E-mail Classification

张霖，*学号2120151063*，杨冰，*学号2120151052*

***摘要*—本文采用逻辑回归模型进行垃圾邮件分类。邮件分类属于二分类问题，而按照用户兴趣的邮件分类则属于多分类问题。本文首先将数据集进行预处理，提取元数据；然后对数据进行分词处理，提取出关键词，去除无意义的词；然后将数据划分为训练集和测试集，利用逻辑回归模型进行训练，并在测试集上测试，与 Groundtruth 进行对比，测试准确性。实验表明，本文所采用的提取邮件的关键词频度特征，使用逻辑回归模型进行分类的方法，实现较简单，分类效果较优。**

***关键词*—数据挖掘，邮件分类，逻辑回归**

I. 简介

随着互联网应用的不断深入，电子邮件已成为人们日常生活中不可或缺的一部分。而各种广告、产品销售、交友等垃圾邮件极大地占用用户空间、浪费用户时间、消耗网络带宽，如何有效过滤垃圾邮件已成为当前研究的热门课题。与此同时，知识管理（KM）在大数据时代之下要求对大量的用户数据进行管理、统计与分析。所以，用户邮件的分类是一类最基本的问题数据。

本实验拟利用已有的邮件数据库，并通过以下步骤进行设计和实验：

1. 在数据集上对数据进行类型标注，如个人邮件、公务邮件、垃圾邮件、广告邮件等，建立 Groundtruth。
2. 对数据进行预处理。从数据中提取出元数据（发件、收件、抄送、主题）等。
3. 对数据进行分词处理。选取每个数据的关键词集合，去除无意义的词。
4. 利用关键词，基于某种算法建立每个数据的特征向量。
5. 设计分类算法。
6. 划分数据集为训练集和测试集。
7. 依据设计的分类算法，使用训练集对分类模型进行训练。
8. 使用测试集在分类模型上进行测试，并与 Groundtruth 进行比较，记录算法的准确度。
9. 重复训练和测试过程，对算法性能进行评价。

本文为北京理工大学数据挖掘作业项目报告。如有疑问请联系作者，联系方式为：

张霖，杨冰，中心教学楼 823（e-mail：cs_zhanglin@163.com; bityangbing@bit.edu.cn）。

II. 问题陈述

垃圾邮件分类属于二分类问题，按照用户兴趣的邮件分类甚至按照发信人进行邮件分类则属于多分类问题。目前对于邮件分类问题有如下几种实现方案，其一是使用深度神经网络作为判别模型直接进行邮件分类，该方法效果优良。其中深度信念网络结合受限玻尔兹曼机产生一种高效快速地机器学习算法，可以对神经网络快速训练，以及快速测试分类。其二是使用回归模型例如逻辑回归作为分类器，该方法一般用于二分类问题，也可以用于多分类情况。其三是使用朴素贝叶斯分类器，计算并比较邮件 X 属于 Ci 类的后验概率，将邮件分至概率大的类中，该方法优点在于用户可以通过手动调整邮件所属的类别，从而动态改变邮件频度，实现动态分类。本文所选用的是提取邮件的关键词频度特征，使用逻辑回归模型进行分类的方法。该方法实现较简单，分类效果较优。

III. 技术方案

A. *算法介绍*

逻辑回归即 Logistic 回归，Logistic 回归与多重线性回归实际上有很多相同之处，最大的区别就在于它们的因变量不同，其他的基本都差不多。正是因为如此，这两种回归可以归于同一个家族，即广义线性模型（generalizedlinear model）。

这一家族中的模型形式基本上都差不多，不同的就是因变量不同。

如果是连续的，就是多重线性回归；

如果是二项分布，就是 Logistic 回归；

如果是 Poisson 分布，就是 Poisson 回归；

如果是负二项分布，就是负二项回归。

Logistic 回归的因变量可以是二分类的，也可以是多分类的，但是二分类的更为常用，也更加容易解释。所以实际中最常用的就是二分类的 Logistic 回归。垃圾邮件分类可以使用 Logistic 回归完成，按照用户姓名分类邮件的问题也可以使用 Logistic 回归对每一类计算概率，最后选择概率最大的类即可。

