实战案例6：垃圾短信检测

作者： Robin
日期： 2018/03
提问： 小象问答
数据集来源： kaggle

1. 案例描述

垃圾短信是指未经用户同意向用户发送的用户不愿意收到的短信息，或用户不能根据自己的意愿拒绝接收的短信息，主要包含以下属性：1. 未经用户同意向用户发送的商业类、广告类等短信息；2. 其他违反行业自律性规范的短信息。垃圾短信泛滥，已经严重影响到人们正常生活、运营商形象乃至社会稳定。

本案例中尝试通过文本处理及机器学习对垃圾短信进行检测。

2. 数据集描述

- Kaggle提供的数据集
- 数据字典
  - **text_type**: 短信类型，spam->垃圾短信，ham->非垃圾短信
  - **text**: 短信内容

3. 任务描述

- 使用文本处理，特征提取及机器学习方法检测垃圾短信

4. 主要代码解释

- 代码结构

```
lect09_proj
├── data
    ├── spam.csv    # 原始数据集
    ├── proc_spam.csv    # 文本预处理后的数据集
├── main.py      # 主程序
├── utils.py      # 工具文件，包含文本预处理、特征工程等
├── config.y     # 配置文件
├── lect09_proj_readme.pdf  # 案例讲解文档
```

- **utils.py**

使用Series的map()函数将text_type根据字典转换为标签

```
def prepare_data():
    ...
    # 添加标签
    all_data['label'] = all_data['text_type'].map(config.text_type_dict)
    ...
```

- **utils.py**

对每个文本进行预处理操作，包括：

1. 去除标点符号
2. 分词
3. 词形归一化
4. 去除停用词

预处理后的文本为“空格”隔开单词的字符串，如：原始文本为 'U dun say so early hor... U c already then say...'，处理后结果为'u dun say early hor u c already say'

```
def prepare_data():
    ...
    # 添加预处理后的文本
    all_data['proc_text'] = all_data['text'].apply(preprocess_text)
    ...
```

- **utils.py**

在该案例中使用了两种文本特征：

1. TF-IDF
2. 词袋模型

最终的特征为两种特征的合并。注意，为节省空间，sklearn中的文本特征提取结果为稀疏矩阵，如果需要对特征进行操作，需要使用to_array()将其转换为普通ndarray

```
def do_feature_engineering(train_data, test_data):
    ...
    # TF-IDF特征提取
    tfidf_vectorizer = TfidfVectorizer()
    train_tfidf_feat = tfidf_vectorizer.fit_transform(train_proc_text).toarray()
    test_tfidf_feat = tfidf_vectorizer.transform(test_proc_text).toarray()

    # 词袋模型
    count_vectorizer = CountVectorizer()
    train_count_feat = count_vectorizer.fit_transform(train_proc_text).toarray()
    testcount_feat = count_vectorizer.transform(test_proc_text).toarray()
    ...
```

5. 案例总结

- 该项目通过使用常用的文本预处理及特征提取操作，实现了垃圾短信的检测，包含了如下内容:
    - 文本预处理：分词，词形归一化，去除停用词
    - 文本特征提取：TF-IDF，词袋模型（词频统计）
    - 朴素贝叶斯模型

6. 课后练习

- 修改代码，对比使用一种特征和多种特征对模型性能的影响

参考资料

1. sklearn文本处理
2. sklearn文本特征
3. NLTK