性别语音识别项目报告

I. 问题的定义

项目概述

此项目属于语音识别领域的一个应用，来源于 Kaggle 的 *Gender Recognition by Voice*[1] 项目。此项目的任务是通过语音识别说话者的性别，使用的主要数据集是 voice.csv，可以从 Kaggle 上获取。该数据集是从男性和女性说话者收集而来的 3,168 个语音样本在 R 中使用 seewave 和 tuneR 软件包通过声学分析进行预处理而得。另外，从实际生活中收集了 6 个 10s 的语音样本，经过同样的预处理保存在 real_voice.csv 中。

语音是人机交互的重要方式之一，在许多方面，它比传统的通过人机界面交互的方式更快速便捷，在未来将有更广泛的应用。支持这种语音交互方式的核心技术就是语音识别技术，我想通过此项目能对语音识别有更多的了解。

问题陈述

该项目要解决的问题是通过说话者的一段语音判断该说话者的性别。采用的数据集是已经用 R 语言从语音样本中提取特征后的数据集，且包含有目标分类的 label（即性别），而性别只有男性和女性，所以这是一个可以用相应的机器学习算法来解决的二分类的问题。

二分类的机器学习算法有不少，如逻辑回归、朴素贝叶斯、支持向量机、决策树等。在此项目中，我打算使用 XGBoost 模型算法，这是一种集成算法，相对普通的单算法有更好的鲁棒性和准确性。

要解决此项目问题，首先得探索数据，了解数据的基本情况，然后据此做必要的数据预处理。接下来，训练基准模型和解决问题的模型，并对他们进行评估比较，了解模型的好坏情况。最后，利用网格搜索法对模型进行超参调优以找出最好的模型，并对其评估。预期最终模型在测试集上有 99%左右的准确率。

评价指标

在此项目中，并不偏重某个分类，而是看整体的预测准确情况，因此，我选用准确率做为模型的评估指标。准确率的定义[2]如下：

$$accuracy(y,\hat{y})=\frac{1}{n_{samples}}\sum_{0}^{n_{samples}-1}I(\hat{y}_i=y_i)$$

其中， $\hat{y}_i$ 为索引为 i 的样本的预测值， y_i 为其相应的真实值， $n_{samples}$ 为预测的样本总数， $I(x)$ 为指标函数（即 x 为 True 时其值为 1，否则为 0）。

II. 分析

数据的探索及可视化

数据集 voice.csv 为主要数据集，只对它探索一下。首先，看看几个数据样本:

	0	1	2	3	4
meanfreq	0.059781	0.0660087	0.0773155	0.151228	0.13512
sd	0.0642413	0.06731	0.0838294	0.0721106	0.0791461
median	0.0320269	0.0402287	0.0367185	0.158011	0.124656
Q25	0.0150715	0.0194139	0.00870106	0.0965817	0.0787202
Q75	0.0901934	0.0926662	0.131908	0.207955	0.206045
IQR	0.075122	0.0732523	0.123207	0.111374	0.127325
skew	12.8635	22.4233	30.7572	1.23283	1.10117
kurt	274.403	634.614	1024.93	4.1773	4.33371
sp.ent	0.893369	0.892193	0.846389	0.963322	0.971955
sfm	0.491918	0.513724	0.478905	0.727232	0.783568
mode	0	0	0	0.0838782	0.104261
centroid	0.059781	0.0660087	0.0773155	0.151228	0.13512
meanfun	0.0842791	0.107937	0.0987063	0.0889648	0.106398
minfun	0.0157017	0.0158259	0.0156556	0.0177976	0.0169312
maxfun	0.275862	0.25	0.271186	0.25	0.266667
meandom	0.0078125	0.00901442	0.00799006	0.201497	0.712812
mindom	0.0078125	0.0078125	0.0078125	0.0078125	0.0078125
maxdom	0.0078125	0.0546875	0.015625	0.5625	5.48438
dfrange	0	0.046875	0.0078125	0.554688	5.47656
modindx	0	0.0526316	0.0465116	0.247119	0.208274
label	male	male	male	male	male

