基于文本挖掘的“智慧政务”

摘要

随着各种网络问政平台的增加、公开、开放，各种机构收集的非结构化数据不断增加，依靠人工进行留言划分和热点整理显然不够合理，工作量过大且效率过低。文本挖掘的价值定位和流行度也处于上升阶段，其有效地提高数学模型的效用，文本挖掘对于从文本资源库中提取知识具有极其重要的作用，能够加强政府和人民之间的有效沟通，更好的为人民服务。

对于三个不同的问题所要求的内容是一样的：本质就是进行自然语言处理和文本挖掘。其实三个问题就是一个处理“智慧政务”的完整流程，首先对群众的留言进行一个分类，其次我们可以看一下最近一段时间群众所集中关心的一些问题，最后就是根据政府部门对这些热点问题进行回答后，群众是否满意的一个评价，符合我们生活中的真实情况。

对于问题 1，实际上是一个文本分类（多分类）问题，一般常用的就是最经典的 word2vec 工具，需要将中文文本转换成数值形式，利用 KNN 来建立一级标签分类模型，并用 F-Score 对分类方法进行评价。

对于问题 2，就是如何从众多留言中识别出相似的留言，分词，统计词频，利用 KMeans 把特定地点或人群的数据聚类，即把相似的留言归为同一类问题，之后根据词频，建立热度评价指标，计算热度值，对指标进行排名。

对于问题 3，需要构建指标从相关性即相关部门答复意见的内容是否与问题相关、完整性即能否满足某种规范、可解释性答复意见中内容的相关解释，三方面描述量化，可用最便捷而清晰的流程图来描述。

关键词： 自然语言处理；文本挖掘；非结构数据量化；KNN；KMeans 聚类

一、 问题重述

近年来，随着微信、微博、市长信箱、阳光热线等网络问政平台逐步成为政府了解民意、汇聚民智、凝聚民气的重要渠道，各类社情民意相关的文本数据量不断攀升，给以往主要依靠人工来进行留言划分和热点整理的相关部门的工作带来了极大挑战。同时，随着大数据、云计算、人工智能等技术的发展，建立基于自然语言处理技术的智慧政务系统已经是社会治理创新发展的新趋势，对提升政府的管理水平和施政效率具有极大的推动作用。附件给出了收集自互联网公开来源的群众问政留言记录，及相关部门对部分群众留言的答复意见。请利用自然语言处理和文本挖掘的方法解决下面的问题。

1、群众留言分类

在处理网络问政平台的群众留言时，工作人员首先按照一定的划分体系(参考附件 1 提供的内容分类三级标签体系)对留言进行分类，以便后续将群众留言分派至相应的职能部门处理。目前，大部分电子政务系统还是依靠人工根据经验处理，存在工作量大、效率低，且差错率高等问题。请根据附件 2 给出的数据，建立关于留言内容的- -级标签分类模型。

通常使用 F-Score 对分类方法进行评价:

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2P_iR_i}{P_i + R_i}$$

其中 P_i 为第 i 类的查准率， R_i 为第 i 类的查全率。

2、热点问题挖掘

某一时段内群众集中反映的某一问题可称为热点问题，如“XXX 小区多位业主多次反映入夏以来小区楼下烧烤店深夜经营导致噪音和油烟扰民”。及时发现热点问题，有助于相关部门进行有针对性地处理，提升服务效率。请根据附件 3 将某一时段内反映特定地点或特定人群问题的留言进行归类，定义合理的热度评价指标，并给出评价结果，按表 1 的格式给出排名前 5 的热点问题，并保存为文件“热点问题表. xls”。按表 2 的格式给出相应热点问题，对应的留言信息，并保存为“热点问题留言明细表. xls”。

