**华中科技大学计算机科学与技术学院**

**机器学习报告**



**专 业： 计算机科学与技术**

**班 级： 计算机1807班**

**学 号： U201814680**

**姓 名： 陈 文 扬**

**成 绩：**

**指导教师： 邹 复 好**

**完成日期： 2020年6月16日**

**目　　录**

[1 绪论 1](#_Toc43223427)

[1.1 声音检测识别技术的背景 1](#_Toc43223428)

[1.2 本次实验的研究重点 1](#_Toc43223431)

[1.3 用于声音检测识别技术的计算机语言 2](#_Toc43223432)

[2 声音特征提取和数据集构成 3](#_Toc43223433)

[2.1 声音特征提取 3](#_Toc43223434)

[2.2 数据集构成 3](#_Toc43223435)

[3 分类器算法 5](#_Toc43223436)

[3.1 高斯朴素贝叶斯分类算法 5](#_Toc43223437)

[3.2 高斯朴素贝叶斯算法实现步骤 5](#_Toc43223440)

[3.3 分类器实现 7](#_Toc43223443)

[4 测试分析和优化设计 10](#_Toc43223446)

[4.1 数据集预处理方法 10](#_Toc43223447)

[4.2 基本性能测试 11](#_Toc43223450)

[4.3 预处理优化设计 12](#_Toc43223453)

[4.4 分类器优化设计 21](#_Toc43223459)

[4.5 优化性能测试 24](#_Toc43223464)

[4.6 结果分析 26](#_Toc43223467)

[5 总结与展望 28](#_Toc43223470)

[5.1 总结 28](#_Toc43223471)

[5.2 展望 28](#_Toc43223472)

[参考文献 30](#_Toc43223473)

[附录 voice.csv数据集参考说明 31](#_Toc43223474)

1 绪论

## 1.1 声音检测识别技术的背景

声音信号在日常生活中分布非常广泛，包含信息量大，也是人类交流的基本方式。声音传播媒介多种多样且分布非常广泛，同时声音信号的获取也较为方便，便于开展声音信号相关方面的研究工作。

### 1.1.1 声音检测识别技术的分类和应用

声音是由物体的振动产生的，并通过传播媒介传播开来。声音检测识别技术根据信号来源和分析要求可分为两个主要类别：一是非语音信号的检测识别技术；二是语音信号检测识别技术。

非语音信号的检测识别技术包括环境声音分析和判别性语音分析，该项技术主要进行定性分析。应用场景包括：环境异响预警、健康状况检测、发声源分析等。

语音信号的检测识别技术主要指人类语音识别技术，该项技术主要进行人类语音信息到文字信息的转换工作，目的是实现人与计算机之间的自然语言通信，隶属于自然语言处理范畴。

### 1.1.2 声音检测识别技术研究现状

非语音信号的检测识别技术的应用面较窄，研究进度相对缓慢。国外主要将声音信号的检测技术用于健康状况的检测中，通过提取呼吸声音信号、分解分析其中的异常声音片段，确定被检测对象的健康状况。国内则主要将该技术用于管道和精密仪器的定损，通过声音传感器提取待测物件的声音信息，综合分析是否存在缺陷和异常。另外，该项技术在人物画像等的刑侦领域也有一定的应用。

语音信号的检测识别技术应用非常广泛，已经渗透到人类生活的各个方面。国外的语言信号研究从20世纪50年代就已经开始，目前已经得到了很大的发展，广泛应用于各类人机对话系统。国内相关研究从20世纪70年代末起步，但相对缓慢，随着计算机在我国的普及，该方面的研究才得到加速，目前我国在中文语音分析技术和语音合成技术方面已经取得了一系列成果。

## 1.2 本次实验的研究重点

本次实验的研究重点是语音信号的发声源分析，隶属于非语音信号的检测识别技术。主要内容是基于声音频谱分析的男/女声判别，主要围绕声音样本的获取和频谱特征的提取、分类器算法设计、测试分析和优化设计三个方面进行。

## 1.3 用于声音检测识别技术的计算机语言

可用于声音检测识别技术的计算机语言相当丰富；例如Matlab、R等语言在统计计算和统计制图方面优势明显，而Java、Python等语言则在分析算法设计上提供快捷的实现可能。可使用的声学分析库也很多；例如Python库pyAudioAnalysis、librosa，R语言的warbleR等。

出于实现效率和便捷性的考虑，本次实验将主要使用Python语言和R语言，主要使用的库有Python图形库matplotlib、线性代数库numpy、数据分析库pandas，R频谱处理库warbleR等。

2 声音特征提取和数据集构成

## 2.1 声音特征提取

声音特征主要包括响度、音频和音色，本次实验研究将仅关注声音的频率特征。准备提取的声音频率特征如表格1所示。

1. 备选声音频率特征

|  |  |  |
| --- | --- | --- |
| 英文学名 | 简记名 | 中文学名 |
| mean frequency | meanfreq | 平均声音频率 |
| standard deviation of frequency | sd | 声音频率标准差 |
| median frequency | median | 声音频率中位数 |
| first quantile | Q25 | 声音频率的第一个四分位数 |
| third quantile | Q75 | 声音频率的第三个四分位数 |
| interquantile range | IQR | 声音频率的四分位间距 |
| skewness | skew | 声音频谱偏度 |
| kurtosis | kurt | 声音频谱峰度 |
| spectral entropy | sp.ent | 声音频谱熵 |
| spectral flatness | sfm | 声音频谱平坦度 |
| mode frequency | mode | 声音频率的主要模态频率 |
| frequency centroid | centroid | 声音频谱质心 |
| average of fundamental frequency measured across acoustic signal | meanfun | 平均基频 |
| minimum fundamental frequency measured across acoustic signal | minfun | 基频最小值 |
| maximum fundamental frequency measured across acoustic signal | maxfun | 基频最大值 |
| average of dominant frequency measured across acoustic signal | meandom | 平均主频 |
| minimum of dominant frequency measured across acoustic signal | mindom | 主频最小值 |
| maximum of dominant frequency measured across acoustic signal | maxdom | 主频最大值 |
| range of dominant frequency measured across acoustic signal | dfrange | 主频测量范围 |
| modulation index | modindx | 调制指数 |

使用R频谱处理库warbleR可以便捷地从无损音频格式文件（.wav）中提取上述音频特征，并转存为逗号分隔值文件（.csv）。

## 2.2 数据集构成

本次实验的主要音频用例来自The Harvard-Haskins Database of Regularly-Timed Speech、Telecommunications & Signal Processing Laboratory (TSP) Speech Database at McGill University、VoxForge Speech Corpus以及Festvox CMU\_ARCTIC Speech Database at Carnegie Mellon University。

考虑到存储容量问题，本次实验总计选取3168份音频文件，使用warbleR对这些音频文件前20秒的录制内容进行特征提取，生成数据集。按照测试集占比0.3的规格进行100次随机划分，生成用于交叉验证的100组训练/测试集。

数据集存储采用.csv格式，包含如表格1所示的全部声音频率特征，并在最后一栏填入标签（male/female），用以标记该行数据所对应的声音性别。

由于数据集主要来自于演讲录音，并不能覆盖人类发声的所有模式；因此，作为测试集的补充，将使用录音软件录制5份男声音频文件、5份女声音频文件，并使用warbleR提取这10份测试用例的音频特征，作为额外的测试用例。额外测试用例说明如表格2所示。

1. 额外测试用例说明

|  |  |  |
| --- | --- | --- |
| 编号 | 声音性别 | 说明 |
| 1 | male | 男声中速朗读 |
| 2 | male | 男声快速朗读 |
| 3 | male | 男声日常交际 |
| 4 | male | 男中音歌唱 |
| 5 | male | 假声男高音歌唱 |
| 6 | female | 女声中速朗读 |
| 7 | female | 女声快速朗读 |
| 8 | female | 女声日常交际 |
| 9 | female | 女中音歌唱 |
| 10 | female | 女声低音吟唱 |

3 分类器算法

## 3.1 高斯朴素贝叶斯分类算法

高斯朴素贝叶斯（Gaussian Naïve Bayes）分类算法是基于频率统计的分类技术中最常用的方法之一。该分类算法具有形式简单，耗时短，抗过拟合等显著优点。

