C 题：“智慧政务”中的文本挖掘应用

一、前言

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

本文针对提供的数据，建立相关智慧政务系统模型，以解决群众留言分类、热点问题挖掘、答复意见的评价等问题。首先对所给数据进行预处理，建立了相关的统计模型，并运用 anaconda、Excel 等软件进行统计分析，最后对系统进行评价，并找出最优解。

二、问题描述

2.1. 问题背景

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。

2.2. 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一级标签分类模型。
通常使用 F-Score 对分类方法进行评价：

2.3. 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”。

2.4. 答复意见的评价

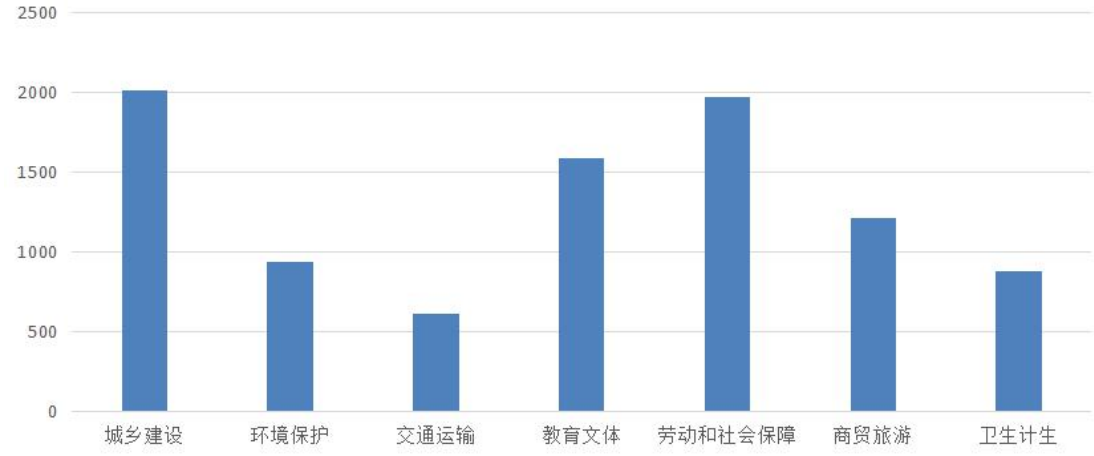
针对附件 4 相关部门对留言的答复意见，从答复的相关性、完整性、可解释性等角度对答复意见的质量给出一套评价方案，并尝试实现。

三、问题一

3.1 计算所用理论与方法

在处理网络问政平台的群众留言时，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。根据附件 2 给出的数据，建立关于留言内容的一级标签分类模型。

测试集[1]对于训练完成的神经网络,测试集用于客观的评价神经网络的性能，是机器学习学科中，学习样本三部分之一，测试集用来检验最终选择最优的模型的性能如何，指的是用于训练的样本集合,主要用来训练神经网络中的参数，是机器学习学科中，学习样本三部分之一，训练集用于建立模型。通常会取 80%作为训练集，20%作为测试集。验证集用来确定网络结构或者控制模型复杂程度的参数，而测试集则检验最终选择最优的模型的性能如何。在比赛数据中，各类一级标签数量，对应的柱状图如图一所示。



图一：各类一级标签数量柱状图

在比赛数据中，各类一级标签的具体数量以及根据程序运算得到从中选取的各类标签数量所占的比例如表一所示：

表一:各类一级标签具体数量及对应标签选取所占比例

一级标签	数量	所占比例
城乡建设	2009	21.8%
环境保护	938	10.2%
交通运输	613	6.7%
教育文体	1589	17.3%
劳动和社会保障	1969	21.4%
商贸旅游	1215	13.2%
卫生计生	877	9.5%

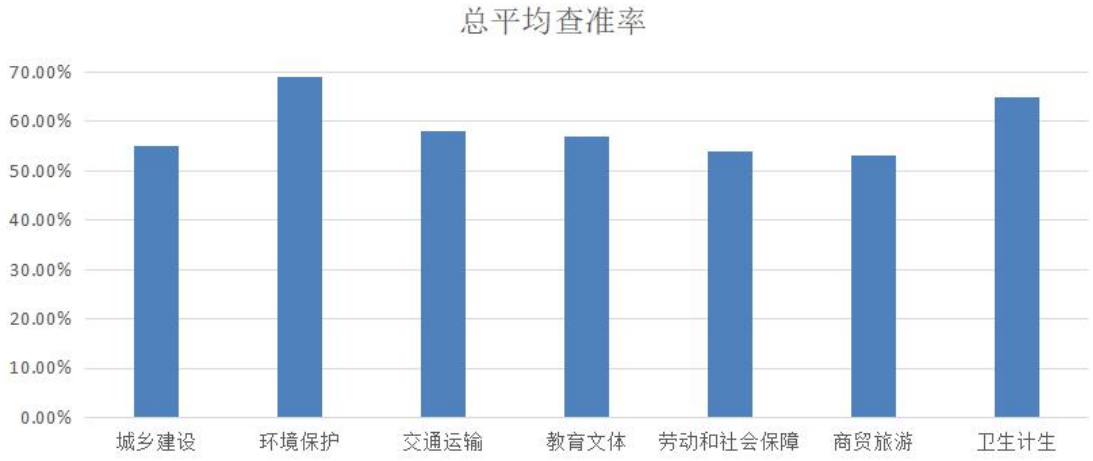
查准率（精度）[2]是衡量某一检索系统的信号噪声比的一种指标，即检出的相关文献量与检出的文献总量的百分比。普遍表示为：查准率=（检索出的相关