3、答复意见的评价

针对附件 4 相关部门对留言的答复意见，从答复的相关性、完整性、可解释

性等角度对答复意见的质量给出一套评价方案，并尝试实现。

二、 问题背景

智慧政务[1]的建设是实现电子政务升级发展的突破口，是政府从“管理型”走向“服务型、智慧型”的必然产物，也是引导智慧城市建设的主干线，现如今传统沟通方式与科技进步之间存在一种矛盾，“智慧政务”则是传统沟通方式迈向人工智能的一个典型例子。依靠深度学习来处理生活中问题已经成为主流，建立基于自然语言处理技术的智慧政务系统已经是社会治理创新发展的新趋势，对提升政府的管理水平和施政效率具有极大的推动作用。

社会公众在博客、论坛、社交网站和推特上用文字记录的办事经历，对政府政策、行政的看法和评价，公共服务诉求，投诉举报抱怨等是一种有非结构化的数据。如何把在散布在网络上的非结构化文本资源整合起来，并从中为政府挖掘有用的信息？

问题分析

在拥有完整三级标签体系基础上，将留言进行贴标签，看其归属于哪一类别的问题，其次存在众多留言的情况时，我们如何将其进行同类归类，比如说群众反映的是同一地区或同一件事，如何定义问题的热度，进行排序，如同我们常见的微博热搜话题，都会有一个热度的前后顺序，热度极高就被会贴上“爆”字醒目的标签，政府有关部门回复的质量也直接反应了该问题是否得到了满意的解决。

问题 1：利用 Python 的 jieba 进行留言分词，然后将附件二进行训练集和测试集的划分，用支持向量机对其进行分类，利用 F-Score 对分类方法进行评价。

问题 2：针对文本相似判定，利用 KMeans 方法进行聚类，需要将文字进行向量转换，统计词频分出热点话题。

问题 3:最简单的评价模型就是利用群众对政府相关部门回复的信息，我们进行后台的收集，对数据进行一定的分析和可视化，由于数据目前无法获取，无法进行建模过程，即用流程图来解答。

三、 模型的建立与求解

4.1 问题 1

4.1.1 选取变量并划分训练集和测试集

首先先进行了变量的选取，文档中的变量并不是所有都有用，第一问我们选择了留言主题和一级分类这两个变量，该操作有助于简化问题，减少不必要的因素产生误导。之后设置随机种子，将附件 2 中的数据进行 70%和 30%的训练集和测试集的划分，并分别存为 Data.train、Data.test 的 csv 文件和 txt 文档，见压缩包。

4.1.2 对测试集和训练集进行数据清洗

我们对数据进行查看是否存在缺失值，去掉缺失值，查看评论是否有重复，去掉重复的评论，为后面的工作做好准备。

4.1.3 对测试集和训练集进行分词

其中分词前，我们先将文档中存在的没用的停用词，比如说英文字母、数字、感叹词、关联词、符号、后缀变化等去掉，减少文本数据中出现的噪声，这里去掉的效果就和使用的停用词库有很大关系，我们这里选择使用搜狗的停用词表，尽量使词语不被拆分。这里只展示训练集和测试集的部分，如下所示：

```
[[1]]
[1] "市区"     "农村教师" "工资"     "福利待遇"

[[2]]
[1] "县到"   "路段"   "公交车" "节假日" "旅客"   "多时"   "出现"   "拒载"   "现象"

[[3]]
 [1] "区"     "捞刀河" "镇白霞" "村"     "梦"     "旅"     "噪音"   "排放"   "达标"   "严重"
[11] "扰民"

[[4]]
[1] "商品"   "混凝土" "质量"   "是否"   "可靠"

[[5]]
[1] "请"     "公平"   "对待"   "艾滋病" "患者"

[[6]]
[1] "工伤保险" "审批"     "程序"     "复杂"     "工作人员" "不在"     "岗位"
```

```
[[1]]
[1] "丁字街" "商户"   "乱"     "摆摊"

[[2]]
[1] "南门"   "街"     "干净"   "整洁"   "几天"   "老样子"

[[3]]
 [1] "县冷" "江东" "路"   "蓝波" "旺"   "酒店" "外墙" "装修" "无人" "施工"
```

