数据挖掘
E-mail
Classification

小组成员：

张　霖 2120151063

杨　冰 2120151052

1. 算法背景

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。

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。

2. 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。

3. 逻辑回归分类器算法步骤

Regression 问题的常规步骤为：

（1）寻找 h 函数（即 hypothesis）。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），所以利用了 Logistic 函数（或称为 Sigmoid 函数），函数形式为：

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

对于线性边界的情况，边界形式如下：

$$\theta_0+\theta_1x_1+,\ldots,+\theta_nx_n=\sum_{i=1}^{n}\theta_ix_i=\theta^Tx$$

构造预测函数为：

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

（2）构造 J 函数（损失函数）。Cost 函数和 J 函数如下，它们是基于最大似然估计推导得到的。

$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) \ if \ y = 1 \\ -\log(1 - h_\theta(x)) \ if \ y = 0 \end{cases}$$

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

（3）想办法使得 J 函数最小并求得回归参数（θ）。θ 使用梯度下降法求得，更新过程如下：

$$\theta_j := \theta_j - \alpha \frac{\delta}{\delta_{\theta_j}} J(\theta)$$

$$\frac{\delta}{\delta_{\theta_j}} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left(y_i \frac{1}{h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) - (1 - y_i) \frac{1}{1 - h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) \right)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i \frac{1}{g(\theta^T x_i)} - (1 - y_i) \frac{1}{1 - g(\theta^T x_i)} \right) \frac{\delta}{\delta_{\theta_j}} g(\theta^T x_i)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i \frac{1}{g(\theta^T x_i)} - (1 - y_i) \frac{1}{1 - g(\theta^T x_i)} \right) g(\theta^T x_i)(1 - g(\theta^T x_i)) \frac{\delta}{\delta_{\theta_j}} \theta^T x_i$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i (1 - g(\theta^T x_i)) - (1 - y_i) g(\theta^T x_i) \right) x_i^j$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i - g(\theta^T x_i) \right) x_i^j$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x_i) - y_i \right) x_i^j$$

θ 更新过程可以写成：

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

4. 邮件分类系统设计

我们的邮件分类系统大致步骤如下：

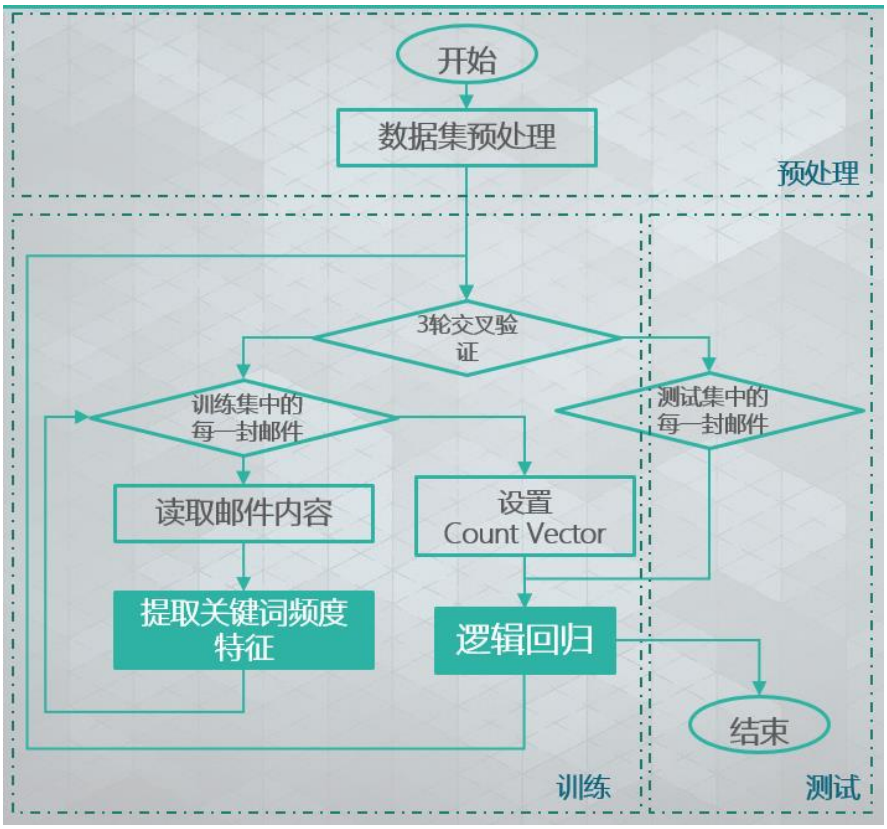
（1） 首先训练分类器，提取文本的关键词频度特征。

（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：



5. 实验环境

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

6. 实验数据集

本次实验的数据采用 Enron E-mail 数据集，是目前最丰富的从实际用户的邮件中获取的数据集。每篇邮件格式大概如下，第一行为标题，第二行开始为正文。

```
Subject: key dates and impact of upcoming sap implementation
over the next few weeks , project apollo and beyond will conduct its final sap
implementation □ ) this implementation will impact approximately 12 , 000 new
users plus all existing system users . sap brings a new dynamic to enron ,
enhancing the timely flow and sharing of specific project , human resources ,
procurement , and financial information across business units and across
continents .
this final implementation will retire multiple , disparate systems and replace
them with a common , integrated system encompassing many processes including
payroll , timekeeping , benefits , project management , and numerous financial
processes .
employees will be empowered to update and / or view their personal information
via the intranet - based ehronline - - a single front - end to sap ' s self service
functionality and enron ' s global information system ( gis ) . among other
things , individuals will be able to update personal information ( including
w - 4 , addresses and personal banking information ) , manage their individual
time using a new time entry tool , view their benefit elections , and view
their personal payroll information on - line .
all enron employees paid out of corporate payroll in houston , excluding
azurix employees
the financial communities of enron energy services , enron investment
partners , enron north america , enron renewable energy corporation , gas
pipeline group , global finance , global it , enron networks , and global
products .
```

7. 提取关键词频度特征

核心代码如下：

```
for 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。

8. 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')
```

9. 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。