### 3.1.1 高斯朴素贝叶斯算法基本假设

高斯朴素贝叶斯分类算法基于频率统计，采用极大似然估计法。

条件无关假设。朴素贝叶斯算法假设对于某确定标签，各特征取值的条件概率条件无关。

连续性假设。认为各特征的量化值是连续的，且在样本数量充分大的条件下呈现正态分布。

### 3.1.2 高斯朴素贝叶斯算法决策原理

假设受试样本具有特征向量，标签集合，根据贝叶斯公式和朴素贝叶斯假设计算条件概率，选取最大条件概率对应的标签作为预测标签，即。

## 3.2 高斯朴素贝叶斯算法实现步骤

### 3.2.1 参数学习

分类器需要学习的参数包括先验概率、不同标签下各项特征值的统计平均值和方差。

假设有个特征，训练集有个特征向量，标签集合，表示特征向量的第维分量，表示所有标签为的样本第维分量的平均值，表示所有标签为的样本第维分量的方差，表示标签出现的次数，表示特征向量的标签。

不难得到：

假设各标签的先验概率为，则有：

这样就完成了参数学习，时间复杂度为。

### 3.2.2 标签预测

变量定义同3.2.1，假设受试样本具有特征向量，根据贝叶斯公式，不难得到：

根据朴素贝叶斯假设，得到：

根据中心极限定理，得到：

从而有概率密度函数：

改用概率密度函数表示，可得：

对于确定的参数模型和确定的，分母为一定值，记为，该值无需计算。

展开表达式可得：

对于测试向量，分类器选择最大的条件概率对应的标签作为预测值，即预测值。

这样就完成了标签预测，时间复杂度为。

## 3.3 分类器实现

### 3.3.1 数据结构实现

分类器主要使用的数据结构是numpy库提供的array矩阵结构和pandas库提供的dataFrame数据框架结构。

分类器封装为Python类对象GaussianNaiveBayesClassfier，其类成员及其说明如表格3所示。

1. GaussianNaiveBayesClassfier类成员及其说明

|  |  |  |
| --- | --- | --- |
| 类成员 | 类型 | 说明 |
| prior | numpy.array | 先验概率矩阵，各类标签占比 |
| averages | numpy.array | 各类均值矩阵，每个行向量包含某个类所有特征的均值 |
| variances | numpy.array | 各类方差矩阵，每个行向量包含某个类所有特征的方差 |
| n\_class | int | 类别数量 |

在Python实现中将类成员定义在初始化方法\_\_init\_\_中，如下所示：

|  |
| --- |
| def \_\_init\_\_(self):  self.prior = None # 先验概率矩阵  self.averages = None # 各类均值矩阵  self.variances = None # 各类方差矩阵  self.n\_class = None # 类别数量 |

### 3.3.2 算法实现

根据算法设计，首先进行参数学习。

使用dataFrame的value\_counts方法获得各标签计数，将各标签计数除以总样本数即得先验概率矩阵prior。

实现如下：

|  |
| --- |
| def obtainPrior(self, dataFrame):  # 获取计数矩阵  labelCounts = np.array(dataFrame.iloc[:, -1].value\_counts()).astype(np.float)  prior = labelCounts / dataFrame.iloc[:, -1].size  return prior |

使用dataFrame的mean和var方法分别获得各标签下各特征值的平均数及方差。使用列表生成器实现如下：

|  |
| --- |
| """  方法说明:计算均值矩阵  均值矩阵  format:  [[<----Avg(label==0)---->]  [<----Avg(label==1)---->]  ...  [<----Avg(label==n-1)---->]]  """  def obtainAverages(self, dataFrame):  return np.array([dataFrame[dataFrame.iloc[:,-1]==i].mean(axis=0).iloc[0:-1]  for i in range(self.n\_class)])  """  方法说明:计算方差矩阵  方差矩阵  format:  [[<----Var(label==0)---->]  [<----Var(label==1)---->]  ...  [<----Var(label==n-1)---->]]  """  def obtainVariances(self,dataFrame):  return np.array([dataFrame[dataFrame.iloc[:, -1] == i].var(axis=0).iloc[0:-1]  for i in range(self.n\_class)]) |

然后导入测试集进行标签预测。

根据正态分布概率密度公式和朴素贝叶斯假设，计算分项特征条件概率密度和联合条件概率密度矩阵。

实现中无需计算常数，实现如下：

|  |
| --- |
| """  方法说明:计算条件概率矩阵  Parameters:  vector - 测试向量  Returns:  条件概率矩阵（非严格定义的条件概率，采用不带系数的概率密度矩阵代替）  format:  [P(vector|label==0) P(vector|label==1) ... P(vector|label==n-1)]  """  def obtainProbDensity(self, vector):  # 高斯分布下的各类各特征概率密度(不带1/sqrt(2\*pi)系数)矩阵  density = np.exp(-(vector - self.averages) \*\* 2 / (2 \* self.variances))  / np.sqrt(self.variances)  # 贝叶斯独立假设下的各类条件概率密度矩阵(每行特征概率密度求积)  unidensity = density.prod(axis=1)  # 处理被约为0的数  unidensity[unidensity==0.0] = 2.0 \*\* (-1023)  return unidensity |

根据算法设计，对每一个测试向量，分别计算对每种标签的条件概率。

实现中无需计算常数并且在最后进行总和归一化，实现如下：

|  |
| --- |
| """  方法说明:计算预测概率矩阵  Parameters:  testFrame - 测试数据集  Returns:  jointProb - 预测概率矩阵（非严格定义的概率，不计算相同的分母）  format:  [[P(label==0|vector0) P(label==1|vector0) ... P(label==n-1|vector0)]  [P(label==0|vector1) P(label==1|vector1) ... P(label==n-1|vector1)]  ...  [P(label==0|vector(n-1)) P(label==1|vector(n-1)) ... P(label==n-1|vector(n-1))]]  """  def predictProb(self, testFrame):  # testFrame测试数据的条件概率密度矩阵  probDensity = np.apply\_along\_axis(self.obtainProbDensity, axis=1,  arr=testFrame.values)  # (标签,特征)联合概率密度矩阵  jointProb = self.prior \* probDensity  # 总和归一化的(标签,特征)联合概率密度矩阵  jointProb = jointProb / jointProb.sum(axis=1)[:,None]  return jointProb |

分类器选择最大的条件概率对应的标签作为预测值，在预测概率矩阵的基础上，取jointProb.argmax(axis=1)即得测试集的标签矩阵

4 测试分析和优化设计

## 4.1 数据集预处理方法

实验使用的数据集命名为voice.csv，其中的缺省值填充0.0，标签有“male”“female”两种。

### 4.1.1 数据集划分

按照训练集:测试集=7:3的比例划分数据集，使用sklearn库提供的train\_test\_split方法，然后将划分好的数据集导出为“.csv”格式，命名规范为：训练集—voice\_train\_n.csv、测试集—voice\_test\_n.csv（n=1,2,…）。

数据集划分函数实现如下：

|  |
| --- |
| """  函数说明:数据集生成，读取所有数据，划分成若干份训练/测试数据csv文件  Parameters:  csvFile - 原始数据集csv文件名 default:'voice/voice.csv'  num - 生成的训练/测试数据csv文件数量 default:1  """  def createDataset(csvFile='voice/voice.csv', num=1):  df = pd.read\_csv(csvFile) # 读取csv-->dataFrame  trainFrames, testFrames = divideDataset(df, num) # 划分num份训练/测试集  for i in range(len(trainFrames)): # 写入训练/测试集dataFrame-->csv  trainFrames[i].to\_csv('voice/voice\_train\_%d.csv' % (i + 1), index=False)  testFrames[i].to\_csv('voice/voice\_test\_%d.csv' % (i + 1), index=False) |