4.1.4 构造矩阵，获取语料库

做分类时要先将文本转换为矩阵，用到 tm 软件包，先将训练集和预测集去除停词后的结果合并为 All，记住前 346（1:346）条数据是训练集，后 149（347:485）条是预测集。获取 jAll 的语料库，并且得到文档-词条矩阵，将其转换为普通矩阵。

4.1.5 利用 KNN 算法进行预测

所谓 K 近邻算法，即是给定一个训练数据集，对新的输入实例，在训练数据集中找到与该实例最邻近的 K 个实例（也就是上面所说的 K 个邻居）， 这 K 个实例的多数属于某个类，就把该输入实例分类到这个类中，即要根据训练集得到分类模型再为其做分类的预测。得到的预测结果如下：


数据编辑器

文件 编辑 帮助

	type	text
1	劳动和社会保障	"K8县丁字街的商户乱摆摊"
2	教育文体	"K8县南门街干净整洁了几天，又是老样子了"
3	劳动和社会保障	"K8县冷江东路蓝波旺酒店外墙装修无人施工"
4	教育文体	"K8县九亿广场的公厕要安装照明灯"
5	教育文体	"K4县石期市镇老农贸市场旁边的公厕（旱厕）里面脏、>
6	教育文体	"K市域轨道交通规划建议"
7	教育文体	"关于K市域轨道交通规划的建议"
8	教育文体	"请问A市乘坐地跌是否可以使用"爱心卡"？"
9	教育文体	"A市能不能像北方一样给居民小区统一建设供暖设备啊？
10	城乡建设	"A市可以实现集中供暖吗？"
11	教育文体	"反映K6县公交车监控的有关问题"
12	教育文体	"K4县农村信用合作联社208户合伙建房工程招投标有问？
13	城乡建设	"投诉A市盛世耀凯小区物业无故停水"
14	城乡建设	"楚江世纪城段沿江风光带园林区野草泛滥成灾"
15	教育文体	"A2区泰华一村小区第四届非法业委会涉嫌侵占小区业主>
16	教育文体	"L9县一幼建设教学楼不要与L9县十三五规划修缮熊公馆>
17	劳动和社会保障	"关于对M7市首府印象增加容积率和提高建筑密度的质疑"
18	城乡建设	"建议对东六线海德公元小区单层露台的违章建筑进行最>
19	城乡建设	"关于加快C4市白田特色小城镇建设的建议"

4.1.6 对 KNN 分类结果进行评价

为了展示多类别设置中非得分分类器的性能指标，将一级指标按照 1-7 的数字进行标签，将预测的结果也按照同样的数字进行标签，利用 F1 值进行评价模型拟合的效果。效果如下：

```
[1] "-------------f1--------------------"
        1         2         3         4         5         6         7
0.9295775 0.7777778 0.5217391 0.8235294 0.9375000 0.7619048 0.8484848
[1] "--------------mean(f1)------------------"
[1] 0.8000733
>
```

即利用 KNN 算法得出模型的准确率约为 80%。准确率不算太高，在分类效果不理想的情况下，改进分类效果需要丰富训练集，让训练集特征尽量明显，这个在实际问题是一个很繁琐却不能敷衍的过程。

4.2 问题 2

4.2.1 识别相似留言

首先将留言主题的文本提取出来，对文本进行预处理操作：去除缺失值、去掉重复的留言内容、对留言文本进行分词等。

接着利用 R 语言的 tm 包进行文本挖掘，tm 包是 R 语言中为文本挖掘提供综合性处理的 package，在 tm 包中主要的管理文件的结构被称为语料库（Corpus），代表了一系列的文档集合。语料库是一个概要性的概念，在这里分为动态语料库（Volatile Corpus，作为 R 对象保存在内存中）和静态语料库（Permanent Corpus，R 外部保存）。