从这几个样本中，可以知道有哪些特征，也可以大概了解点数据情况。不妨作下可能性预估：除 label 特征外，其它特征的值皆为数值型且都为非负数。

再来看看数据集的基本信息：

```
#查看数据信息
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3168 entries, 0 to 3167
Data columns (total 21 columns):
meanfreq    3168 non-null float64
sd          3168 non-null float64
median      3168 non-null float64
Q25         3168 non-null float64
Q75         3168 non-null float64
IQR         3168 non-null float64
skew        3168 non-null float64
kurt        3168 non-null float64
sp.ent      3168 non-null float64
sfm         3168 non-null float64
mode        3168 non-null float64
centroid    3168 non-null float64
meanfun     3168 non-null float64
minfun      3168 non-null float64
maxfun      3168 non-null float64
meandom     3168 non-null float64
mindom      3168 non-null float64
maxdom      3168 non-null float64
dfrange     3168 non-null float64
modindx     3168 non-null float64
label       3168 non-null object
dtypes: float64(20), object(1)
memory usage: 519.8+ KB
```

从这个基本信息中，可以知道总共有 3168 个样本，21 个特征（其中 label 特征为目标特征），除 label 特征外其它特征的值都为浮点型（验证了前面的数值型预估），所有特征都没有缺失值。

再看看目标特征值的情况：

```
#查看label的值的情况
data['label'].value_counts()
```

```
male      1584
female    1584
Name: label, dtype: int64
```