信息量/检索出的信息总量）x100%。使用专指性较强的检索语言(如上位类、上位主题词)能提高查准率，但查全率下降。

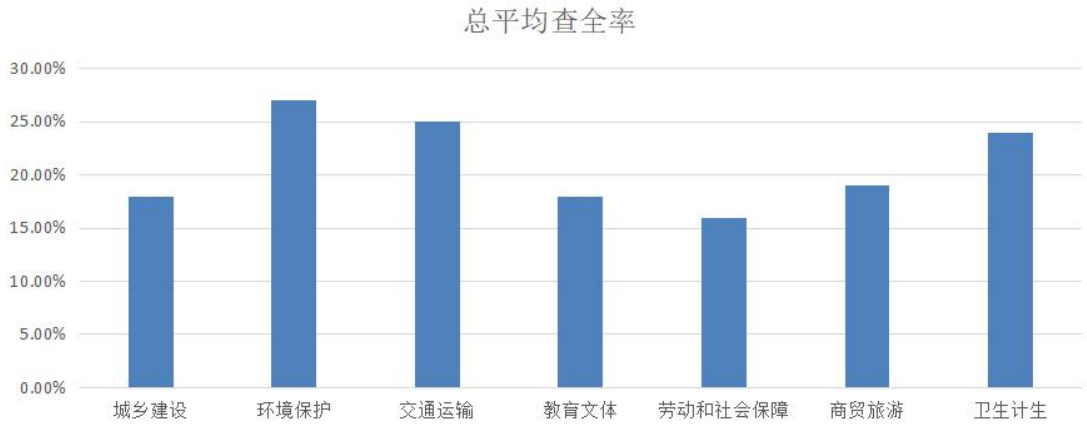
查全率（召回率），是衡量某一检索系统从文献集合中检出相关文献成功度的一项指标，即检出的相关文献量与检索系统中相关文献总量的百分比。普遍表示为：查全率=（检索出的相关信息量/系统中的相关信息总量）x100%。使用泛指性较强的检索语言(如上位类、上位主题词)能提高查全率，但查准率下降。