### 4.1.2 数据整理

数据整理的规则是：

（1）对于训练集中含有缺省值的数据行，在导入后删除。

（2）对于测试集中含有缺省值的数据行，在导入后填充缺省值对应特征的平均值。

（3）标签列的“male”转换为数值1，“female”转换为数值0。

数据整理函数实现如下：

|  |
| --- |
| """  函数说明:处理缺省值和标签数值化  Parameters:  dataFrame - 解析的数据集  dicard - 舍弃带有缺省值的行或填充平均值 default:False  """  def fixDeafults(dataFrame, discard=False):  for index, row in dataFrame.iterrows():  for i in range(len(row)-1):  if row[i] == 0.0: # 处理缺省值  dataFrame.iloc[index, i] = np.nan # 缺省值统一置为NaN  # 标签二值化  if row[-1] == 'male': dataFrame.iloc[index, -1] = 1 # male标签置1  else: dataFrame.iloc[index, -1] = 0 # female标签置0  # 根据参数dicard处理  if discard: dataFrame.dropna(axis=0, inplace=True) # 舍弃带缺省值的行  else:  dataFrame.fillna(dataFrame.mean(), inplace=True) # 缺省值填充均值  dataFrame.dropna(axis=1, inplace=True) # 舍弃带缺省值的列 |

调用fixDeafults作如下数据整理：

|  |
| --- |
| df\_train = loadDataset(train\_csv)  df\_test = loadDataset(test\_csv)  fixDeafults(df\_train, discard=True)  fixDeafults(df\_test, discard=False) |

## 4.2 基本性能测试

### 4.2.1 主数据集测试结果

调用createDataset(csvFile='voice/voice.csv', num=50)生成50份随机划分的训练集/测试集。使用分类器GaussianNaiveBayesClassfier进行分类，得到分类结果，如下：

|  |
| --- |
| ------------测试总次数=50------------  累积混淆矩阵：  男声 女声  预测男声 22004 2750  预测女声 1722 21074  总男声比例：49.90%  最大准确度：92.74%  最低准确度：87.07%  平均准确度：90.60%  准确度方差：0.000140343  平均男声查准率：88.89%  平均男声查全率：92.74%  平均女声查准率：92.45%  平均女声查全率：88.46%  平均男声似然比：8.0345  平均女声似然比：12.1878  判别男声相关：  平均诊断比值比：97.9226  平均F1分数：0.9078 |

可见，高斯朴素贝叶斯分类器的平均准确度可达90%，且在男声查全率和女声查准率上相对更具优势。

### 4.2.2 额外用例测试结果

使用数据集voice.csv进行训练，对附加的10份测试用例进行分类预测。

测试结果细节如表格4所示。

1. 额外用例测试结果细节

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 声音性别 | 说明 | 预测相对概率 | | 预测结果 |
| 女声 | 男声 |
| 1 | male | 男声中速朗读 | 0.000432 | 0.999 | male |
| 2 | male | 男声快速朗读 | 0.00677 | 0.993 | male |
| 3 | male | 男声日常交际 | 0.00394 | 0.996 | male |
| 4 | male | 男中音歌唱 | ≈1 | 0.00000406 | female |
| 5 | male | 假声男高音歌唱 | 0.515 | 0.485 | female |
| 6 | female | 女声中速朗读 | ≈0 | ≈1 | male |
| 7 | female | 女声快速朗读 | 0.994 | 0.00585 | female |
| 8 | female | 女声日常交际 | 0.0000403 | ≈1 | male |
| 9 | female | 女中音歌唱 | ≈0 | ≈1 | male |
| 10 | female | 女声低音吟唱 | ≈0 | ≈1 | male |

对额外用例的分类效果非常不理想，准确度仅有40%。且对于本应较易判断的“女声中速朗读”项却以极高概率判别为男声。简单分析认为，由于这些用例取自实际环境下的录音，噪音更大，因此对预测产生了很大干扰。

## 4.3 预处理优化设计

人们主要依赖声音的音调来辨别说话者的性别，而音调取决于基频，个体的声带大小又影响着基频的高低。由于男性的声带普遍大于女性和儿童，导致男性声音的基频比女性声音低[[1]](#footnote-1)。由此推知，基频统计平均值meanfun在声音的分类中起主要作用，并且各种声音频率特征对声音分类的重要性也不尽相同。

另一方面，由于数据集主要是从演讲录音片段提取的，而非来自于专业的声音测试，所以数据集的噪音实则偏大。特征类别的增加势将引入更多的噪音。

为了突出最重要的语音性别特征，同时尽可能减少噪音，优化设计将从主特征选择和降噪方法着手。

### 4.3.1 特征值在分类中的重要性

根据一项利用语音基音统计特征的说话人性别判识的研究[[2]](#footnote-2)，单纯使用基音频率进行语音性别判识的准确率就已经可达89%以上。然而在此前的基本性能测试中，使用的频率特征数有20个，最低准确率却仅有87%，因此有必要重新审视特征选择问题。可以预见基音频率对语音性别判识的帮助非常大；相对的，在噪音偏大的条件下，某些其他特征对语音性别判识的帮助很小，甚至起反作用。

观察图1a、b所示的特征Q75、meanfun的分布情况，发现特征Q75的性别差异相关性很低，而特征meanfun性别相关性显著。

(a) 声音频率第三个四分位点Q75的分布情况

(b) 平均基频meanfun的分布情况

1. 特征分布情况

将图1所示的直观现象量化，对特征值在分类中的重要性的量化评估可以采用如下几种指标：类间相对均值差、类间方差、梯度提升决策树统计特征等。

### 4.3.2 基于类间相对均值差的主特征选择

对于二分类问题，假设有个特征，训练集有个特征向量，标签集合，表示特征向量的第维分量，所有维数均已做归一化处理，表示所有标签为的样本第维分量的平均值，表示所有标签为的样本第维分量的方差。表示标签出现的次数，表示特征向量的标签。

不难得到：

所有特征向量第维分量的类间相对均值差的定义是：

假设需要提取的主特征数为，则取最大的个对应的特征作为主特征。

对于本实验，实现如下：

|  |
| --- |
| # stdTrainFrame----最大最小值归一化的训练集  if method == 'mean': # 以类间相对均值差最大为标准提取主要特征  # 获取male各项特征均值列表  maleSampleMean = list(stdTrainFrame[stdTrainFrame.iloc[:,-1]==1]  .mean(axis=0).iloc[0:-1])  # 获取female各项特征均值列表  femaleSampleMean = list(stdTrainFrame[stdTrainFrame.iloc[:,-1]==0]  .mean(axis=0).iloc[0:-1])  # 求类均值差列表  different = [abs(maleSampleMean[i] - femaleSampleMean[i]) /  min(maleSampleMean[i], femaleSampleMean[i])  for i in range(len(maleSampleMean))]  # 获取降序特征索引  index = list(np.argsort(different)[-1:-feature\_num-1:-1])  index.append(-1) # 分类标签附加在最后  colName = stdTrainFrame.columns[index] # 获取对应索引值的列标签 |

### 4.3.3 基于类间方差的主特征选择

变量定义同4.3.2。记，。

所有特征向量第维分量的类内方差的定义是：

所有特征向量第维分量的类间方差的定义是：

总方差、类内方差、类间方差的关系是：

假设需要提取的主特征数为，则取最大的个对应的特征作为主特征。

对于本实验，实现如下：

|  |
| --- |
| # stdTrainFrame----最大最小值归一化的训练集  if method == 'var': # 以类间方差最大为标准提取主要特征  # 获取male各项特征均值列表  maleSampleMean = list(stdTrainFrame[stdTrainFrame.iloc[:, -1] ==  1].mean(axis=0).iloc[0:-1])  # 获取female各项特征均值列表  femaleSampleMean = list(stdTrainFrame[stdTrainFrame.iloc[:, -1] ==  0].mean(axis=0).iloc[0:-1])  # 对male/female标签计数  classWeight = np.array(stdTrainFrame.iloc[:, -1].value\_counts()).prod() /  stdTrainFrame.iloc[:, -1].size  # 求类间方差列表  different = [classWeight \* (maleSampleMean[i] - femaleSampleMean[i]) \*\* 2  for i in range(len(maleSampleMean))]  #print(different)  # 获取降序特征索引  index = list(np.argsort(different)[-1:-feature\_num - 1:-1])  index.append(-1) # 分类标签附加在最后  colName = stdTrainFrame.columns[index] # 获取对应索引值的列标签 |