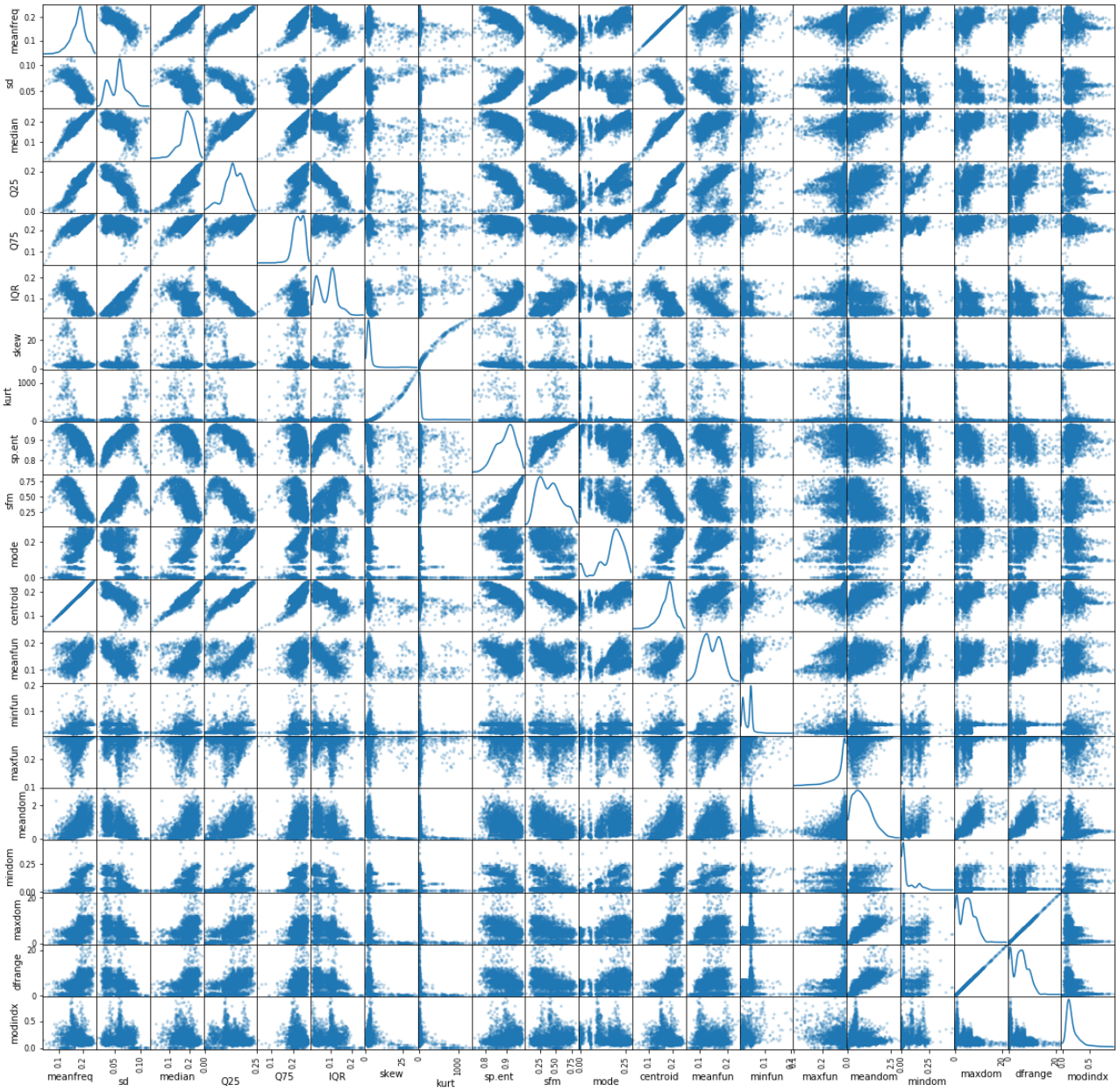
目标特征只有两个值：male 和 female，且二者占有的样本数量一样。

接下来看看数值型特征的统计量：

	count	mean	std	min	25%	50%	75%	max
meanfreq	3168.0	0.180907	0.029918	0.039363	0.163662	0.184838	0.199146	0.251124
sd	3168.0	0.057126	0.016652	0.018363	0.041954	0.059155	0.067020	0.115273
median	3168.0	0.185621	0.036360	0.010975	0.169593	0.190032	0.210618	0.261224
Q25	3168.0	0.140456	0.048680	0.000229	0.111087	0.140286	0.175939	0.247347
Q75	3168.0	0.224765	0.023639	0.042946	0.208747	0.225684	0.243660	0.273469
IQR	3168.0	0.084309	0.042783	0.014558	0.042560	0.094280	0.114175	0.252225
skew	3168.0	3.140168	4.240529	0.141735	1.649569	2.197101	2.931694	34.725453
kurt	3168.0	36.568461	134.928661	2.068455	5.669547	8.318463	13.648905	1309.612887
sp.ent	3168.0	0.895127	0.044980	0.738651	0.861811	0.901767	0.928713	0.981997
sfm	3168.0	0.408216	0.177521	0.036876	0.258041	0.396335	0.533676	0.842936
mode	3168.0	0.165282	0.077203	0.000000	0.118016	0.186599	0.221104	0.280000
centroid	3168.0	0.180907	0.029918	0.039363	0.163662	0.184838	0.199146	0.251124
meanfun	3168.0	0.142807	0.032304	0.055565	0.116998	0.140519	0.169581	0.237636
minfun	3168.0	0.036802	0.019220	0.009775	0.018223	0.046110	0.047904	0.204082
maxfun	3168.0	0.258842	0.030077	0.103093	0.253968	0.271186	0.277457	0.279114
meandom	3168.0	0.829211	0.525205	0.007812	0.419828	0.765795	1.177166	2.957682
mindom	3168.0	0.052647	0.063299	0.004883	0.007812	0.023438	0.070312	0.458984
maxdom	3168.0	5.047277	3.521157	0.007812	2.070312	4.992188	7.007812	21.867188
dfrange	3168.0	4.994630	3.520039	0.000000	2.044922	4.945312	6.992188	21.843750
modindx	3168.0	0.173752	0.119454	0.000000	0.099766	0.139357	0.209183	0.932374

从中可知，特征值的最小值都为非负数（验证了非负数预估），大部分特征最大值都没超过 1。

最后再看看各特征的分布情况及它们之间的相关性（对角线上为特征的分布图，其它的为两特征之间的散点图）：



有几个特征近似服从正态分布，但大部分的离正态分布相差较大，有的甚至严重偏斜。meanfreq 特征与 centroid 特征之间，maxdom 特征与 dfrange 特征之间有很强的相关性，skew 特征与 kurt 特征之间也有较强的线性关系。其它各特征之间的相关性都不怎么强。

算法和技术

此项目主要使用了如下算法或技术：

XGBoost 算法[3]

XGBoost，Extreme Gradient Boosting 的简称，是 gradient boosted trees 的一种实现。 Gradient Boosting 是弗里德曼在其书 *Greedy Function*

Approximation: A Gradient Boosting Machine [4] 中提出，XGBoost 就是建立在这个原始模型之上的。Gradient Boosting 是一种有监督学习的集成学习算法，其基学习器为 CART 决策树，其训练过程就是最优化目标函数（一个用来衡量给定一组参数的模型的性能的函数）。