查全率[3]与查准率是信息检索领域内的概念，二者是反映检索效果的重要指标。根据查准率和查全率可绘制系统的 PR 曲线，可根据曲线判断系统的优劣。

画出各类一级标签的查准率、查全率柱状图，如图二、图三所示。

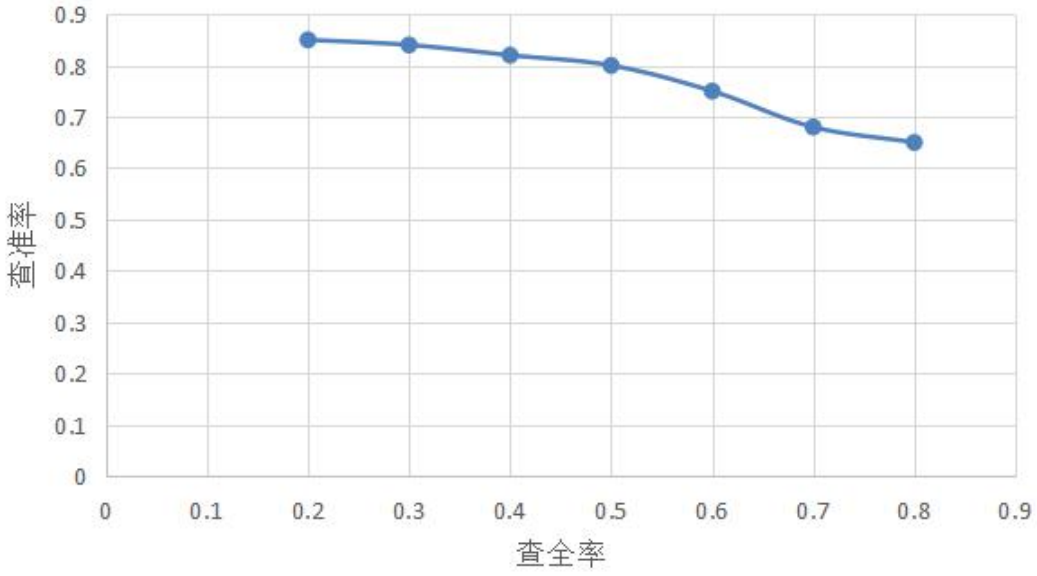


图二：各类一级标签总平均查准率



图二：**各类一级标签总平均查全率**

列出查全率—查准率折线示意图，如图三所示。



图三：查全率—查准率折线示意图

从查全率—查准率折线图可知，总体曲线符合大致规律，趋势为随着查全率的增加，查准率下降。

3.2 anaconda 程序

所用程序如下：

```
#生成 data
import jieba
import pandas as pd
import time
begin_time=time.time()
#附件 2.xlsx 为要处理的文件
#“一级分类”是人工识别结果
#注意这个“一级分类”中应包含各类尽量不要太少
data=pd.read_excel('附件 2.xlsx')
#修改列名
#data.columns=['fileId','result','text']
#每行用函数处理，进行分词
data['留言详情']=data['留言详情'].apply(jieba.cut).apply(list) #耗时
#读入停用词库，事先分析 text，将没有正确分词的词加入词库 stoplist.txt
stop_word=pd.read_table('stoplist.txt',sep='aaaa',encoding='utf-8')
stop_word=stop_word.iloc[:,0].tolist()
#停用词过滤，找出有价值词
data['留言详情']=data['留言详情'].apply(lambda x:[i for i in x if i not in stop_word])
#耗时
#  把列表转成字符串
data['留言详情'] = data['留言详情'].apply(lambda x:' '.join(x))
#  划分训练集测试集
from sklearn.model_selection import train_test_split
train_data,test_data,train_label,test_label = train_test_split(
```

```
data['留言详情'],data['一级分类'],test_size=0.2)
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
train_cv = cv.fit_transform(train_data)
train_cv = train_cv.toarray() #MemoryError
# 贝叶斯算法
from sklearn.naive_bayes import MultinomialNB
model_nb = MultinomialNB()
model_nb.fit(train_cv,train_label)
# 训练集预测值
pre = model_nb.predict(train_cv) # 对训练样本预测
# 正确率
sum(pre == train_label) / len(train_label)
#测试集预测值
cv1 = CountVectorizer(vocabulary=cv.vocabulary_) # 使用训练集的词汇表
test_cv = cv1.fit_transform(test_data)
test_cv = test_cv.toarray() #MemoryError
test_pre = model_nb.predict(test_cv) # 用训练好的模型预测测试集数据
# 测试集正确率
sum(test_pre == test_label) / len(test_label)
from sklearn.metrics import f1_score
print('F1_score(macro):',f1_score(test_label, test_pre, average='macro'))    #
print('F1_score(micro):',f1_score(test_label, test_pre, average='micro'))    #
print('F1_score(weighted):',f1_score(test_label, test_pre, average='weighted'))
end_time=time.time()
print('耗时:',end_time-begin_time)
#保存模型
from sklearn.externals import joblib
# 保存成 sklearn 自带的文件格式
joblib.dump(model_nb , 'model_nb.pkl')
joblib.dump(cv1 , 'cv1.pkl')
```

其中，得到相关度数据如下

F1_score(macro): 0.890143873641

F1_score(micro): 0.947070707071

F1_score(weighted): 0.887707949777

3.3 对 F-Score 方法进行分析

F-Score[4]（非模型评价打分，区别与 F1_score ）是一种衡量特征在两类之间分辨能力的方法，通过此方法可以实现最有效的特征选择。最初是由台湾国立大学的 Yi-Wei Chen 提出的（参考《Combining SVMs with Various Feature Selection Strategies》），公式如下：

$$F(i)=\frac{(\overline{x}_i^{(+)}-\overline{x}_i)^2+(\overline{x}_i^{(-)}-\overline{x}_i)^2}{\frac{1}{n_+-1}\sum_{k=1}^{n_+}(x^{(+)}_{k,i}-\overline{x}_i^{(+)})^2+\frac{1}{n_--1}\sum_{k=1}^{n_-}(x^{(-)}_{k,i}-\overline{x}_i^{(-)})^2}$$

其中 i 代表第 i 个特征，即每一个特征都会有一个 F-score。x 拔是所有该特征值的平均数，而（+），（-）则分别代表所有阳性样本和阴性样本的特征值（的平均数）。代表 k 是对于具体第 i 个特征的每个实例，分母的两个 sigma 可以理解为阳性样本与阴性样本的特征值的方差。F-score 越大说明该特征的辨别能力越强。谢娟英等人还改进了 F-score，将它推广至多分类的情况（查看文献），公式如下：

$$F(i)=\frac{\sum_{j=1}^{l}(\overline{x_i}^{(j)}-\overline{x_i})^2}{\sum_{j=1}^{l}\frac{1}{n_j-1}\sum_{k=1}^{nj}(x^{(j)}{}_{k,i}-\overline{x_i}^{(j)})^2}$$