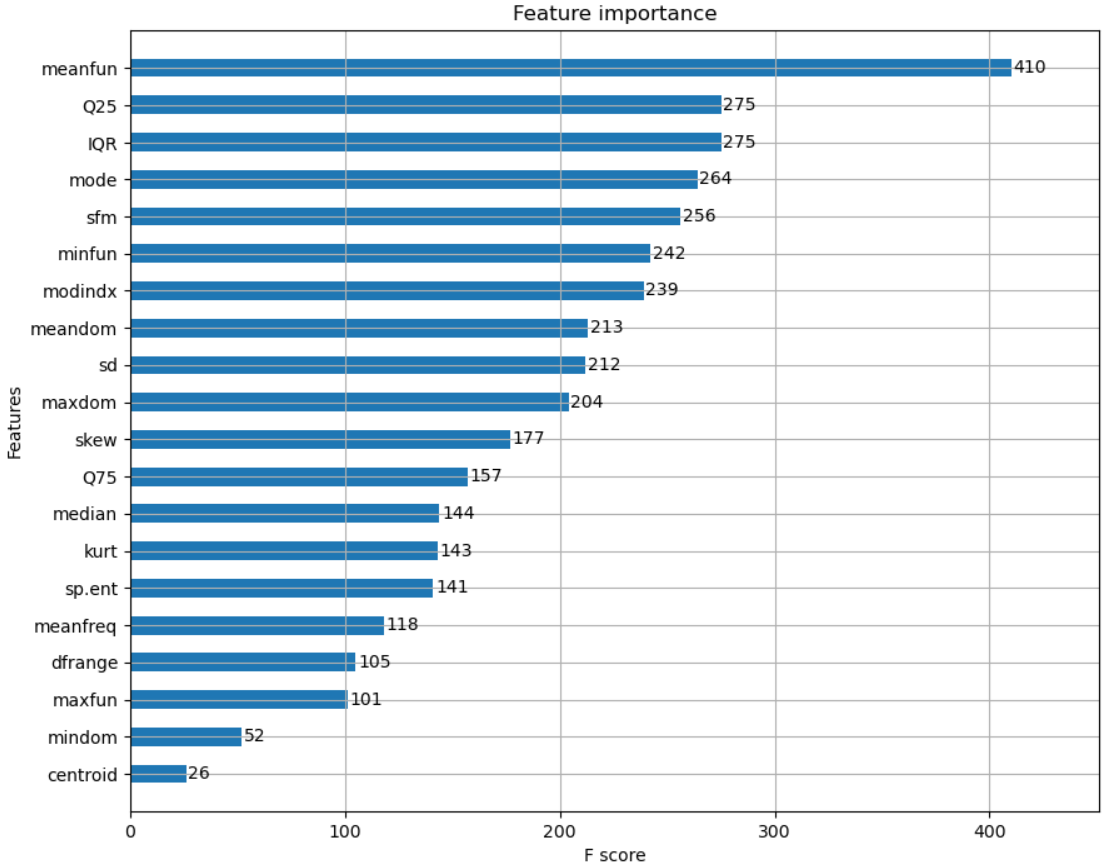
### 4.3.4 基于梯度提升决策树统计特征的主特征选择

基于梯度提升决策树的主特征选择是利用决策树的节点统计特征来指导主特征选择的方法。

（1）基于特征权重

特征权重是指在所有树中某一特征被用来分裂数据的次数。

对主数据集voice.csv作特征权重分析，可得特征重要性统计图，如图2所示。



1. 基于特征权重的特征重要性统计图

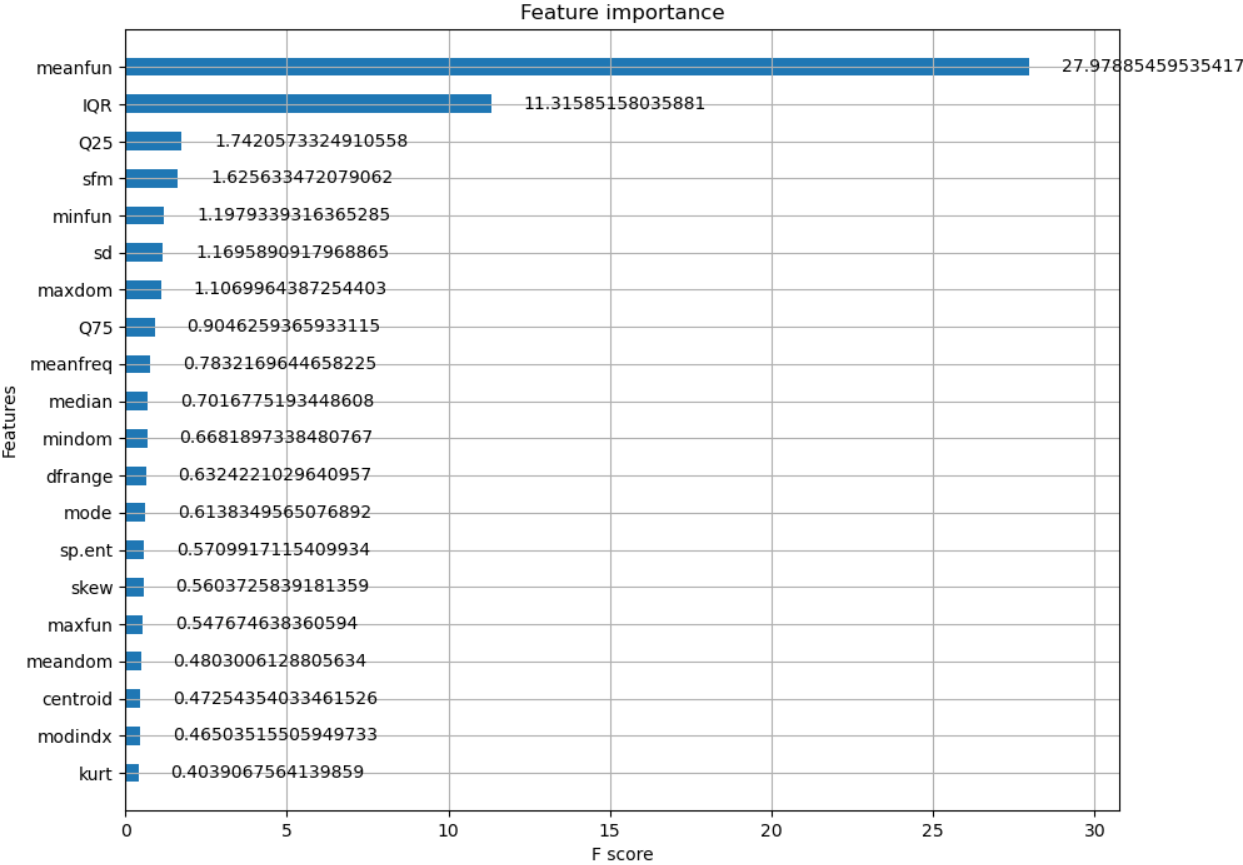
对于本实验，使用xgboost库实现如下：

|  |
| --- |
| if method == 'boost\_weight': # 以决策树特征权重为标准提取主要特征  model = XGBClassifier(learning\_rate =0.1, n\_estimators=1000, max\_depth=20,  min\_child\_weight=1, gamma=0, subsample=0.8, colsample\_bytree=0.8,  objective='binary:logistic', nthread=4, scale\_pos\_weight=1, seed=27)  model.fit(trainFrame.iloc[:, :-1], trainFrame.iloc[:, -1])  imp\_dict = model.get\_booster().get\_score(importance\_type='weight')  # 特征重要性降序排序  imp\_weight = pd.Series(imp\_dict).sort\_values(ascending=False)  tmpCol = list(imp\_weight.index) # 获取降序排序后的列标签  # 将未使用的标签附加在最后  tmpCol += [tok for tok in colName if not (tok in set(tmpCol))]  # 取出前feature\_num个有效主特征  colName = tmpCol[:feature\_num] + ['label'] |