CART 的每个叶结点都有一个预测分（它决定着预测结果），模型的最终预测分 $\hat{y}_i$ 为所有 CART 的累加，即

$$\hat{y}_i=\sum_{k=1}^{K} f_k(x_i)$$

其中，K 为 CART 的数量， f_k 为在对应 CART 上数据点到预测分的映射。于是，目标函数可以写成

$$obj(\theta)=\sum_{i=1}^{n} l(y_i,\hat{y}_i)+\sum_{k=1}^{K}\Omega(f_k)$$

其中， l 为损失函数（用来衡量模型对训练数据的预测性）， Ω 为正则化项（用来控制模型的复杂度以帮助避免过拟合）。由于很难同时训练所有的树，所以采用累加策略：固定模型已经学到的，每次添加一个新的树。把第 t 步的预测分记为 $\hat{y}_i^{(t)}$ ，则

$$\begin{aligned}
&\hat{y}_i^{(0)}=0\\
&\hat{y}_i^{(1)}=f_1(x_i)=\hat{y}_i^{(0)}+f_1(x_i)\\
&\hat{y}_i^{(2)}=f_1(x_i)+f_2(x_i)=\hat{y}_i^{(1)}+f_2(x_i)\\
&\quad\cdots\\
&\hat{y}_i^{(t)}=\sum_{k=1}^{t} f_k(x_i)=\hat{y}_i^{(t-1)}+f_t(x_i)
\end{aligned}$$

于是，每一步要做的就是添加一个优化目标函数

$$obj^{(t)}=\sum_{i=1}^{n} l(y_i,\hat{y}_i^{(t)})+\sum_{k=1}^{K}\Omega(f_k)$$

的树。

XGBoost 在此基础上又充分利用计算机的可用资源以加快计算速度、提高模型性能。因此，XGBoost 继承了决策树的容易理解和解释、需要的数据前处理少、可以处理多输出问题等优点，同时也缓解了决策树容易过拟合、不稳定的缺点，另外还具有运行速度快、模型性能好的优点。

在此项目中，XGBoost 模型算法用来训练及测试数据。涉及的主要参数包括：

X：特征矩阵

y：标签即目标变量

max_depth： 基学习器的最大深度

n_estimators：模型训练用的基学习器的数量

learning_rate：boosting 学习率

交叉验证

交叉验证或 K 折交叉验证，是把训练集分成 K 个子集，然后分别把每个子集作为验证集，其他 K-1 个子集作为训练集，如此训练、测试数据 K 次。最后把 K 次得出的验证的评分的平均值作为 K 折交叉验证的评分。涉及的主要参数包括：

estimator：用来训练的模型
X：特征矩阵
y：标签即目标变量
cv： 分割成的子集个数即 K

网格搜索法

网格搜索法就是穷尽模型超参空间组合以找出评分最高的超参组合从而找出最优模型。此项目中，网格搜索法用来调节 XGBoost 模型的超参，其使用的评分就是通过交叉验证得来的评分。涉及的主要参数包括：

estimator：用来训练的模型
param_grid：模型的超参空间
scoring：评分指标
cv：使用的交叉验证或交叉验证中分割成的子集个数即 K
X： 特征矩阵
y： 标签即目标变量