Logistic 回归的主要用途：

寻找危险因素：寻找某一疾病的危险因素等；

预测：根据模型，预测在不同的自变量情况下，发生某病或某种情况的概率有多大；

判别：实际上跟预测有些类似，也是根据模型，判断某人属于某病或属于某种情况的概率有多大，也就是看一下这个人有多大的可能性是属于某病。

Logistic 回归主要在流行病学中应用较多，比较常用的情形是探索某疾病的危险因素，根据危险因素预测某疾病发生的概率，等等。例如，想探讨胃癌发生的危险因素，可以选择两组人群，一组是胃癌组，一组是非胃癌组，两组人群肯定有不同的体征和生活方式等。这里的因变量就是是否胃癌，即“是”或“否”，自变量就可以包括很多了，例如年龄、性别、饮食习惯、幽门螺杆菌感染等。自变量既可以是连续的，也可以是分类的。

B. *逻辑回归分类器算法步骤*

Logistic 回归虽然名字里带“回归”，但是它实际上是一种分类方法，主要用于两分类问题（即输出只有两种，分别代表两个类别），首先构造预测函数 h：预测函数需要借助 Sigmoid 函数，其形式为：

$$g(z)=\frac{1}{1+e^{-z}} \tag{1}$$

构造预测函数为：

$$h_\theta(x)=g(\theta^T x)=\frac{1}{1+e^{-\theta^T x}} \tag{2}$$

其中 θ 为：

$$\theta_0+\theta_1\theta_1+,...,+\theta_n x_n=\sum_{i=1}^{n}\theta_i x_i=\theta^T x \tag{3}$$

那么，对于输入 x 分类结果为类别 1 和类别 0 的概率分别为：

$$P(y=1\mid x;\theta)=h_\theta(x) \tag{4}$$

$$P(y=0\mid x;\theta)=1-h_\theta(x) \tag{5}$$

对于本例多分类问题，可对于所有的备选 item 分类，找出其 $h_\theta(x)$最大的分类，最终逐步获取最适应的 item。

第二步为构造损失函数 J：先使用最大似然估计推导损失函数 Cost，则 J 函数为：

$$\begin{aligned}J(\theta)&=\frac{1}{m}\sum_{i=1}^{n}\mathrm{Cost}(h_\theta(x_i),y_i)\\&=-\frac{1}{m}\left[\sum_{i=1}^{n}y_i\log h_\theta(x_i)+(1-y_i)\log(1-h_\theta(x_i))\right]\end{aligned} \tag{6}$$

最后为寻找回归参数 θ。使用梯度下降法求 θ，梯度下降流公式为：

$$\frac{\delta}{\delta\theta_j}J(\theta)=\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)x_i^j \tag{7}$$

则更新过程为：

$$\theta_j:=\theta_j-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)x_i^j \tag{8}$$

C. *邮件分类系统设计*

我们的邮件分类系统大致步骤如下：

（1） 首先训练分类器，提取文本的关键词频度特征。

（2） 使用关键词频度训练逻辑回归分类器。

（3） 交叉验证分类器效果。

（4） 测试数据，观察分类结果。

系统流程图大致如下：

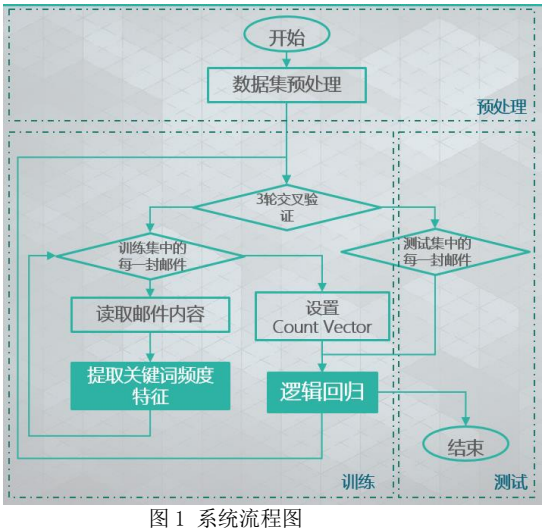


图 1 系统流程图

IV. 方法实现

A. *提取关键词频度特征*

核心代码如下：

```
    or filename in os.listdir('enron/ham/'):
        lines = [line.rstrip('\n') for line in open('enron/ham/'+filename)]
        tempsubject = lines[0][9:].lower()
        cleaned = [e for e in tempsubject.split(' ') if e.isalnum()]
        cleaned = [word for word in cleaned if word not in stop_words and word!='']
        cleaned = ' '.join(cleaned)
        if len(cleaned)>0:
            ham_subject.append(cleaned)
        else:
            ham_subject.append(' ')

        temp_ham_text = ' '
        for line in lines[1:]:
            temptext = line.lower()
            cleaned = [e for e in temptext.split(' ') if e.isalnum()]
            cleaned = [word for word in cleaned if word not in stop_words and word!='']
            cleaned = ' '.join(cleaned)
            if len(cleaned)>0:
                temp_ham_text+=cleaned+' '
        ham_text.append(temp_ham_text.strip())

        for i in range(len(ham_subject)):
        ham_subj_text.append(ham_subject[i]+' '+ham_text[i])
```