不难发现，原版的 F-score 是改进版的一个特例，即 l=2, 分别代表正类与负类（我们通常用 0 和 1 表示）

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 Pi 为第 i 类的查准率，Ri 为第 i 类的查全率。

四、问题二的计算情况：

4.1 问题二的计算方法和理论

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”。

其中附表一为

表 1-热点问题表

热度排名	问题 ID	热度指数	时间范围	地点/人群	问题描述
1	1	…	2019/08/18 至 2019/09/04	A 市 A5 区魅力之城小区	小区临街餐饮店油烟噪音扰民
2	2	…	2017/06/08 至 2019/11/22	A 市经济学院学生	学校强制学生去定点企业实习
…	…	…	…	…	…

附表二为：

表 2-热点问题留言明细表

问题ID	留言编号	留言用户	留言主题	留言时间	留言详情	点赞数	反对数
1	360104	A012417	A 市魅力之城商铺无排烟管道，小区内到处油烟味	2019/08/18 14:44:00	A 市魅力之城小区自打交房入住后，底层商铺无排烟管道，经营餐馆导致大量油烟排入小区内，每天到凌晨还在营业……	0	0
1	360105	A120356	A5 区魅力之城小区一楼被搞成商业门面，噪音扰民严重	2019/08/26 08:33:03	我们是魅力之城小区居民，小区朝北大门两侧的楼栋下面一楼，本来应是架空层，现搞成商业门面，噪声严重扰民，有很大的油烟味往楼上窜，没办法居住……	1	0
1	360106	A235367	A 市魅力之城小区底层商铺营业到凌晨，各种噪音好痛苦	2019/08/26 01:50:38	2019 年 5 月起，小区楼下商铺越发嚣张，不仅营业到凌晨不休息，各种烧烤、喝酒的噪音严重影响了小区居民休息……	0	0
…	…	…	…	…	…	…	…
1	360109	A008025 2	魅力之城小区底层门店深夜经营，各种噪音扰民	2019/09/04 21:00:18	您好：我是魅力之城小区的业主，小区临街的一楼是商铺，尤其是餐馆夜宵摊等，每到凌晨都还在营业，每到晚上睡觉耳边都充斥着吆喝……	0	0
2	360110	A110021	A 市经济学院寒假过年期间组织学生去工厂工作	2019/11/22 14:42:14	西地省 A 市经济学院寒假过年期间组织学生去工厂工作，过年本该是家人团聚的时光，很多家长一年回来一次，也就过年和自己孩子见一次面，可是这样搞……	0	0
2	360111	A120445 5	A 市经济学院组织学生外出打工合理吗？	2019/11/5 10:31:38	学校组织我们学生在外边打工，在东莞做流水线工作，还要倒白夜班。本来都在学校好好上课，十月底突然说组织到外省打工……	1	0
…	…	…	…	…	…	…	…

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，其中热点问题表，热点问题留言明细表如下所示。

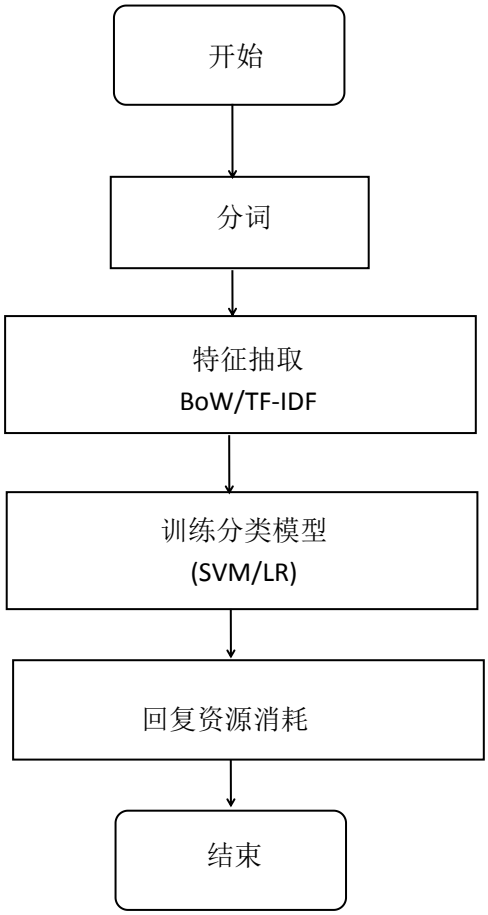
表二：热点问题表

热度排名	问题ID	热度指数	时间范围	地点/人群	问题描述
1	1	2344	2019/2/23至2019/3/01	A市市民	车贷案
2	2	2099	2019/1/23至2019/8/19	A市汇金路小区	小区物业纠纷
3	3	1778	2019/4/11至2019/5/29	A市学生	配套入学问题
4	4	732	2019/8/01至2019/9/05	A市高铁市民	高铁选址问题
5	5	91	2019/2/25至2019/11/07	A市市民	房子质量问题