基准模型

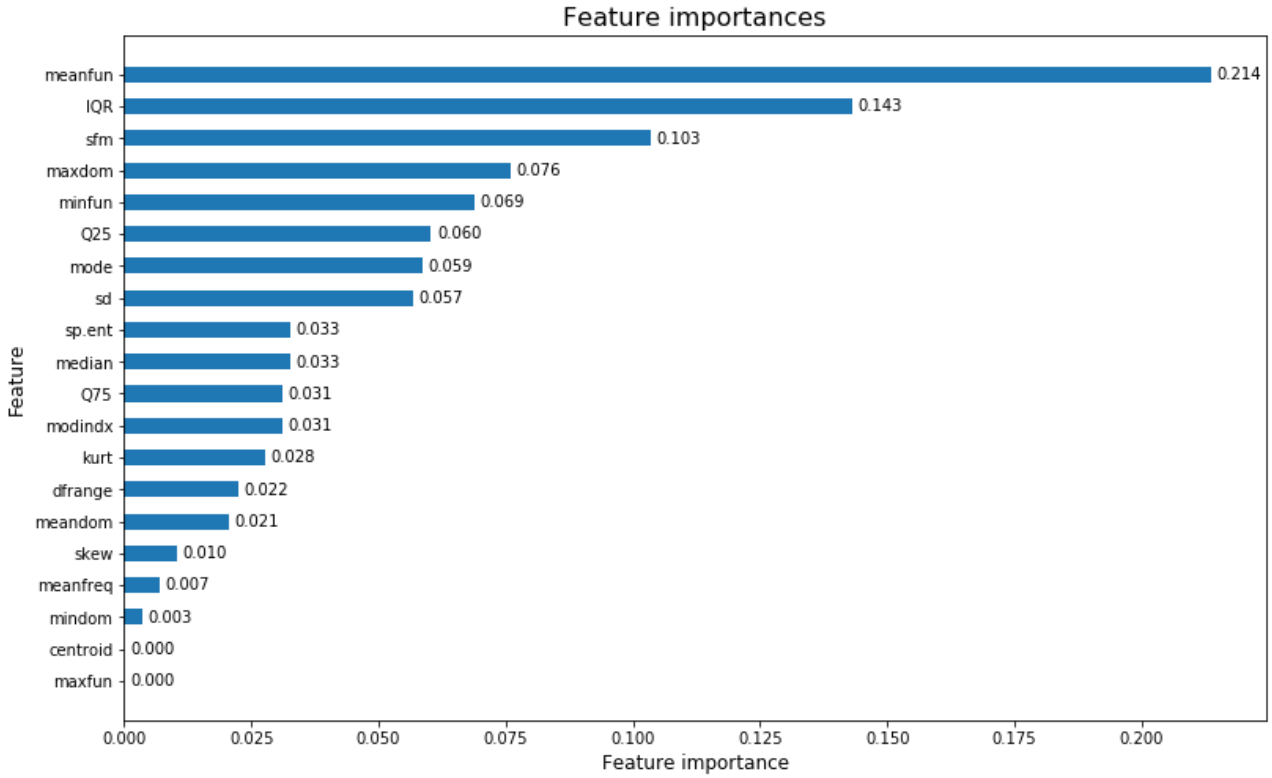
此项目使用的基准模型为 CART 算法的决策树，这是一个相对简单的、可解释性强的有监督学习算法，且也比较适合本项目，可以作为本项目的一个基准模型。用它训练数据并使用与解决方案模型相同的评估指标来评估其结果，以此结果作为基准。预期解决方案模型的性能比该基准要好些。

III. 方法

数据预处理

此项目是个二分类问题，目标特征的值是字符串类型，可以把它转化 0 和 1，以方便处理。由于采用的模型的基学习器是决策树，而决策树对大数值范围、异常值有很好的鲁棒性，因此对数据集无需做标准化处理或异常值处理。从前面的数据探索可知，数据集并没有缺失值，因此也无需做缺失值处理。有两对特征有很强的相关性，即存在冗余特征，可以删掉冗余特征以减少模型的复杂度，另外，有些特征对算法没什么用甚至造成干扰，也应该去掉。此项目采用

的是基于决策树的模型，可以利用模型中的 feature importances 属性来进行特征刷选，feature importances 展示如下：



对于两对强相关性特征 meanfreq 与 centroid，maxdom 与 dfrange，centroid 和 dfrange 的 feature importance 更小，因此我把它们去掉。maxfun 的 feature importance 为 0，是无关特征，也要去掉。

执行过程

此项目是基于 python 语言来实现的，算法或技术主要使用的是 scikit-learn[5]包和 xgboost[6]包。预处理后的数据集混洗后分割成训练集和测试集（测试集占比 0.2，混洗的 random state 为 0），分别用于模型训练和模型测试或评估。模型训练的执行过程分三步：导入模型分类器的包、设置模型分类器的超参、训练模型（调用 fit 方法）。对于基准模型，使用的是 DecisionTreeClassifier 分类器，其超参为默认参数（主要参数 max_depth=None，即不限制树的深度）。对于解决方案模型，使用的是 XGBClassifier 分类器，其初始模型使用的超参也为默认参数（主要参数：max_depth=3，learning_rate=0.1，n_estimators=100）。模型评估的执行过程是让在训练集上训练好的模型在测试集上进行预测（调用 predict 方法）并对比预测结果和真实结果算出预测的准确率（调用 accuracy_score 方法进行计算）。测得基准模型的准确率为 95.74%，初始解决方案模型的准确率为 98.58%。