然后利用 DocumentTermMatrix 将处理后的语料库进行断字处理，生成词频权重矩阵，为下一步的归类做好数据准备。

4.2.2 留言归类

本文对留言主题的归类问题，采用 Kmeans 聚类方法。聚类算法是指将一堆没有标签的数据自动划分成几类的方法，属于无监督学习方法。Kmeans 聚类方法在数据挖掘和分析中运用较为广泛，本文选取适当的 k=5，将数据分成 5 类，分类结果详见 csv 文档（Data_kmeansRes.csv），并分类研究不同聚类下的数据。

Kmeans 聚类数学原理：如果用数据表达式表示，假设簇划分为$(C_1,C_2,...C_k)$，则我们的目标是最小化平方误差 E ：

$$E=\sum_{i=1}^{k}\sum_{x\in C_i}\|x-\mu_i\|_2^2$$

其中μ_i是簇C_i的均值向量，有时也称为质心，表达式为：

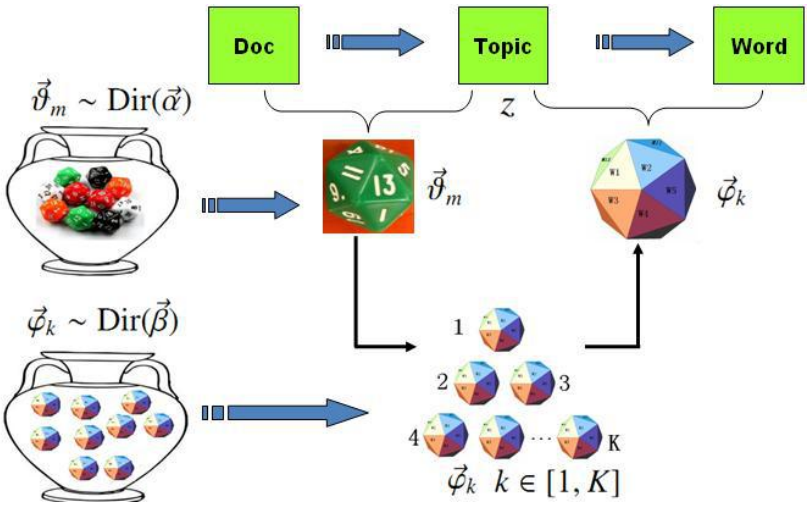
$$\mu_i = \frac{1}{|C_i|}\sum_{x \in C_i} x$$

如果我们想直接求上式的最小值并不容易，这是一个 NP 难的问题,因此只能采用启发式的迭代方法。

4.2.3 热度评价

热度评价指标根据主题模型分析来定义，主题模型就是对文字中隐含主题的一种建模方法。用数学的语言来描述的话，主题就是词汇表上词语的条件概率分布，与主题关系越密切的词语，它的条件概率越大，反之则越小。主题模型训练推理的方法主要有两种，一个是 pLSA（Probabilistic Latent Semantic Analysis），另一个是 LDA（Latent Dirichlet Allocation）。pLSA 主要使用的是 EM（期望最大化）算法；LDA 采用的是 Gibbs sampling 方法。