（2）基于特征增益

特征增益是指某一特征用于分裂的平均增益，即训练损失减少量的平均值。

对主数据集voice.csv作特征增益分析，可得特征重要性统计图，如图3所示。



1. 基于特征增益的特征重要性统计图

对于本实验，使用xgboost库实现如下：

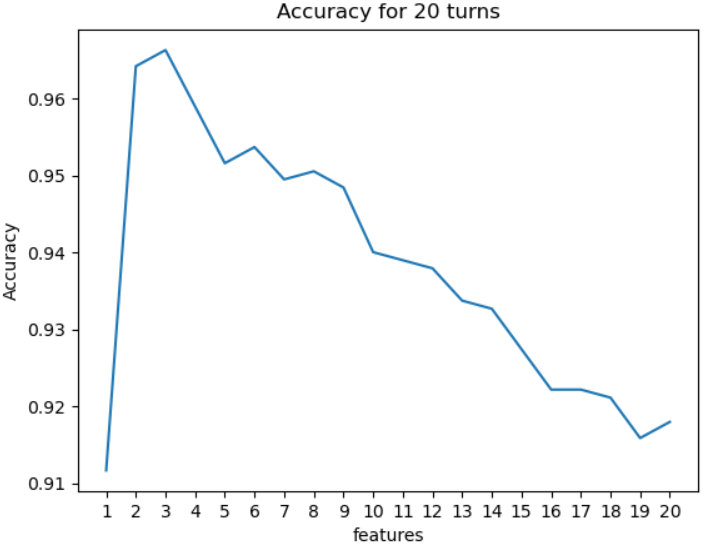
|  |
| --- |
| if method == 'boost\_gain': # 以决策树特征增益为标准提取主要特征  model = XGBClassifier(learning\_rate =0.1, n\_estimators=1000, max\_depth=20,  min\_child\_weight=1, gamma=0, subsample=0.8, colsample\_bytree=0.8,  objective= 'binary:logistic', nthread=4, scale\_pos\_weight=1, seed=27)  model.fit(trainFrame.iloc[:, :-1], trainFrame.iloc[:, -1])  imp\_dict = model.get\_booster().get\_score(importance\_type='gain')  # 特征重要性降序排序  imp\_gain = pd.Series(imp\_dict).sort\_values(ascending=False)  tmpCol = list(imp\_gain.index) # 获取降序排序后的列标签  # 将未使用的标签附加在最后  tmpCol += [tok for tok in colName if not (tok in set(tmpCol))]  # 取出前feature\_num个有效主特征  colName = tmpCol[:feature\_num] + ['label'] |

### 4.3.5 进行主特征选择的性能测试

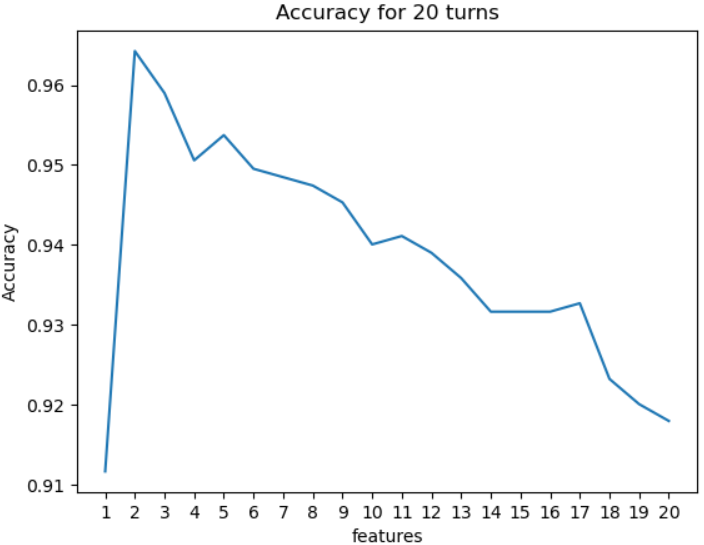
（1）初步确定各主特征选择方式的最佳提取数量

使用生成的第1份训练/测试集voice\_train\_1.csv/voice\_test\_1.csv进行该项测试。

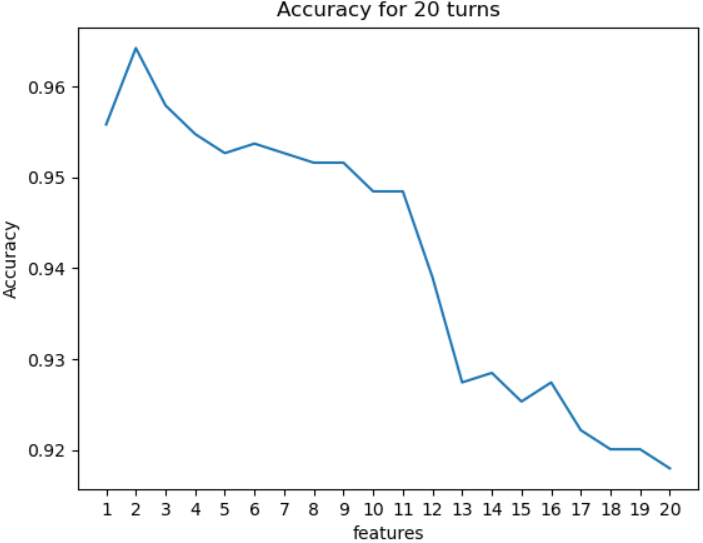
测试结果的主特征数-准确度统计图如图4、5、6、7所示。



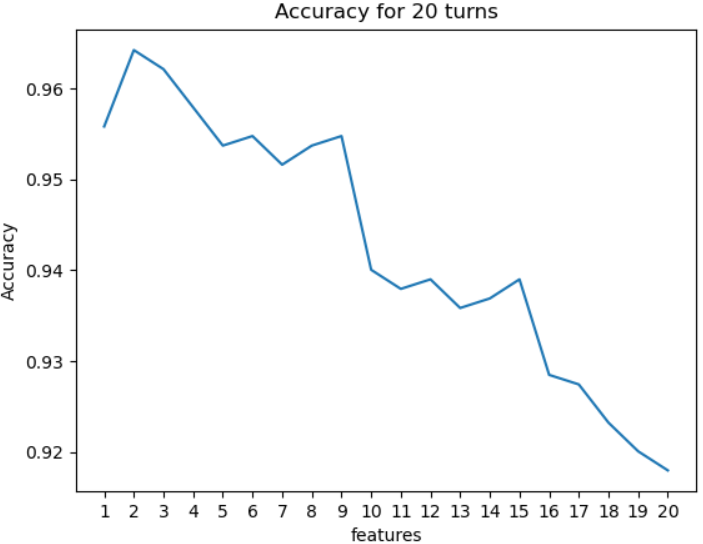
1. 类间相对均值差提取主特征数-准确度统计图



1. 类间方差提取主特征数-准确度统计图



1. 决策树特征权重提取主特征数-准确度统计图



1. 决策树特征增益提取主特征数-准确度统计图

统计输出如下：

|  |
| --- |
| ------------模式准确度列表------------  In mode: method = 'mean'  模式最大准确度：96.64%  with ['IQR', 'meanfun', 'mindom', 'label']  模式最低准确度：91.17%  模式平均准确度：93.83%  模式准确度方差：0.000264827  -------------------------------------  In mode: method = 'var'  模式最大准确度：96.42%  with ['IQR', 'meanfun', 'label']  模式最低准确度：91.17%  模式平均准确度：93.87%  模式准确度方差：0.000185523  -------------------------------------  In mode: method = 'boost\_weight'  模式最大准确度：96.42%  with ['meanfun', 'IQR', 'label']  模式最低准确度：91.80%  模式平均准确度：94.10%  模式准确度方差：0.000227101  -------------------------------------  In mode: method = 'boost\_gain'  模式最大准确度：96.42%  with ['meanfun', 'IQR', 'label']  模式最低准确度：91.80%  模式平均准确度：94.27%  模式准确度方差：0.000196757  -------------------------------------  最大准确度：96.64% |

分析图4、5、6、7以及统计结果可知：

<1>随提取主特征数的增加，总体上准确度呈现先增后减的趋势。

<2>当提取主特征数为2或3时，各提取方式下可以达到最大准确度。

<3>准确率较高时，特征数集中在2~9范围内。

可以推测：

<1>特征数大于9后，噪音干扰变大，导致正确率明显下降。

<2>平均基频meanfun和四分位间距IQR这两个特征对分类的帮助很大。

选取每种主特征选择方式准确率前4位对应的提取数量作为下一轮测试的参考数据：

类间相对均值差：2、3、4、6。

类间方差：2、3、4、5。

决策树特征权重：1、2、3、4。

决策树特征增益：1、2、3、4。

（2）提取方式/数量相关性能测试

使用4.2.1生成的50份随机划分的训练集/测试集。根据（1）的结果使用前述4种主特征选择方式分别进行主特征选择，然后使用分类器GaussianNaiveBayesClassfier进行分类，统计结果如表格5所示。