完善

此项目此项主要是通过网格搜索法来调节模型超参以完善模型的。模型涉及的主要超参有 max_depth、learning_rate 和 n_estimators，都是参数值越大，模型就越复杂，反之亦然。调节模型复杂度其实就是平衡方差与偏差以达到最佳模型。max_depth 决定了模型的基学习器本身的复杂度，是需要调节的，而 learning_rate 和 n_estimators 都是控制基学习器集成的复杂度，可固定其中一个通过另一个调节集成的复杂度。此项目采用固定 n_estimators=1000，调节 max_depth 的范围为[2, 3, 4, 5]，调节 learning_rate 的范围为[0.02, 0.04, 0.06, 0.08, 0.1]，使用 5 折交叉验证来评分。网格搜索法得到的各参数组合的模型的交叉验证分及排名如下：

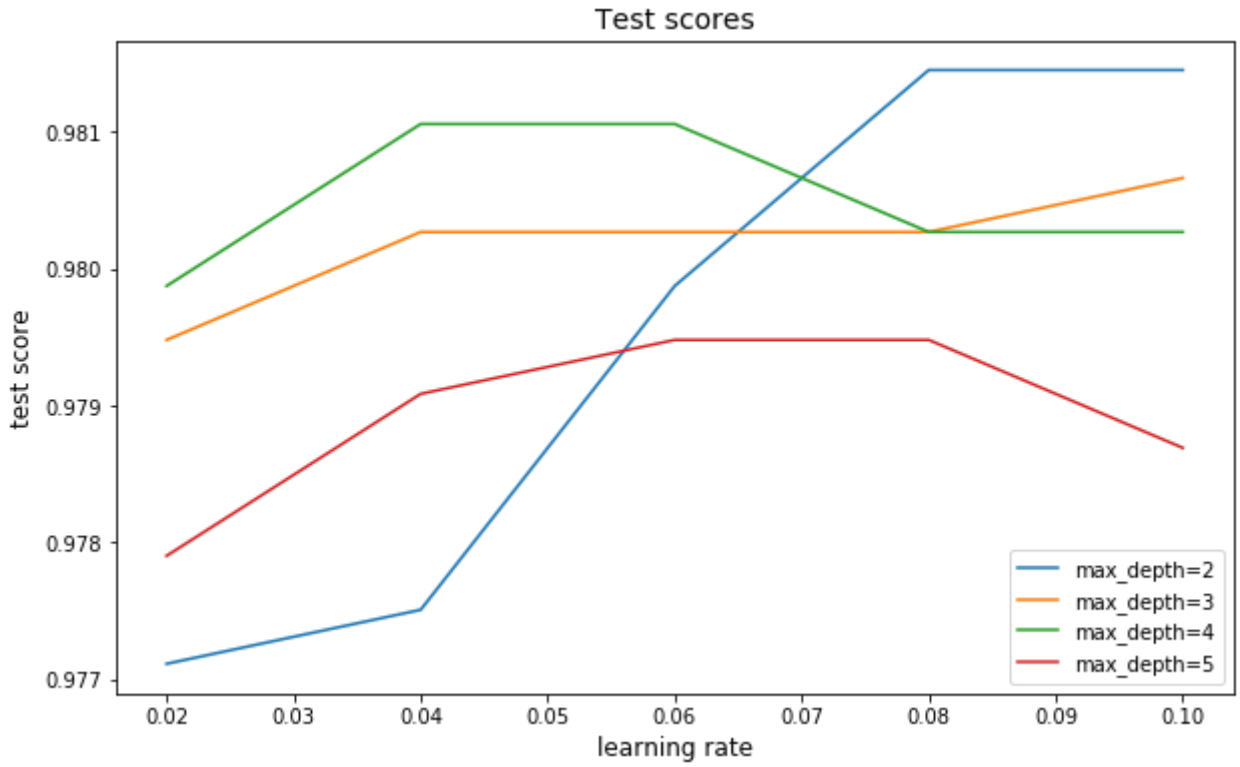
	param_max_depth	param_learning_rate	mean_test_score	rank_test_score
0	2	0.02	0.977111	20
1	3	0.02	0.979479	13
2	4	0.02	0.979874	11
3	5	0.02	0.977901	18
4	2	0.04	0.977506	19
5	3	0.04	0.980268	6
6	4	0.04	0.981058	3
7	5	0.04	0.979084	16
8	2	0.06	0.979874	11
9	3	0.06	0.980268	6
10	4	0.06	0.981058	3
11	5	0.06	0.979479	13
12	2	0.08	0.981452	1
13	3	0.08	0.980268	6
14	4	0.08	0.980268	6
15	5	0.08	0.979479	13
16	2	0.1	0.981452	1
17	3	0.1	0.980663	5
18	4	0.1	0.980268	6
19	5	0.1	0.978690	17