表三：热点问题留言明细表

问题ID	留言编号	留言用户	留言主题	留言时间	留言详情	点赞数	反对数
1	220711	A00031682	注A市A4区	/2/21 18:4	消息总是失	0	821
1	217032	A00056543	贷特大集资	9/2/25 9:5	东、苏纳弟	0	790
1	194343	A00010616	贷案警官应	9/3/1 22:1	并没有跟进	0	733
2	208636	A00077171	矿万境K9县	/8/19 11:3	人，请问有	0	2097
2	208069	A00094436	开发商与施	9/5/5 13:5	量是否能符	0	2
3	220716	A00089049	路之间路段	/1/23 10:3	天烂泥巴到	0	0
3	223297	A00087522	毛湾配套入	/4/11 21:0	入配套入学，	5	1762
3	224997	A000111541	1区嘉苑的	/5/29 15:3	满足《A市城	0	11
4	263672	A00041448	赣高铁最近	9/9/5 13:0	我如下问题	0	669
4	203187	A00024716	市高铁站选址	9/8/1 13:4	东进的步伐	53	10
5	281898	A00096623	多栋房子现	/2/25 15:1	处理，陷入	5	55
5	283631	A00042509	噪音污染严	/11/7 10:1	窗透气，可	2	29

将模型的输出结果[5]作为任务完成度的评价指标，同时统计对话消耗的轮数作为对话资源消耗 N，将这二者经过判断得出用户满意度这一指标，若任务未完成，则认为用户满意度最差（即为 0），若任务完成但对话资源损耗超过阈值（实验中设定 N=20）则认为用户满意度为中等（即为 1），若任务完成且对话资源消耗在阈值之内，则认为该任务型对话系统又好又快的完成了用户提出的任务请求，即认为用户满意度最高（即为 2）相关流程图如图四[11]所示。



图四：评分流程图

该图表示了对任务型对话系统回复质量评价的流程，按照这个流程及上述的指标确定，最后得到的是对数据集中测试集内每一个回复完成度，回复资源消耗，和解决方案合理性的均值，这三个返回的指标就可以当作回复系统的最终评价结果。除测试集外，实验中还设定了一个测试文档，在测试文档中放入一个未在测试集和训练集中的对话 session，将这个文档输入评价系统，最终返回针对这个文档的评价结果。

精确率[5]（Accuracy，A）的计算方法。对于任务完成的回复来说，评价指标结果相同，都可以反应模型正确分类的能力，具体表示为被正确分类的样本数与样本总数的比值。

$$A = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

准确率（Precision，P）的计算分为两部分，对回复较好的样本来说（将任务完成的对话段默认为正类 positive），用于表示被分类模型判定为完成的样本中真正完成的比例。对回复不合理的信息来说（未完成对话段默认为负类 negative），用于表示被分类模型判定为未完成的样本中真正的未完成的比例。

$$P_R = \frac{TP}{TP + FP}$$

$$P_T = \frac{TN}{TN + FN}$$

召回率[6]（Recall，R）的计算也有两部分，对未完成的信息来说，表示被合理回复分类的完成的对话段样本数与数据集中真正完成的样本总数的比值，对回复未解决问题的来说，反应被正确分类的未完成样本数占数据集中真正未完成的样本总数的比例，公式如下。