1. 提取方式/数量相关性能测试统计结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 提取数量 | 提取方式 | 平均准确度 | 最大准确度 | 最低准确度 | 准确度方差 |
| 20（对照） | 无 | 90.60% | 92.74% | 87.07% | 1.40343\*10-4 |
| 1 | 决策树特征权重 | 95.23% | 96.64% | 93.80% | 3.90871\*10-5 |
| 决策树特征增益 | 95.23% | 96.64% | 93.80% | 3.90871\*10-5 |
| 2 | 类间相对均值差 | 96.58% | 97.79% | 95.27% | 2.43790\*10-5 |
| 类间方差 | 96.58% | 97.79% | 95.27% | 2.43790\*10-5 |
| 决策树特征权重 | 96.27% | 97.79% | 93.80% | 5.59137\*10-5 |
| 决策树特征增益 | 96.55% | 97.48% | 95.27% | 2.49889\*10-5 |
| 3 | 类间相对均值差 | 96.30% | 97.69% | 95.27% | 2.22030\*10-5 |
| 类间方差 | 95.97% | 97.48% | 94.85% | 2.25316\*10-5 |
| 决策树特征权重 | 96.21% | 97.58% | 94.43% | 4.17253\*10-5 |
| 决策树特征增益 | 96.29% | 97.58% | 95.27% | 2.30889\*10-5 |
| 4 | 类间相对均值差 | 95.86% | 97.27% | 95.06% | 2.03843\*10-5 |
| 类间方差 | 95.41% | 96.95% | 94.22% | 2.95639\*10-5 |
| 决策树特征权重 | 96.07% | 97.48% | 94.85% | 3.43211\*10-5 |
| 决策树特征增益 | 95.91% | 97.37% | 94.74% | 2.93237\*10-5 |
| 5 | 类间方差 | 94.91% | 96.64% | 93.80% | 3.54268\*10-5 |
| 6 | 类间相对均值差 | 94.89% | 96.95% | 93.80% | 4.29898\*10-5 |

分析表格5可知：

<1>总体上准确度维持在较高水平。

<2>平均准确度最大值在“提取数量2、类间相对均值差/类间方差提取方式”下取得。

<3>最大准确度最大值在“提取数量2、类间相对均值差/类间方差/决策树特征权重提取方式”下取得。

<4>最大准确度最大值在“提取数量2、类间相对均值差/类间方差/决策树特征增益提取方式”“提取数量3、类间相对均值差/决策树特征增益提取方式”下取得。

<5>准确度方差最小值在“提取数量4、类间相对均值差提取方式” 下取得。

<6>综合各项指标，“提取数量2、类间相对均值差/类间方差提取方式”是目前最优，相对于无预处理优化时的结果有约6.2%的准确率提升。

## 4.4 分类器优化设计

在预处理优化设计中提到，进行主特征选择能够有效降低噪音干扰以提高分类性能；同样地，在分类器内部也可以进一步提取特征，但这种特征选择的依据是验证集的准确度等测试指标。

### 4.4.1 动态降噪

首先，按照一定比例从训练集中划分出验证集。

在每一个迭代步骤中，设当前有效特征集合为，选取重要性最低的特征，构造集合。分别在和的条件下训练分类器，根据分类器在验证集上的性能表现决定是否从中删除特征。

简单起见，在本次实验中仅使用准确度指标。如果在条件下训练的分类器具有较高准确度，则从中删除特征；否则，不删除特征，进行下一轮迭代。

### 4.4.2 集成

前述动态降噪设计中只使用了一组验证集，验证集的划分具有一定偶然性，为了降低由验证集引入的偶然因素的影响，可以集成分类器。

以不同方式划分出若干验证集，在动态降噪模式下分别训练数个分类器，在预测标签时，综合各分类器给出的条件概率矩阵给出预测结果。

集成分类器的主要好处在于稳定分类器性能，减小泛化误差中的方差项，可能提升准确度。

### 4.4.3 分类器优化实现

（1）添加动态降噪和集成相关的类成员

|  |
| --- |
| """  方法说明:初始化  Parameters:  has\_denoise - 使用/不使用动态降噪 default:False  bagging\_rate - 附加分类器数量 default:0  bagging\_weakness - 附加分类器学习强度（1-训练集比例） default:0.3  """  def \_\_init\_\_(self, has\_denoise=False, bagging\_rate=0, bagging\_weakness=0.3):  self.prior = None # 先验概率矩阵  self.averages = None # 各类均值矩阵  self.variances = None # 各类方差矩阵  self.n\_class = None # 类别数量  self.denoise = has\_denoise # 动态降噪开关  self.ignore\_dim = None # 降噪后忽略的维数列表  self.mix = bagging\_rate # 附加分类器数量  self.bagging\_classifiers = None # 附加分类器  self.bagging\_weakness = bagging\_weakness # 附加分类器强度 |

（2）实现动态降噪

|  |
| --- |
| """  方法说明:动态降噪训练  Parameters:  trainFrame - 训练集  regulateFrame - 验证集  """  def fitDenoise(self, trainFrame, regulateFrame):  self.ignore\_dim = []  delCol = -2 # 从除去最后的分类标签列的倒数第一列开始删除特征  for i in range(trainFrame.shape[1] - 1):  # 计算未删除当前特征时的验证集正确率  self.setAll(trainFrame)  subPredict = self.predictProb(regulateFrame.iloc[:, 0:-1]).argmax(axis=1)  subRef = regulateFrame.iloc[:, -1].values  diff = subPredict - subRef  totGood = sum(diff == 0)  decTrainFrame = trainFrame.drop(trainFrame.columns[delCol], axis=1)  # 计算删除当前特征后的验证集正确率  self.ignore\_dim.append(trainFrame.columns[delCol])  self.setAll(decTrainFrame)  subPredict = self.predictProb(regulateFrame.iloc[:, 0:-1]).argmax(axis=1)  subRef = regulateFrame.iloc[:, -1].values  diff = subPredict - subRef  totGoodDel = sum(diff == 0)  if totGood < totGoodDel: # 删除后正确率更高时  # 去除该特征  trainFrame = trainFrame.drop(trainFrame.columns[delCol], axis=1)  else: # 删除前正确率更高时  delCol = delCol - 1 # 考察前一个特征  self.ignore\_dim.pop(-1) # 回溯到前一状态  self.setAll(trainFrame) # 去除特征完毕后重新训练 |

（3）实现集成训练和预测

|  |
| --- |
| """  方法说明:集成  Parameters:  trainFrame - 训练集  """  def fitBagging(self, trainFrame):  self.bagging\_classifiers = []  for i in range(self.mix): # 创建附加分类器  self.bagging\_classifiers.append(GaussianNaiveBayesClassfier(  has\_denoise=self.denoise, bagging\_rate=0))  for classifier in self.bagging\_classifiers: # 根据分类器强度划分训练/验证集  splitedTrainFrame, nouseFrame = train\_test\_split(trainFrame.copy(),  test\_size=self.bagging\_weakness)  classifier.fit(splitedTrainFrame) |
| """  方法说明:计算预测矩阵  Parameters:  testFrame - 测试集  format - 返回格式  format: 'label' - 返回预测标签矩阵  'prob' - 返回预测概率矩阵  """  def predict(self, testFrame, format='label'):  probArr = self.predictProb(testFrame)  if self.mix > 0: # 累加各分类器返回的预测概率矩阵  for classifier in self.bagging\_classifiers:  probArr = probArr + classifier.predictProb(testFrame)  #print(classifier.predictProb(testFrame))  #print(probArr)  if format == 'label':  return probArr.argmax(axis=1) # 返回预测标签矩阵  elif format == 'prob':  return probArr / probArr.sum(axis=1)[:,None] # 返回预测概率矩阵  else:  print('format not matched, only \'label\' or \'prob\' is valid')  exit(-1) |