对于训练集中的每一封邮件，我们先读取邮件内容，之后将邮件内容分为标题 tempsubject 和正文 temptext 两部分。对于每一部分文本，以正文为例，我们都提取其中

所有的英文单词存储在 temptext 中，之后去掉标点符号以及 the、of 等虚词，存储在 cleaned 中。之后统计每一篇文章每一个单词的出现个数，并将标题 tempsubject 和正文 temptext 的单词个数特征拼接起来。

将标题 tempsubject 和正文 temptext 分开统计再拼接的原因是，经由我们实验发现，这样做比之直接统计全文单词出现频率，正确率更高。

B. *训练逻辑回归模型*

核心代码如下：

```
clf = LogisticRegression()

clf.fit(counts,data['class'])
```

其中 counts 为统计好的训练集全部特征向量。之后我们使用交叉验证方法计算出了训练好的逻辑回归模型正确率：

```
scores = cross_val_score(clf, counts, data['class'], cv=3)
precisions = cross_val_score(clf, counts, data['class'], cv=3, scoring='precision_weighted')
recalls = cross_val_score(clf, counts, data['class'], cv=3, scoring='recall_weighted')
f1s = cross_val_score(clf, counts, data['class'], cv=3, scoring='f1_weighted')
```

C. *测试数据*

核心代码如下：

```
print clf.predict_proba(transformed)

print clf.predict(transformed)
```

提取待测试 E-mail 的特征之后，直接使用定义好的逻辑回归类对象 clf 预测即可，第一行输出的是预测概率以及误差，第二行输出分类标签。

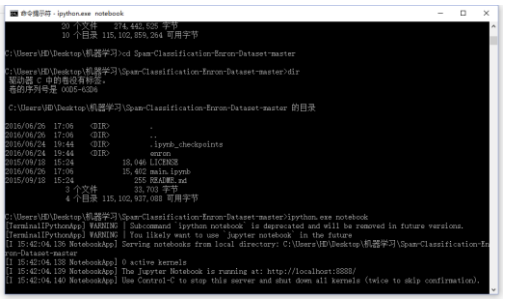
V. 实验结果

A. *环境配置*

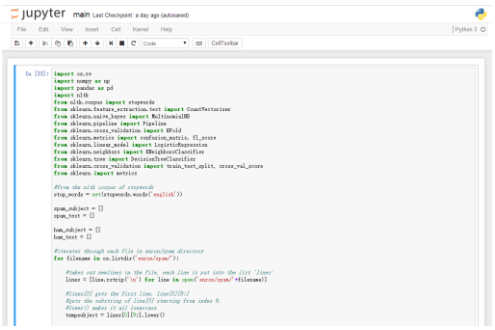
系统：　　Windows10
开发语言：　　Python2.7.11
IDE：　　Ipython Jupyter
环境：　　Anaconda 数据分析环境
工具包：　　Sklearn，Nltk，pandas，numpy

B. *项目运行*

（1）下载安装 Python2.7.11。
（2）下载安装 Anaconda3-4.0.0
（3）安装 Ipython Jupyter
在 cmd 中输入 pip install Jupyter，安装 python 的开发 IDE Jupyter。
（4）安装依赖环境包
在 cmd 中输入 pip install Sklearn,pip install nltk，pip install pandas，pip install numpy。依次安装完成。
（5）项目运行
在命令行中输入 cd 命令跳转至代码所在文件夹中，即 E-mail-classification--Spam-filter—master 中。
之后输入 Ipython.exe notebook 命令，启动 python 开发 IDE，此时会在 web 上建立一个 python 服务器，用来编辑 python 代码以及编译运行。部分过程如下图 2 所示：



(a)



(b)