$$R_R = \frac{TP}{TP + FN}$$

$$R_T = \frac{TN}{TN + FP}$$

F1 值的计算方法也就是计算了准确率与召回率的调和平均值，公式如下。

$$\frac{2}{F_1} = \frac{1}{P} + \frac{1}{R}$$

4.2 问题二所用程序

问题二留言相似度及评价所用程序如下：

```
#附件 3 留言详情相似度计算

#https://blog.csdn.net/mouday/article/details/87921873

#使用 gensim 对中文文本进行相似度计算

import logging

import pandas as pd

import jieba

from gensim import corpora, models, similarities



'''

path='D:/aaazyw/泰迪/2020-8/C 题全部数据/'
```

```
file=path+'chineseStopWords.txt'
#gb18030,utf-8,ISO-8859-1,gb2312
with open(file, 'r', encoding='utf-8') as f:
    try:
        text = f.read()
    except:
        with open(file, 'r', encoding='gb2312') as f:
            text = f.read()

stopwords1=text.split('\n')
'''


logging.basicConfig(level=logging.DEBUG)
jieba.setLogLevel(logging.INFO)


class DocumentSimilar(object):
    def __init__(self, documents):
        self.documents = documents
        self.dictionary = None
        self.tfidf = None
        self.similar_matrix = None
        self.calculate_similar_matrix()
```

```
@staticmethod
def split_word(document):
    """
    分词，去除停用词
    """
    path = 'D:/aaazyw/泰迪/2020-8/C 题全部数据/'
    file = path + 'chineseStopWords.txt'
    # gb18030,utf-8,ISO-8859-1,gb2312
    with open(file, 'r', encoding='utf-8') as f:
        try:
            text = f.read()
        except:
            with open(file, 'r', encoding='gb2312') as f:
                text = f.read()

    stopwords1 = text.split('\n')
    stop_words = set(stopwords1) #{":", "的", "， ", "”"}

    text = []
    for word in jieba.cut(document):
        if word not in stop_words:
            text.append(word)
```

```
        logging.debug(text)


        return text


def calculate_similar_matrix(self):
        """
        计算相似度矩阵及一些必要数据
        """
        words = [self.split_word(document) for document in self.documents]


        self.dictionary = corpora.Dictionary(words)
        corpus = [self.dictionary.doc2bow(word) for word in words]
        self.tfidf = models.TfidfModel(corpus)
        corpus_tfidf = self.tfidf[corpus]
        self.similar_matrix = similarities.MatrixSimilarity(corpus_tfidf)


def get_similar(self, document):
        """
        计算要比较的文档与语料库中每篇文档的相似度
        """
        words = self.split_word(document)
        corpus = self.dictionary.doc2bow(words)
        corpus_tfidf = self.tfidf[corpus]
        return self.similar_matrix[corpus_tfidf]
```

```
#if __name__ == '__main__':

path = 'D:/aaazyw/泰迪/2020-8/C 题全部数据/'

file = path + '附件 3.xlsx'

data=pd.read_excel(file)

documents1=data['留言详情']

documents2=documents1


documents0 = [

        "货运物流供应商 Flexport 完成 10 亿美元融资",

        "一笔 300 亿并购落地，一个新游戏帝国崛起",

        "讯轻科技”累计完成近千万元融资",

        "窝趣公寓完成近 2 亿元 B 轮融资主打品质和轻松社交的居住环境",

        "IBM 的区块链副总裁 JesseLund:比特币将达到 100 万美元",

]


doc_similar = DocumentSimilar(documents1)


#  要比较的文档

#new_doc = "讯轻科技”累计完成近千万元融资" #"窝趣公寓完成近 2 亿元 B 轮融资"

aa=[]

for i in documents1:
```

```
        a=doc_similar.get_similar(i)

        aa.append(a)

'''

new_doc = "讯轻科技”累计完成近千万元融资" #"窝趣公寓完成近2亿元B轮融资
"

for value, document in zip(doc_similar.get_similar(new_doc), documents):

        print("{:.2f}".format(value), document)

'''

ab=aa[2]>0.1 #aa[1]<1

print(sum(ab)-1)



import pickle

#  使用 dump()将数据序列化到文件中

fw = open(path+'dataFile3.dat', 'wb')

# Pickle the list using the highest protocol available.

dataList=[aa,documents1]

pickle.dump(dataList, fw, -1)

# Pickle dictionary using protocol 0.

#pickle.dump(dataDic, fw)

fw.close()



###############

#  使用 load()将数据从文件中序列化读出

fr = open(path+'dataFile.dat', 'rb')
```

```
data1 = pickle.load(fr)

aa=data1[0]

documents1=data1[1]


print(data1)

data2 = pickle.load(fr)

print(data2)

fr.close()
```

五、问题三的计算过程

5.1 问题三的计算理论及方法

问题三：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。

由于任务型对话系统[11]的目标任务明确，完成过程完整可观测，所以本实验希望可以根据对话资源消耗，任务完成满意度等具体指标对任务型对话系统的回复质量进行定性定量分析评价。所以评价思路[11]就一目了然了，通过线性回归的方法对已标注的数据集求出一个可以用来表示用户满意度的权重指标，也就是将最大化任务完成成功率与最小化交互轮数作为任务型对话系统评测的指标。具体的实验流程是：首先对输入的数据进行分词，将分词结果分别进行 Bo W 和 TF-IDF 特征抽取，将抽取得到的结果作为分类模型的训练的输入，将模型输出的结果作为任务完成程度的指标，通过计算对话 session 中的消耗轮数得到对话资源损耗，将完成程度与对话资源损耗进行结合，最终得到用户满意度。这三个指标的计算方法如表 2-9 所示。

指标	取值	指标介绍	计算方法
相关性	0，1	0：任务未完成 1：任务完成	分类模型输出
完整性	N	N：对话 session 的轮数（一问一答为两轮）	读取数据时统计
可解释性	0，1，2	0：任务未完成 1：任务完成但消耗大于 N 2：任务完成且消耗小于 N	通过任务完成程度与对话资源消耗进行计算