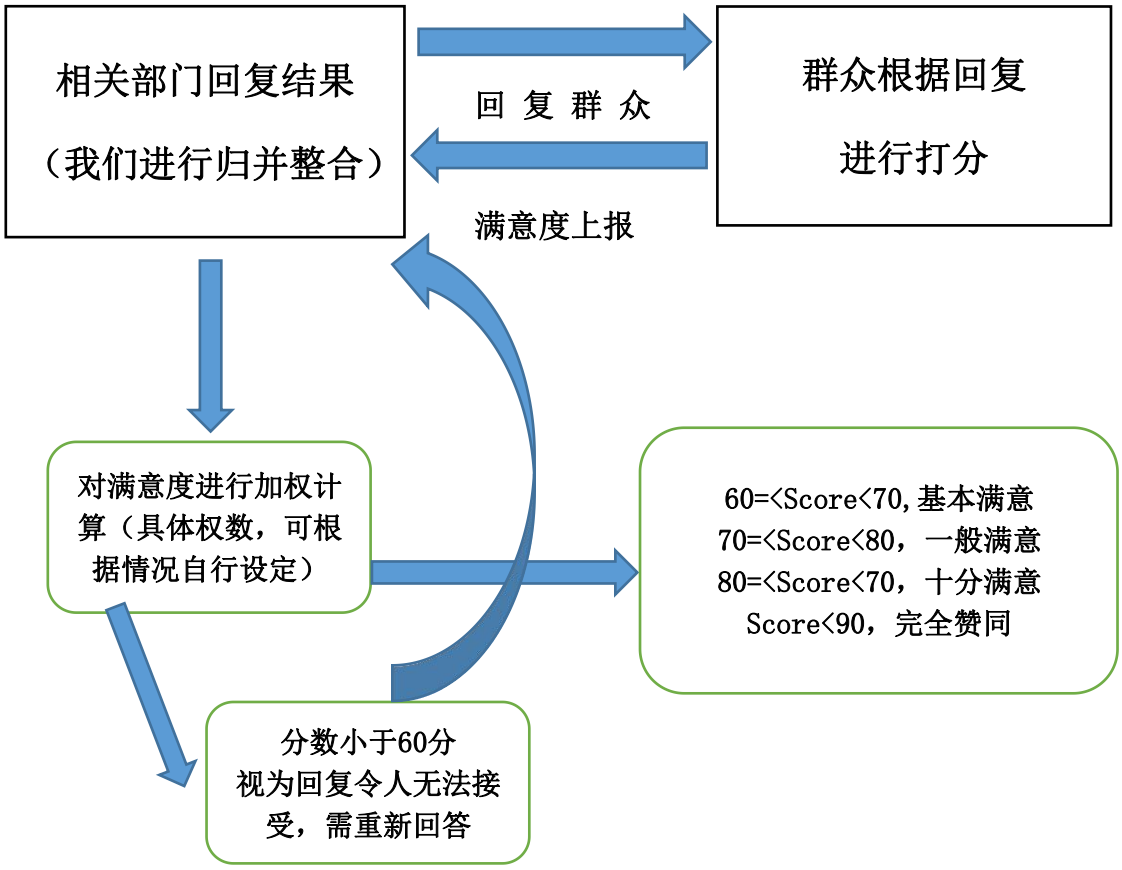
本文采用 LDA 主题模型来对热度指标进行计算和评价,LDA 模型简要图如下：



4.2 问题 3

我们先对相同热门问题的回复进行汇总，之后将其回复进行归并，因为回复的字数和措辞其实在一定程度上并不能代表着什么，我们将其回复的内容进行一个相同覆盖，然后将其公布出来，再由群众对其部门的回复进行打分，比如说现在微博上比较流行的投票，投票的选项可以从答复的相关性、完整性、可解释性等角度对答复意见的质量进行评价，我们利用后台的数据，对票数进行处理。比

如说，我们对选项进行加权计算分数，相关性、完整性、可解释性三部分各占⅓的比例，之后算出来的加权分数，如果超过及格线分数（比如说 60），就可以评定该回复基本上满意，超过 70 分、超过 80 分、超过 90 分分别对应满意、十分满意、完全赞同等评价类词语。如下流程图所示：



四、 模型的优缺点分析

优点：选用的模型都是比较简单的模型，理论也比较容易理解

缺点：对于问题 1：word2vec 的安装本身就是个很麻烦的问题，更不要说使用了，最后建立模型的 F-Score 较低，数据也出现不平衡的问题，且长文本的无意义表达过多，虽然进行了一些处理，但是还是比较乱。对于问题 2，相似度计算量过大，准确率较低。对于问题 3，因为只是理论上的建模，描述性量化是交给了人民群众完成。

五、 参考文献

[1] 智能政务，http://city.pingan.com/solution/governmentAffairs.html
[2] Hellooooooworld.文本分类的Python实现-基于支持向量机分类,https://www.jianshu.com/p/f2575e88cf6f,2017.05.10
[3] 刘勇.文本挖掘之相似度判定，https://www.cnblogs.com/lyssym/p/4880896.html，2015.10.15