### 4.4.4 优化分类器性能测试

使用4.2.1生成的50份随机划分的训练集/测试集。不进行主特征选择，直接使用分类器GaussianNaiveBayesClassfier进行分类：

<1>开启动态降噪has\_denoise=True。

<2>设置集成附加分类器数量bagging\_rate。

<3>设置集成分类器弱度bagging\_weakness

统计结果如表格6所示。

1. 动态降噪模式下集成数量/弱度相关性能测试统计结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 附加分类器数量 | 分类器弱度 | 平均准确度 | 最大准确度 | 最低准确度 | 准确度方差 |
| 0 | / | 94.32% | 96.21% | 91.80% | 1.27685\*10-4 |
| 5 | 0.1 | 94.46% | 96.42% | 92.74% | 4.57514\*10-5 |
| 0.3 | 94.16% | 96.00% | 92.01% | 6.58664\*10-5 |
| 0.5 | 93.71% | 95.48% | 92.11% | 5.97651\*10-5 |
| 9 | 0.1 | 94.46% | 97.48% | 92.85% | 6.30521\*10-5 |
| 0.3 | 94.14% | 95.69% | 92.43% | 4.46068\*10-5 |
| 0.5 | 93.77% | 96.00% | 91.80% | 7.52339\*10-5 |
| 13 | 0.1 | 94.52% | 96.21% | 93.17% | 4.18239\*10-5 |
| 0.3 | 94.11% | 95.79% | 92.74% | 4.56651\*10-5 |
| 0.5 | 93.68% | 95.58% | 91.80% | 6.37611\*10-5 |

无优化对照组：

|  |  |  |  |
| --- | --- | --- | --- |
| 平均准确度 | 最大准确度 | 最低准确度 | 准确度方差 |
| 90.60% | 92.74% | 87.07% | 1.40343\*10-4 |

分析表格6可知：

<1>单纯使用动态降噪时的平均准确度相比无优化时提升约4.1%，准确度方差降低约9.0%。

<2>在该统计结果中，开启动态降噪条件下，使用集成后，准确度方差至少降低48.41%，集成对提升分类性能稳定性的效力显著。

<3>随集成分类器弱度上升，分类性能下降。当分类器弱度为0.1时，平均准确度能得到略微提升；当分类器弱度为0.3、0.5时，平均准确度略微下降；说明集成对提升准确度作用不大。

<4>在该统计结果中，当分类器弱度为0.1时，随集成分类器数量增加，平均准确度逐步提升，准确度方差逐步下降，分类性能提升。

## 4.5 优化性能测试

该部分测试将综合使用预处理优化和分类器优化，寻找最佳参数，观察并分析优化结果。

### 4.5.1 主数据集测试结果

使用4.2.1生成的50份随机划分的训练集/测试集。使用分类器GaussianNaiveBayesClassfier进行分类，选取最佳参数如下：

<1>主特征选择方式：基于梯度提升决策树特征增益。

<2>主特征选择数量：2。

<3>开启动态降噪has\_denoise=True。

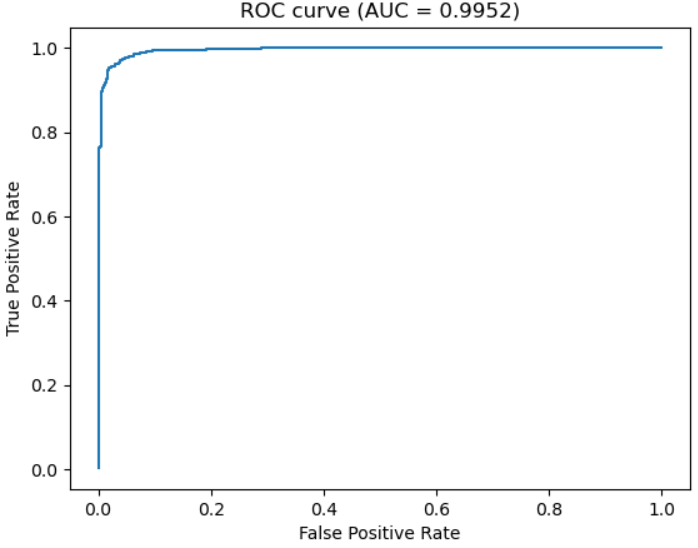
<4>设置集成附加分类器数量bagging\_rate=13。

<5>设置集成分类器弱度bagging\_weakness=0.1。

得到如下分类结果：

|  |
| --- |
| ------------测试总次数=50------------  动态降噪： True  集成： 13  累积混淆矩阵：  男声 女声  预测男声 22971 865  预测女声 755 22959  总男声比例：49.90%  最大准确度：97.58%  最大准确度使用特征： ['meanfun', 'IQR', 'label']  最低准确度：95.16%  最低准确度使用特征： ['meanfun', 'IQR', 'label']  平均准确度：96.59%  准确度方差：2.77311e-05  平均男声查准率：96.37%  平均男声查全率：96.82%  平均女声查准率：96.82%  平均女声查全率：96.37%  平均男声似然比：26.6658  平均女声似然比：30.2842  判别男声相关：  平均诊断比值比：807.5507  平均F1分数：0.9659 |

典型的受试者工作特征（ROC）曲线如图8所示，该曲线取自voice\_test\_1.csv的测试结果。



1. voice\_test\_1.csv测试集ROC曲线

由测试结果可知：

<1>综合使用两方面优化后，准确度相对于无优化时提升显著，相对于单独使用某一方面优化时略有提升，准确度方差维持在低水平。

<2>在本次测试选用的参数条件下，因为提取的主特征数为2，因此动态降噪发挥的作用很小。集成对降低方差的效力依然十分显著。

<3>在本次测试选用的参数条件下，对男/女声的辨别效果相当，没有明显的不平衡倾向。

<4> ROC曲线显示分类器辨别力很强。

### 4.5.2 额外用例测试结果

使用数据集voice.csv进行训练，对附加的10份测试用例进行分类预测。选取最佳参数如下：

<1>主特征选择方式：基于梯度提升决策树特征增益。

<2>主特征选择数量：4。

<3>开启动态降噪has\_denoise=True。

<4>设置集成附加分类器数量bagging\_rate=13。

<5>设置集成分类器弱度bagging\_weakness=0.1。

测试结果细节如表格7所示。

1. 额外用例测试结果细节

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 声音性别 | 说明 | 预测相对概率 | | 预测结果 |
| 女声 | 男声 |
| 1 | male | 男声中速朗读 | 0.0612 | 0.939 | male |
| 2 | male | 男声快速朗读 | 0.461 | 0.539 | male |
| 3 | male | 男声日常交际 | 0.305 | 0.695 | male |
| 4 | male | 男中音歌唱 | ≈1 | 0.0000209 | female |
| 5 | male | 假声男高音歌唱 | ≈1 | ≈0 | female |
| 6 | female | 女声中速朗读 | ≈1 | 0.000232 | female |
| 7 | female | 女声快速朗读 | 0.999 | 0.000552 | female |
| 8 | female | 女声日常交际 | 0.997 | 0.00266 | female |
| 9 | female | 女中音歌唱 | 0.999 | 0.000537 | female |
| 10 | female | 女声低音吟唱 | ≈1 | ≈0 | female |

对额外用例的分类效果也有较大提升，准确度为80%。其中对说话类型的声音片段判别较为准确，对歌唱类型的声音片段全部判别为女声。

相比无优化条件下的测试结果，主特征选择和分类器优化确实提高了准确度，但由于缺乏音色相关的特征，分类器对歌唱类型的声音片段几乎没有判别能力。

## 4.6 结果分析

以查全率作为男/女声判别的正确率指标，将主数据集最优测试结果表述如下：

|  |  |
| --- | --- |
| 男声正确率：96.82% | 男声错误率：3.18% |
| 女声正确率：96.37% | 女声错误率：3.63% |

通过优化设计，主数据集的测试结果得到了显著提升，但对额外选取的特殊用例进行分类的效果仍然较差。此处将基于前述测试结果对优化设计效果进行评述，并分析声音特征提取、预处理过程、分类器存在的缺陷。