本次设计主要参考使用了两种机器学习模型分类方法对任务型对话系统回复质量评价进行了实验，分别是逻辑斯蒂回归分类模型和支持向量机。

逻辑斯蒂回归[7]是一种适用二元分类任务的机器学习模型，一般用来估算某一标签出现的概率。针对输入的数据及特征，利用 Sigmoid 函数逻辑回归模型对输入的数据进行非线性变化将样本为某一标签的概率进行泛化，使其值在 0-1 之间以概率的形式输出，若输出的概率值大于 0.5，则该样本属于正类，若输出概率值小于 0.5，该样本属于负类。

公式中的$P（y=1\backslash x:\theta)$为某一样本属于正标签的概率，公式$P（y=0\backslash x:\theta)$表示某一样本属于负标签的概率。逻辑斯蒂模型中的训练得到的待估计参数为θ。

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

$$P(y=1\backslash x;\theta)=h_\theta(x)$$

$$P(y=0\backslash x;\theta)=1-h_\theta(x)$$

逻辑斯蒂回归模型中，损失函数使用的是对数似然损失函数，下所表示的，通过梯度下降法求解当前模型的最优参数。

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

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

支持向量机[8]（Support Vector Machine，SVM）是一种最大间隔的二元线性分类器，通过有监督的学习方法，将结构风险最小化，降低泛化误差，提升学习能力。

假设一个固定空间内有一定数量的样本且包含他们各自所属的标签，并假设这些样本是线性可分的，那么就可以说能够找到多个平面将这些样本点都按照标签分隔开。为了得到泛化能力最优的模型，我们会希望通过对模型的训练得到一个分隔效果最好的平面。我们将相同标签的样本点中抽取距离这个分隔平面最近的点，那么这个点与平面的距离就是分类的间隔，SVM 会寻找到距离两侧样本点间隔最大的平面作为最终的分隔平面。在优化过程中，求得的分隔平面与离它最近的样本点之间的间隔越大，则认为得到的分类器效果越好。通过这个思维的转换，SVM 面临的学习问题就会被转换为所示的约数最优化问题。

$$\min_{w,b,\xi}\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i$$

s.t.

$$y_i(w\cdot x_i+b)\geq 1-\xi_i$$

$$\xi_i\geq 0$$

其中的 C 为惩罚项参数，C 值增大时，对应将加大对误分类样本的惩罚值，同理当 C 值减小时，对应对误分类样本的惩罚值也会降低。

对于一些线性不可分的问题，SVM 采用的是对特征进行高维映射的方法来将非线性的问题转化解决。

支持向量机模型[9]在处理非线性的高维特征问题时，需要在非线性的高维特征空间中寻找最优的分隔超平面，通过引入拉格朗日乘子来解决凸二次规划问题。目前常用的核函数有四种类型，分别为线性、径向基核函数、Sigmoid、多项式[10]，不同的核函数针对于不同的数据形态会有不同的效果。本实验使用了台湾大学林智仁教授团队开发的 SVM 工具包 Libsvm，选用了径向基核函数作为核函数，通过将输入特征从低维空间映射到高维空间中，构建分类器。

