.hanlp.classification.tokenizers.HanLPTokenizer')
BlankTokenizer = JClass('com.hankcs.hanlp.classification.tokenizers.BlankTokenizer')
tokenizer = BigramTokenizer()
```

3. 准备训练集

```
FileDataSet = JClass('com.hankcs.hanlp.classification.corpus.FileDataSet')
training_corpus = FileDataSet().setTokenizer(tokenizer).load(corpus_path, "UTF-8", 0.9)
```

4. 卡方特征选择

这个方法和相应的参数都被隐藏的模型中，只需知道分词后需要进行卡方特征选择。

5. 准备分类器

```
NaiveBayesClassifier =
JClass('com.hankcs.hanlp.classification.classifiers.NaiveBayesClassifier')
```

```
LinearSVMClassifier =
JClass('com.hankcs.hanlp.classification.classifiers.LinearSVMClassifier')
model_class = LinearSVMClassifier
```

6. 训练模型

```
IOUtil = SafeJClass('com.hankcs.hanlp.corpus.io.IOUtil')
model_path = r'xxxx'
def train_or_load_classifier(model_class, model_path, training_corpus):
    classifier = model_class()
    model_path += classifier.getClass().getSimpleName() + '.ser'
    if os.path.isfile(model_path):
        print(model_path)
        return model_class(IOUtil.readObjectFrom(model_path))
    classifier.train(training_corpus)
    model = classifier.getModel()
    IOUtil.saveObjectTo(model, model_path)
    return model_class(model)


classifier = train_or_load_classifier(model_class, model_path, training_corpus)
```

7. 评测训练效果

```
Evaluator = JClass('com.hankcs.hanlp.classification.statistics.evaluations.Evaluator')


def evaluate(classifier, corpus_path, tokenizer):
    testing_corpus = MemoryDataSet(classifier.getModel()).load(corpus_path, "UTF-8",
-0.1)
    result = Evaluator.evaluate(classifier, testing_corpus)
    print(classifier.getClass().getSimpleName() + "+" +
tokenizer.getClass().getSimpleName())
    print(result)


evaluate(classifier, corpus_path, tokenizer)
```

评测结果:

LinearSVMClassifier+BigramTokenizer

P	R	F1	A	
94.17	97.00	95.57	98.20	教育
99.00	99.00	99.00	99.60	汽车
98.98	97.00	97.98	99.20	健康
98.02	99.00	98.51	99.40	军事
100.00	98.00	98.99	99.60	体育
98.03	98.00	98.02	98.00	avg.

data size = 500, speed = 16666.67 doc/s