图 2 项目运行

C. *二分类结果*

分类垃圾邮件的问题中，手动将训练数据分为两类：垃圾邮件 0 与非垃圾邮件 1，使用训练数据进行训练。交叉验证后所得回归模型的准确率预测率和召回率分别为：0.97,0.97,0.97，如下图：

```
clf = LogisticRegression()

#3-fold cross-validation of the model, 5-fold is more often used but if 3-fold performs well,
#then your model is golden.
#clf,counts,data['class'] is just the model,data/matrix,class-labels for each row in the matrix
#cv=3 is the number of folds in k-fold cross-validation (cross-validation = cv)
scores = cross_val_score(clf, counts, data['class'], cv=3)

#print out the average of the 3 values from the 3-fold cross-validation
print 'Accuracy:  ', np.mean(scores)

#can make precision/recall/f1/etc. with different scoring
precisions = cross_val_score(clf, counts, data['class'], cv=3, scoring='precision_weighted')
print 'Precision: ', np.mean(precisions)
recalls = cross_val_score(clf, counts, data['class'], cv=3, scoring='recall_weighted')
print 'Recall:    ', np.mean(recalls)
f1s = cross_val_score(clf, counts, data['class'], cv=3, scoring='f1_weighted')
print 'F1:        ', np.mean(f1s)
```

```
Accuracy:   0.976677941962
Precision:  0.976783846269
Recall:     0.976677941962
F1:         0.976674495815
```

图 2 二分类回归模型准确率和召回率

效果较优异。使用测试邮件如下进行测试，发现该邮件为垃圾邮件，逻辑回归分类器正确将其分类为 0，即垃圾邮件类。

分类结果如下图：

```
#gotta clean it the same way I did with the training examples for it to work properly
temptext = line.lower()
cleaned = [e for e in temptext.split(' ') if e.isalnum()]
cleaned = [word for word in cleaned if word not in stop_words and word!='']
cleaned = ' '.join(cleaned)
transformed = count_vectorizer.transform([cleaned])

#make the model, train the model, make a prediction.
clf = LogisticRegression()
clf.fit(counts,data['class'])

#probabilities for choosing a class. first in the array is 0's prob, next is 1's prob.
#picks the one with the highest prob.
print clf.predict_proba(transformed)

#spits out the highest prob prediction.
print clf.predict(transformed)

#tada
```

```
[[  9.99542132e-01   4.57867503e-04]]
[0]
```

图 3 分类结果

D. 多分类结果

按照用户姓名分类邮件的问题中，手动将训练数据分为四类：用户 0，用户 1，用户 2，和用户 3，使用训练数据进行训练。交叉验证后所得回归模型的准确率预测率和召回率分别为：0.76, 0.80, 0.66，如下图：

```
clf = LogisticRegression()

#3-fold cross-validation of the model, 5-fold is more often used but if 3-fold performs well,
#then your model is golden.
#clf, counts, data['class'] is just the model, data/matrix, class-labels for each row in the matrix
#cv=3 is the number of folds in k-fold cross-validation (cross-validation = cv)
scores = cross_val_score(clf, counts, data['class'], cv=3)

#print out the average of the 3 values from the 3-fold cross-validation
print 'Accuracy: ', np.mean(scores)

#can make precision/recall/f1/etc. with different scoring
precisions = cross_val_score(clf, counts, data['class'], cv=3, scoring='precision_weighted')
print 'Precision: ', np.mean(precisions)
recalls = cross_val_score(clf, counts, data['class'], cv=3, scoring='recall_weighted')
print 'Recall:    ', np.mean(recalls)
f1s = cross_val_score(clf, counts, data['class'], cv=3, scoring='f1_weighted')
print 'F1:        ', np.mean(f1s)
```

```
Accuracy:  0.761563258452
Precision:  0.809652145832
Recall:    0.666325485125
F1:        0.763486521965
```

图 4 多分类准确率和召回率

E. 实验结果分析

可以看出，二分类邮件问题下的训练效果比多分类邮件问题下的训练效果好很多，这说明逻辑回归模型属于一个二分类模型，虽然可以用于多分类问题，但是效果总是不尽如人意。

所以实验的改进可以考虑更换分类模型为多重线性回归模型，可以分类任意多类。同时可以考虑增加提取文本的特征，组成更长的特征向量，如果特征向量维数过高影响实验效率，可以考虑 PCA 降维等方法。

参考文献

[1] 陈治平, 谭义红, 赵碧海. 基于用户行为的邮件分类算法[M]. 计算机应用:陈治平, 2014. 1369-1372

[2] 1AHMAD, A, AL, SALLAB, 2MOHSEN, A, RASHWAN. E-MAIL CLASSIFICATION USING DEEP NETWORKS[J]. Journal of Theoretical and Applied Information Technology, 2012, 37(2): 241-251

[3] Hong-Jun, Zhang. Design and Implementation of an Intelligent E-mail Classification System[C]. International Journal of Research and Reviews in Computer Science:Hong-Jun Zhang, 2011.

[4] Marsono, M, N. Distributed layer-3 e-mail classification for spam control[C]. Canadian Conference on Electrical and Computer Engineering:Marsono, M.N, 2007.