### 4.6.1 优化设计效果评述

根据预处理优化性能测试结果，认为主特征选择的预处理优化对提升分类性能的准确度、查准率、查全率等指标效果良好，并且能够在一定程度上平衡男/女声分类效果，稳定分类器性能；这项优化措施的优化原理在于放大男/女声统计特征的主要差异，忽略次要差异和无关项以减少样本噪音干扰。

根据分类器优化性能测试结果，经过动态降噪处理，在不进行主特征选择的情况下，分类器性能仍能稳定提升，但对男/女声的分类效果不平衡，准确度方差也较大。

集成分类器可以降低泛化误差中的方差项，当方差项是泛化误差的主要来源时，使用集成可以显著降低泛化误差，稳定分类器性能，主要表现为准确度方差明显下降；当方差项不是泛化误差的主要来源，而噪音是主要来源时，集成效果不明显。

在不进行主特征选择的条件下，可以通过综合使用动态降噪和集成以获得较优的分类性能。

### 4.6.2 分类流程中存在的缺陷

（1）声音特征提取缺陷

1.根据主数据集的测试结果，分类器性能最好时仅需使用声音基频和声音频率四分位差这两项特征。然而，在进行声音特征提取时总计提取了20项特征，这些特征中的大多数对分类的帮助很小，甚至起反作用。

2.本次实验中提取的特征均为反映整体情况的频率特征，缺少局部特征。有相关研究指出，虽然男/女声主要差别体现在基频和音域上，但低音区和高音区的局部特征差异也是不可忽略的；例如，男声的高频泛音丰富度明显低于女声和童声、男声糙度高于女声且分布不同等。

3.构造数据集时，仅对音频文件前20秒的录制内容进行特征提取，局部录音片段具有较大的偶然性，可能由此导致误判；例如，恰好截取了男声假声唱段导致判别结果为女声。

4.本次实验仅提取了音频特征，没有提取任何音色特征，但男/女声在音色上也有一定区别。这一特征的缺失直接导致无法有效地对基频处于交叠区的声音进行分类。

（2）预处理过程缺陷

预处理采用的主特征选择方法是过滤法，次要特征直接删除，因此可能会丢失较多信息。更好的方法是嵌入法，通过调整特征权重在强调主要特征的同时也保留一定的次要特征。

（3）分类器缺陷

本次实验使用的高斯朴素贝叶斯分类器是基于中心极限定理和朴素贝叶斯假设的。中心极限定理成立的条件是样本数量充分多，但本次实验所使用的数据集只有3168条数据，样本数量并不足够多。另外，提取的部分音频特征具有一定的关联性，不完全符合朴素贝叶斯假设；例如，基频最小值和基频平均值有一定相关性。

5 总结与展望

## 5.1 总结

本次实验的主要任务如下：

（1）了解声音检测识别技术的起源、发展历程和意义。

（2）了解声学频率指标及其含义。

（3）分析数据集构成并构建数据集。

（4）掌握高斯朴素贝叶斯分类算法原理。

（5）对预处理过程和分类器进行优化设计。

（6）测试并进行优化评述。

本次实验从了解声音检测识别技术的起源和研究现状开篇，经历了数据集构建、数据预处理、分类器设计、测试和优化的实验流程，较为完整地展现了如何使用机器学习算法完成发声源判别的任务，并最终取得了比较好的结果。

在本次实验中主要使用到2个数据集，其一是采样自The Harvard-Haskins Database等构建的主数据集，其二是采样实际录音片段构建的额外数据集。

主数据集主要采样自演讲片段，音频主要是说话人声，测试的主要目的是评估分类器性能。在优化设计后，分类器的准确度和稳定性都得到显著提升。但取到最优结果时，所使用的主特征仅有基频和四分位差，这在一定程度上反映了特征提取流程存在的缺陷。

额外数据集包含说话人声和歌唱人声，测试的主要目的是寻找分类流程缺陷。由于缺乏对音频局部频率特征和音色特征的提取，分类器总是将歌唱人声判别为女声。并且在未进行优化时，分类结果非常差。由此可见，实际录制的音频可能具有更多的噪音，如果希望将分类器应用到实际，那么特征选择和降噪处理将显得尤为重要。

## 5.2 展望

本次实验实现了使用机器学习的方法判别发声源的基本框架，但仍存在不少缺陷，尤其是在特征选择方面还有很大改进空间。实验中实现特征选择的方法是最简单的过滤法，而如果使用包裹法或嵌入法或将取得更好效果；但为了配合特征选择方法，高斯朴素贝叶斯分类器也需要进行改进，例如为每一特征重新分配权重等。

总之，本次实验只是非语音信号检测识别技术很小的一部分。虽然采样和处理的信号是语言信号，但并不进行语言文字识别，而是判别发声源，因而属于非语音信号检测识别技术。该项识别技术领域的发展虽然落后于以语言文字识别、自然语言处理为代表的语音信号检测识别技术，但其在工业探伤领域、医学健康领域仍有较大的发展潜力。

参考文献

[1] 佚名.The Harvard-Haskins Database of Regularly-Timed Speech[EB/OL].http://nsi.wegall.net.

[2] 佚名.VoxForge Speech Corpus[EB/OL].http://www.repository.voxforge1.org/downloads/SpeechCorpus/Trunk/Audio/Main/8kHz\_16bit.

[3] 佚名.Festvox CMU\_ARCTIC Speech Database at Carnegie Mellon University[EB/OL].http://festvox.org/cmu\_arctic.

[4] Kory Becker.Identifying the Gender of a Voice using Machine Learning[EB/OL].http://www.primaryobjects.com/2016/06/22/identifying-the-gender-of-a-voice-using-machine-learning,2016-6-22.

[5] Kory Becker.Gender Recognition by Voice[EB/OL].https://www.kaggle.com/primaryobjects/voicegender,2016-8-26.

[6] 尹巧萍,赵力.利用语音基音统计特征的说话人性别判识的研究[J].语音、通信及信号处理,2007,26(4):206-208.

[7] 张思.声音性别知觉的影响因素[J].课程教育研究,2018(12).

[8] Jerome Sueur.Spectral Properties[EB/OL].https://www.rdocumentation.org/packages/seewave/versions/2.1.0/topics/specprop.

[9] Otsu N.A Threshold Selection Method from Gray-Level Histograms[J].IEEE Transactions on Systems Man & Cybernetics,2007,9(1):62-66.

[10] 佚名.解读男女声音的区别：亮度，糙度[EB/OL].https://bideyuanli.com/p/863.

附录 voice.csv数据集参考说明

**Voice Gender**

Gender Recognition by Voice and Speech Analysis

This database was created to identify a voice as male or female, based upon acoustic properties of the voice and speech. The dataset consists of 3,168 recorded voice samples, collected from male and female speakers. The voice samples are pre-processed by acoustic analysis in R using the seewave and tuneR packages, with an analyzed frequency range of 0hz-280hz (human vocal range).

**The Dataset**

The following acoustic properties of each voice are measured and included within the CSV:

* meanfreq: mean frequency (in kHz)
* sd: standard deviation of frequency
* median: median frequency (in kHz)
* Q25: first quantile (in kHz)
* Q75: third quantile (in kHz)
* IQR: interquantile range (in kHz)
* skew: skewness (see note in specprop description)
* kurt: kurtosis (see note in specprop description)
* sp.ent: spectral entropy
* sfm: spectral flatness
* mode: mode frequency
* centroid: frequency centroid (see specprop)
* peakf: peak frequency (frequency with highest energy)
* meanfun: average of fundamental frequency measured across acoustic signal
* minfun: minimum fundamental frequency measured across acoustic signal
* maxfun: maximum fundamental frequency measured across acoustic signal
* meandom: average of dominant frequency measured across acoustic signal
* mindom: minimum of dominant frequency measured across acoustic signal
* maxdom: maximum of dominant frequency measured across acoustic signal
* dfrange: range of dominant frequency measured across acoustic signal
* modindx: modulation index. Calculated as the accumulated absolute difference between adjacent measurements of fundamental frequencies divided by the frequency range
* label: male or femal

1. 参见文献[7]《声音性别知觉的影响因素》 [↑](#footnote-ref-1)
2. 参见文献[6]《利用语音基音统计特征的说话人性别判识的研究》 [↑](#footnote-ref-2)