六、 附录

代码：

```
#-----------------------------------C题-----------------#
##---------------------第一问-------------------------#
读取数据
Data <- read.csv('F:/泰迪杯/第八届泰迪杯/附件2.csv', header = TRUE, encoding = 'utf-8')
head(Data)
dim(Data)

#查看是否存在缺失值
sum(is.na(Data))
Data<-na.omit(Data) #去除缺失值
sum(is.na(Data))

#将没用的变量删去
x1<-Data$留言主题
head(x1)
x2<-Data$一级分类
Data<-data.frame(x1,x2)
head(Data)
dim(Data)

#划分训练集和测试集
set.seed(1)
train.rows<-sample(rownames(Data), dim(Data)[1]*0.7)
Data.train<-Data[train.rows, ] #70%的训练集
valid.rows <- setdiff(rownames(Data), train.rows)#30%的测试集
Data.test<-Data[valid.rows, ]
write.csv(Data.train, "F:/泰迪杯/第八届泰迪杯/Data.train.csv", row.names = FALSE)
write.csv(Data.test, "F:/泰迪杯/第八届泰迪杯/Data.test.csv", row.names = FALSE)
write.table(Data.test$x1, "F:/泰迪杯/第八届泰迪杯/Data_test.txt", row.names = FALSE)
```

```
Data.train<-read.csv("F:/泰迪杯/第八届泰迪杯
/Data.train.csv",header=T,stringsAsFactors=F)
length(Data.train)

table(Data.train$x2)

Data_test<-readLines("F:/泰迪杯/第八届泰迪杯/Data_test.txt",encoding
= "GBK")
length(Data_test)
class(Data_test)
head(Data_test)

#数据清洗——评论去重
Data.train<- unique(Data.train)
length(Data.train)

#训练集分词
Data.train1<-gsub("[a-zA-Z]", " ", Data.train$x1)
head(Data.train1)
Data.train2<-gsub("[0-9]", " ", Data.train1)
head(Data.train2)

library(Rwordseg)
library(jiebaRD)
library(Rcpp)
library(jiebaR)
library(tm)

library(pryr)
otype(wk)
class(wk)
show_dictpath()
dir(show_dictpath())
wk = worker(user='user.utf8',bylines = T)

Data.train3<-segment(Data.train2,wk)
head(Data.train3)

#测试集分词
Data_test1<-gsub("[a-zA-Z]", " ", Data_test)
head(Data_test1)
Data_test2<-gsub("[0-9]", " ",Data_test1)
head(Data_test2)
```

```
library(Rwordseg)
Data_test3<-segment(Data_test2,wk)
head(Data_test3)


#去掉训练集停用词
stopwords<- unlist (readLines("F:/数据挖掘/数据挖掘案例三/ch15上机实验数据代码/data/stoplist.txt", encoding = "UTF-8"))
removeStopWords = function(x, words) {
  ret <- character(0)
  index <- 1
  it_max <- length(x)
  while (index <= it_max) {
    if (length(words[words == x[index]]) <1) ret <- c(ret, x[index])
    index <- index + 1
  }
  ret
}
train.words1<-lapply(Data.train3,removeStopWords,stopwords)
head(train.words1)
length(train.words1)

#去掉测试集停用词
stopwords<- unlist (readLines("F:/数据挖掘/数据挖掘案例三/ch15上机实验数据代码/data/stoplist.txt", encoding = "UTF-8"))
removeStopWords = function(x, words) {
  ret <- character(0)
  index <- 1
  it_max <- length(x)
  while (index <= it_max) {
    if (length(words[words == x[index]]) <1) ret <- c(ret, x[index])
    index <- index + 1
  }
  ret
}
test.words1<-lapply(Data_test3,removeStopWords,stopwords)
head(test.words1)
length(test.words1)


#得到矩阵
All<- character(0)
All[1:346] <-train.words1
```

```
All[347:495] <-test.words1