$$K(x,z)=\exp\left(-\frac{\|x-z\|^2}{2\delta^2}\right)$$

5.2 问题三的 anaconda 程序

问题三所用程序如下：

```
#附件 4

#https://blog.csdn.net/mouday/article/details/87921873

#使用 gensim 对中文文本进行相似度计算


import logging

import pandas as pd

import jieba

from gensim import corpora, models, similarities


'''

path='D:/aaazyw/泰迪/2020-8/C 题全部数据/'

file=path+'chineseStopWords.txt'

#gb18030,utf-8,ISO-8859-1,gb2312
```

```
with open(file, 'r', encoding='utf-8') as f:
        try:
                text = f.read()
        except:
                with open(file, 'r', encoding='gb2312') as f:
                        text = f.read()

stopwords1=text.split('\n')
'''



logging.basicConfig(level=logging.DEBUG)
jieba.setLogLevel(logging.INFO)



class DocumentSimilar(object):
        def __init__(self, documents):
                self.documents = documents
                self.dictionary = None
                self.tfidf = None
                self.similar_matrix = None
                self.calculate_similar_matrix()

        @staticmethod
```

```
def split_word(document):
    """
    分词，去除停用词
    """
    path = 'D:/aaazyw/泰迪/2020-8/C 题全部数据/'
    file = path + 'chineseStopWords.txt'
    # gb18030,utf-8,ISO-8859-1,gb2312
    with open(file, 'r', encoding='utf-8') as f:
        try:
            text = f.read()
        except:
            with open(file, 'r', encoding='gb2312') as f:
                text = f.read()

    stopwords1 = text.split('\n')
    stop_words = set(stopwords1) #{":", "的", "， ", "”  "}

    text = []
    for word in jieba.cut(document):
        if word not in stop_words:
            text.append(word)

    logging.debug(text)
```

```
        return text


    def calculate_similar_matrix(self):
        """
        计算相似度矩阵及一些必要数据
        """
        words = [self.split_word(document) for document in self.documents]


        self.dictionary = corpora.Dictionary(words)
        corpus = [self.dictionary.doc2bow(word) for word in words]
        self.tfidf = models.TfidfModel(corpus)
        corpus_tfidf = self.tfidf[corpus]
        self.similar_matrix = similarities.MatrixSimilarity(corpus_tfidf)


    def get_similar(self, document):
        """
        计算要比较的文档与语料库中每篇文档的相似度
        """
        words = self.split_word(document)
        corpus = self.dictionary.doc2bow(words)
        corpus_tfidf = self.tfidf[corpus]
        return self.similar_matrix[corpus_tfidf]
```

```
#if __name__ == '__main__':

path = 'D:/aaazyw/泰迪/2020-8/C 题全部数据/'

file = path + '附件 4.xlsx'

data=pd.read_excel(file)

documents1=data['留言详情']

documents2=data['答复意见']


documents0 = [

        "货运物流供应商 Flexport 完成 10 亿美元融资",

        "一笔 300 亿并购落地，一个新游戏帝国崛起",

        "讯轻科技”累计完成近千万元融资",

        "窝趣公寓完成近 2 亿元 B 轮融资主打品质和轻松社交的居住环境
",

        "IBM 的区块链副总裁 JesseLund:比特币将达到 100 万美元",

]


#doc_similar = DocumentSimilar(documents1)


# 要比较的文档

#new_doc = "讯轻科技”累计完成近千万元融资" #"窝趣公寓完成近 2 亿元 B
轮融资"

aa=[]

for i,item in enumerate(documents1):

    doc_similar = DocumentSimilar([item,'嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯嚯
'])
```

```
        a=doc_similar.get_similar(documents2[i])
        aa.append(a)
    '''
    new_doc = "讯轻科技”累计完成近千万元融资" #"窝趣公寓完成近 2 亿元 B 轮融资"
    for value, document in zip(doc_similar.get_similar(new_doc), documents):
        print("{:.2f}".format(value), document)
    '''
    aan=0
    for i in range(len(aa)):
        if aa[i][0]<0.1:
            aan+=1
    print(aan)


    import pickle
    #  使用 dump()将数据序列化到文件中
    fw = open(path+'dataFile4.dat', 'wb')
    # Pickle the list using the highest protocol available.
    dataList=[aa,documents1]
    pickle.dump(dataList, fw, -1)
    # Pickle dictionary using protocol 0.
    #pickle.dump(dataDic, fw)
    fw.close()
```

```
###############
# 使用 load()将数据从文件中序列化读出
fr = open(path+'dataFile.dat', 'rb')
data1 = pickle.load(fr)
aa=data1[0]
documents1=data1[1]
print(data1)
data2 = pickle.load(fr)
print(data2)
fr.close()
```

六、参考文献

[1] 陈锐，贾晓丰，赵宇.基于模糊聚类的智慧城市多源信息协同结构测度与优化[J].计算机应用研究，2016，33(7):1945-1951.

[2] 宋懿，安小米，范灵俊，马广惠.大数据时代政府信息资源共享的协同机制研究——基于宁波市海曙区政府信息资源中心的案例分析［J］. 情报理论与实践，2018，41(6):64-69.

[3] 马捷，蒲泓宇，张云开，慎镛乐.基于关联数据的政府智慧服务框架与信息协同机制［J］. 情报理论与实践，2018，41(11):20-26.

[4] 罗凯，张明智，吴曦.基于作战环的空间信息时效网关键节点分析模型［J］. 系统工程与电子技术，2016，38(7):1572-1576.

[5] 王宁宁，赵宇，陈锐.基于辐射模型的城市信息空间关联复杂网络研究［J］. 经济地理，2015，35(4):76-83.

[6] 张子柯.在线社交网络信息传播机制与动力学研究综述［J］. 情报学报，2017，36(4):422-431.

[7] 侯贺平，刘艳芳，李纪伟，等.基于改进辐射模型的乡镇人口流动网络研究［J］. 中国人口·资源与环境，2013，23(8):107-115.

[8] 吕鹏辉，刘盛博.学科知识网络实证研究(Ⅳ)合作网络的结构与特征分析［J］. 情报学报，2014，33(4):367-374.

[9] 关鹏，王曰芬，曹嘉君.整合主题的学科知识网络构建与演化分析框架研究［J］. 情报科学，2018，36(9):3-8.

[10] 郭建科，何瑶，侯雅洁.中国沿海集装箱港口航运网络空间联系及区域差异［J］. 地理科学进展，2018，37(11):1499-1509.

[11] 张杨子.面向对话系统回复质量的自动评价研究[D]哈尔滨工业大学,2018