网格搜索法最后返回最佳模型，其主要超参为 max_depth=2、learning_rate=0.08、n_estimators=1000。

IV. 结果

模型的评价与验证

最终模型是通过网格搜索法穷尽一定的超参空间而挑选出的最佳模型。网格搜索法除了会返回最佳模型外还可以获得各模型的交叉验证评分，下面就对其做一下分析。由网格搜索法获得的各模型的交叉验证评分结果根据不同的 max_depth 画出学习率-评分曲线图如下：



由偏差-方差分析可知，随着模型从简单到复杂，模型性能表现会由差逐渐变好然后再变差，中间有个平衡点是其性能表现最好的时候。在此，模型性能表现就是用测试评分来衡量的。由此，再来分析上图四条曲线，max_depth=4 和 max_depth=5 时的曲线都符合这种趋势，并分别在 learning_rate=0.04 和 learning_rate=0.06 时取得最高评分；max_depth=3 的曲线经过一段平稳评分后有微小上升，且整体曲线也比较平缓，若增大 learning_rate，有较大上升的可能性不大，可以认为其在 learning_rate=0.1 时取得最高评分；max_depth=2 的曲线在经过较大上升后进入平稳， 若增大 learning_rate，有较大上升的可能性也不大，可认为平稳时的评分就是最高评分。再看 max_depth，在其增加到 5 时就开始变得比较糟糕了，再增加很可能会更糟糕。综上分析，在所有 max_depth 和 learning_rate 中，最高评分就在以上四条曲线上，而这四条曲线上又在 max_depth=2、learning_rate=0.08 或 0.1 时取得最高评分，考虑奥卡姆剃刀原理应取 learning_rate=0.08，这与网格搜索返回的最佳模型的参数是一致的。

决策树模型是不稳定的，小的数据改变可能会导致生成完全不同的树，但此项目使用的是一种决策树集成的模型，此问题得到了不少缓解。因此，训练数据的微小改变可能会对结果有一定影响，但不会太大。