length(All)

#获取语料库
library(tm)
corpusAll<-Corpus(VectorSource(All))
All.dtm <-DocumentTermMatrix(corpusAll,control=list(wordLengths =
c(2,Inf)))

dtmAll_matrix <-as.matrix(All.dtm)

#分类
rownames(dtmAll_matrix)[1:346] <-Data.train$x2
rownames(dtmAll_matrix)[367:495]<- c("")
train <- dtmAll_matrix[1:346,]
predict <-dtmAll_matrix[347:495,]
trainClass <-as.factor(rownames(train))
library(class)
knnClassify <-knn(train,predict,trainClass)
length(knnClassify)

knnClassify[1:10]

table(knnClassify)

knnResult <-list(type=knnClassify,text=Data_test)
knnResult <-as.data.frame(knnResult)
fix(knnResult)

#F1值对结果进行评价
#pre：预测的分类结果
#y：真实的分类结果
f1_fun = function(pre,y){
  class = sort(unique(y))
  tp=NA
  fp=NA
  fn=NA
  for(i in 1:length(class)){
    tp[i] = sum(pre==class[i] & y==class[i])
    fp[i] = sum(pre==class[i] & y!=class[i])
    fn[i] = sum(pre!=class[i] & y==class[i])
  }
  f1 = 2*tp/(2*tp+fp+fn)
  names(f1) = class
```

```
  print(table(pre,y))
  print('-------------f1--------------------')
  print(f1)
  print('--------------mean(f1)-------------------')
  print(mean(f1))
}
#应用
Data.test<-read.csv("F:/泰迪杯/第八届泰迪杯
/Data.test.csv",header=T,stringsAsFactors=F)
pre_result =knnResult$type
y_true =Data.test$x2
f1_fun(pre_result,y_true)


ref.labels=Data.test$x2
predictions=knnResult$type
df <- data.frame("Prediction" = predictions, "Reference" = ref.labels)

calculate.accuracy <- function(predictions, ref.labels) {
  return(length(which(predictions == ref.labels)) / length(ref.labels))
}
calculate.w.accuracy <- function(predictions, ref.labels, weights) {
  lvls <- levels(ref.labels)
  if (length(weights) != length(lvls)) {
    stop("Number of weights should agree with the number of classes.")
  }
  if (sum(weights) != 1) {
    stop("Weights do not sum to 1")
  }
  accs <- lapply(lvls, function(x) {
    idx <- which(ref.labels == x)
    return(calculate.accuracy(predictions[idx], ref.labels[idx]))
  })
  acc <- mean(unlist(accs))
  return(acc)
}
acc <- calculate.accuracy(df$prediction, df$Reference)
print(paste0("Accuracy is: ", round(acc, 2)))

#加上因子水准
for(i in 1:149)
 {if(knnResult$type[i]=="城乡建设")
 {knnResult$type[i]=1}
 else if(knnResult$type[i]=="环境保护")
```

```
  {knnResult$type[i]=2}
 else if(knnResult$type[i]=="交通运输")
  {knnResult$type[i]=3}
 else if(knnResult$type[i]=="教育文本")
  {knnResult$type[i]=4}
 else if(knnResult$type[i]=="劳动和社会保障")
  {knnResult$type[i]=5}
 else if(knnResult$type[i]=="商贸旅游")
  {knnResult$type[i]=6}
 else (knnResult$type[i]=="卫生计生")
  {knnResult$type[i]=7}}



##-----------------------第二问-----------------------##
#用setwd设置工作空间
setwd('C:/Users/Lenovo/Desktop/C题示例数据/示例数据')
Data <- read.csv('./附件2.csv', header = TRUE, encoding = 'utf-8')

###1.数据预处理
#查看前几行数据、数据结构、数据类型、行名
head(Data)
dim(Data)
class(Data)
names(Data)

#查看是否存在缺失值
sum(is.na(Data))
Data<-na.omit(Data)  #去除缺失值
sum(is.na(Data))  #再次查看

#选取留言主题的内容，并保存为txt文档
theme_jd <- Data$留言主题
write.table(theme_jd, "./theme_jd.txt", row.names = FALSE)
#查看前几行数据及数据长度
head(theme_jd)
length(theme_jd)
tneme_jd <- unique(theme_jd)  #去重
length(theme_jd)  #再次查看长度

#查看前几行原始数据和留言主题数据
head(Data)
head(theme_jd)
```

```
###2.对留言主题进行分词
#载入所需用的包
library(jiebaRD)
library(Rcpp)
library(jiebaR)
library(tm)
wk <- worker()
wk
library(pryr)
otype(wk)
class(wk)
show_dictpath() #查看默认的词库位置
dir(show_dictpath())  # 查看所在目录

#载入系统词典文件jieba.dict.utf8，并打印前50行
scan(file="D:/R/R-3.6.3/library/jiebaRD/dict/jieba.dict.utf8",what=ch
aracter(),nlines=50,sep='\n',encoding='utf-8',fileEncoding='utf-8')
#载入用户词典文件user.dict.utf8，并打印前50行
scan(file="D:/R/R-3.6.3/library/jiebaRD/dict/user.dict.utf8",what=cha
racter(),nlines=50,sep='\n',encoding='utf-8',fileEncoding='utf-8')
wk = worker(user='user.utf8',bylines = T)
wk['./theme_jd.txt']

#导入留言主题分词后数据
Data01<-readLines('./theme_jd.segment.2020-05-08_13_35_43.txt',encodi
ng = 'UTF-8')
head(Data01)

#用segment()函数把每个词分隔开后用worker分词器进行分词
library(Rwordseg)
Data02<-segment(Data01,wk)
head(Data02)

#去掉停用词
stopwords<- unlist (readLines("./stoplist.txt", encoding = "UTF-8"))
removeStopWords = function(x, words) {
  ret <- character(0)
  index <- 1
  it_max <- length(x)
  while (index <= it_max) {
    if (length(words[words == x[index]]) <1) ret <- c(ret, x[index])
    index <- index + 1
  }
  ret
```

```
}
sample.words1<- lapply(Data02,removeStopWords,stopwords)
head(sample.words1)

###3.用tm做文本挖掘处理
library(NLP)
library(tm)
#读取分词后的文件
mydoc<-sample.words1
edit(mydoc) #查看文件，看看是否有乱码的现象

#生成语料库
mydoc.vec<-VectorSource(mydoc)
mydoc.corpus<-Corpus(mydoc.vec)

inspect(mydoc.corpus) #查看语料库中的文档

#导出语料库，方法：writeCorpus(x, path = ".", filenames = NULL)
writeCorpus(mydoc.corpus, path = "C:/Users/Lenovo/Desktop/C题示例数据/
示例数据/语料库/",filenames = paste(seq_along(mydoc.corpus), ".txt",
sep = ""))

#语料库转换为文档-词条矩阵
mydoc.dtm<-
DocumentTermMatrix(mydoc.corpus,control=list(wordLengths=c(2,Inf)))
#再将文档-词条矩阵转换为普通矩阵
mydoc.matrix <- as.matrix(mydoc.dtm)

k <- 5
kmeansRes <- kmeans(mydoc.matrix,k) #k是聚类数
mode(kmeansRes)#kmeansRes的内容

names(kmeansRes)
head(kmeansRes$cluster,10)

mydoc.kmeansRes <- list(content=mydoc,type=kmeansRes$cluster)
write.csv(mydoc.kmeansRes,"mydoc_kmeansRes.csv")
fix(mydoc.kmeansRes)

# 主题模型分析
library(topicmodels)
Gibbs = LDA(mydoc.dtm, k = 5, method = "Gibbs",control = list(seed = 2015,
burnin = 1000, thin = 100, iter = 1000))
# 最可能的主题文档
```

```
Topic1 <- topics(Gibbs, 1)
table(Topic1)
# 每个Topic前10个Term
Terms1 <- terms(Gibbs, 10)
Terms1
```