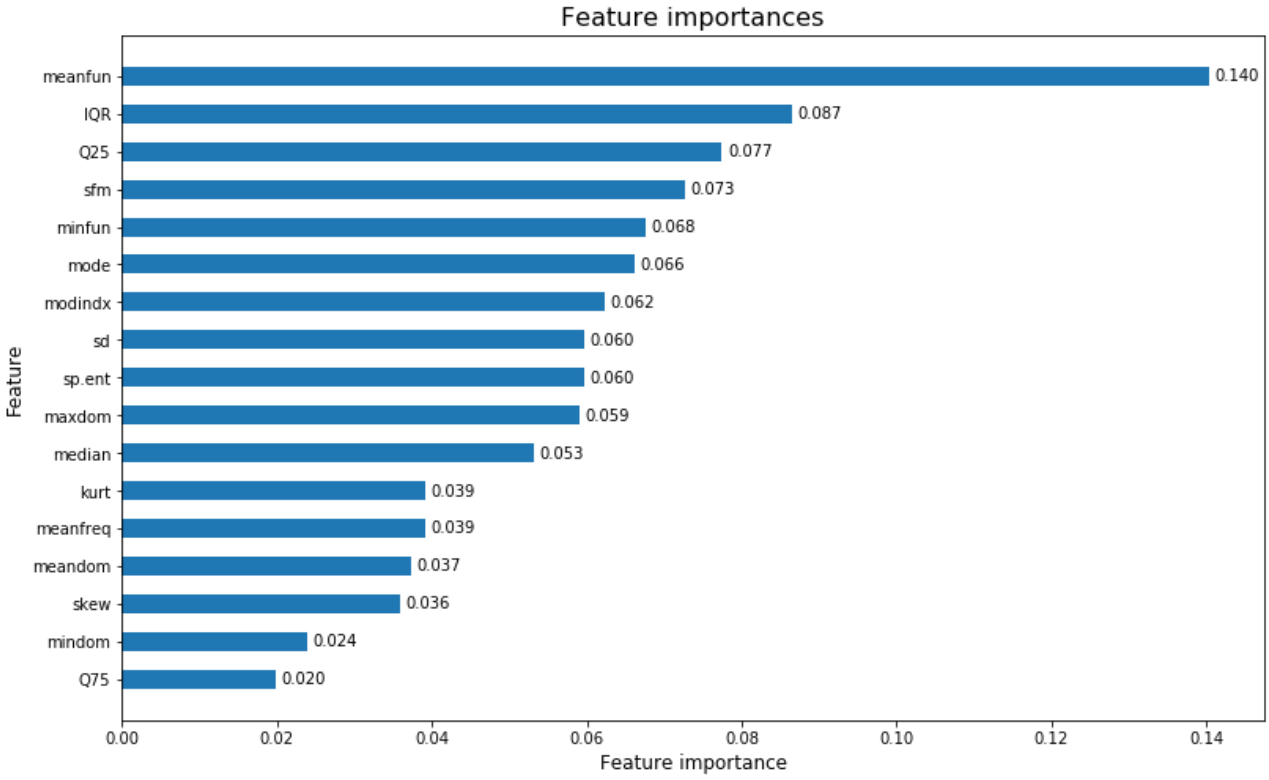
合理性分析

最终模型在测试集上的准确率为 98.74%，比基准模型的 95.74%要好些，还算可以。对于和数据集相似的语音样本，最终模型能较好的预测出其说话者的性别，基本能解决此项目的问题。但用于训练的数据集只是来自真实世界人的声音的很小一部分，想要普遍应用于预测世界各地不同人的性别貌似还是有不少难度的。

V. 项目结论

自由形态的可视化

决策树的训练过程是不断的通过选取一个特征来把训练集划分成不同的子集直到能比较好的判断子集样本的分类从而生成一个树。在这个树中，有些特征占据着重要的作用，而有些特征的作用不大甚至可能没有作用。在基于决策树的模型中，就用 feature importance 来衡量这种特征在树中占据的重要性。此项目的最终模型的 feature importances 如下所示：



从上图可知，meanfun 特征在此模型占着最大的重要性（为 14%），其它特征也各占着 2%到 8.7%的重要性。

对项目的思考

此项目基本按照 数据探索—数据准备—模型训练—模型评估—模型调优—模型测试 的流程来的。整个过程相对简单，没有做什么复杂或特殊的处理。感觉特征选择这步有些困扰，其一，冗余特征是否要去掉，一方面冗余特征并不怎么降低模型性能表现甚至可能会有一丁点作用，另一方面它会增加模型的复杂性从而增加计算开销等不利影响；其二，无关或不重要的特征的刷选是依赖模型的 feature_importances_属性的，但此时最终模型还未确定，而不同模型的这个属性可能是不同的（虽然也可能是差不多的），以哪个模型的这个属性进行刷选呢；其三，特征的 feature importance 低于多少才应该被去掉呢。

最终模型在测试集上测试的准确率为 98.74%，还算可以，虽然离我期望的 99%还有些差距。但是，拿此模型在我从生活中收集来的 6 段语音上预测，表现很糟糕（全部预测为男性，实际有一半女性），用于训练的数据集还是太小，远远不能在通用场景下解决这类问题。

需要做出的改进

此项目此项并没有做什么数据前处理，也许做一定的数据前处理（如删除异常值等）后，最终模型的效果能好点。另外，模型的性能表现评估只采用了准确率这个指标，显得有点单一，可以考虑增加 AUC 指标来综合评估。还有，模型调优只考虑了三个主要超参，还可以考虑增加 gamma, min_child_weight, subsample 等超参进行调优，还可以考虑加入早停技术。

引用

[1] https://www.kaggle.com/primaryobjects/voicegender

[2] 参考 scikit-learn 关于 accuracy score 的定义：http://scikit-learn.org/stable/modules/model_evaluation.html#model-evaluation

[3] 参考：https://xgboost.readthedocs.io/en/latest/model.html

[4] Friedman, Jerome H. "Greedy Function Approximation: A Gradient Boosting Machine." The Annals of Statistics 29, no. 5 (2001): 1189-232. http://www.jstor.org/stable/2699986.

[5] http://scikit-learn.org/stable

[6] https://xgboost.readthedocs.io/en/latest