GENDER VOICE RECOGNITION

- BY VOICE AND SPEECH ANALYSIS

SHALINI GARIKAPATI

Fall 2018

TABLE OF CONTENTS:

Introduction

Aim:

To determine the gender of a person identifying them as either female or male.

Objective:

The objective is to determine the gender using the acoustic properties such as mean frequency, mode ,skew etc.We do this by take different machine learning algorithms and compare each other to get the best result.Later we do the prediction.

The initial objectives are:

- To find a dataset.
- Understand the structure of the data and fit them into models that can be uns.

Overview:

The database was taken from Kaggle. 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).

But I used this database and worked it in Jupyter Notebook in python.

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 female

The database is subjected to supervised learning which means the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to "learn" by comparing its actual output with the "taught" outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabeled data.

To start, I wanted to review three similar but different tasks:

- **feature extraction** and **feature engineering**: transformation of raw data into features suitable for modeling;
- **feature transformation**: transformation of data to improve the accuracy of the algorithm;
- **feature selection**: removing unnecessary features.

Feature Extraction

Here we take the dataset and read the 'csv' file into the Jupyter notebook and convert into numpy.array. using python pandas DataFrame.

Feature Transformation

Monotonic feature transformation is critical for some algorithms and has no effect on others.For data that is very large and the difference between the numerical is very high we use few preprocessing techniques.

In this case I have take Standard Scaler Transformation.

StandardScaler

The idea behind StandardScaler is that it will transform your data such that its distribution will have a mean value 0 and standard deviation of 1. Given the distribution of the data, each value in the dataset will have the sample mean value subtracted, and then divided by the standard deviation of the whole dataset.

The last column of the data is **label** which contains the gender category.It is in the form of a string while others are in *float64* datatype.To change that we do preprocessing taking the class label as 'y' variable which is a dependent variable.

After data preprocessing via StandardScaler using scikit learn the dataset is divided into dependent and independent variables.
X matrix contains all the independent variables and y contains all the dependent variables.
The string male is converted to 1 and female as 0 after feature transformation.

Feature Selection

There are at least two important reasons to get rid of unimportant features. The first is clear to every engineer: the more data, the higher the computational complexity. As long as we work with toy datasets, the size of the data is not a problem, but, for real loaded production systems, hundreds of extra features will be quite tangible. The second reason is that some algorithms take noise (non-informative features) as a signal and overfit.

Selection by modeling:
Another approach is to use some baseline model for feature evaluation because the model will clearly show the importance of the features. Two types of models are usually used: Random Forest or a linear model with Lasso regularization so that it is prone to nullify weights of weak features. The logic is intuitive: if features are clearly useless in a simple model, there is no need to drag them to a more complex one.

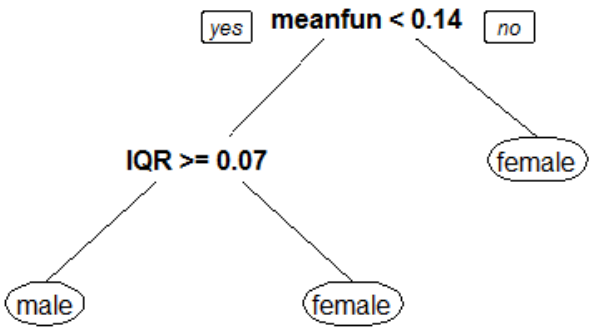
Approach

I have taken reference from Kaggle.
There performance with different algorithms are as follows:
Accuracy(train/test)
Baseline (always predict male)
50% / 50%
Logistic Regression
97% / 98%
CART
96% / 97%
Random Forest
100% / 98%
SVM
100% / 99%

Before fitting my data using various models I used train-test-split model selection followed by cross-validation followed by validation on test sets as it is a large data.

My approach to the data:
CART Algorithm a.k.a Decision tree Regressor.

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.



METRICS USED:

- Accuracy score
- F1 score
- Confusion matrix

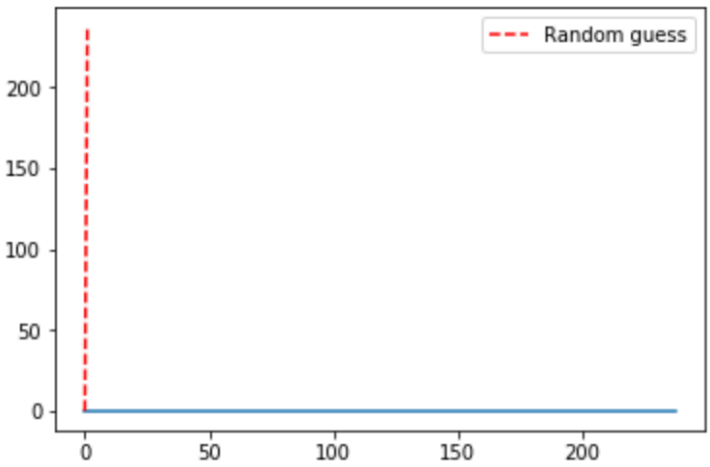
I compared my classification model with an ensemble model - Random Forest Classification and logistic Regression

Sno.	Method	Accuracy Score	F1 Score	Confusion matrix
1	Decision Tree Regression	95.37	95.28	[[1156. 33] [77. 1110.]]
2	Random Forest Classification	**99.7**	96.02	[[1160. 29] [64 1123]]
3	Logistic Regression	97.85		

The data was cross validated using random forest classifier and has got the highest accuracy score when compared to other classifiers.

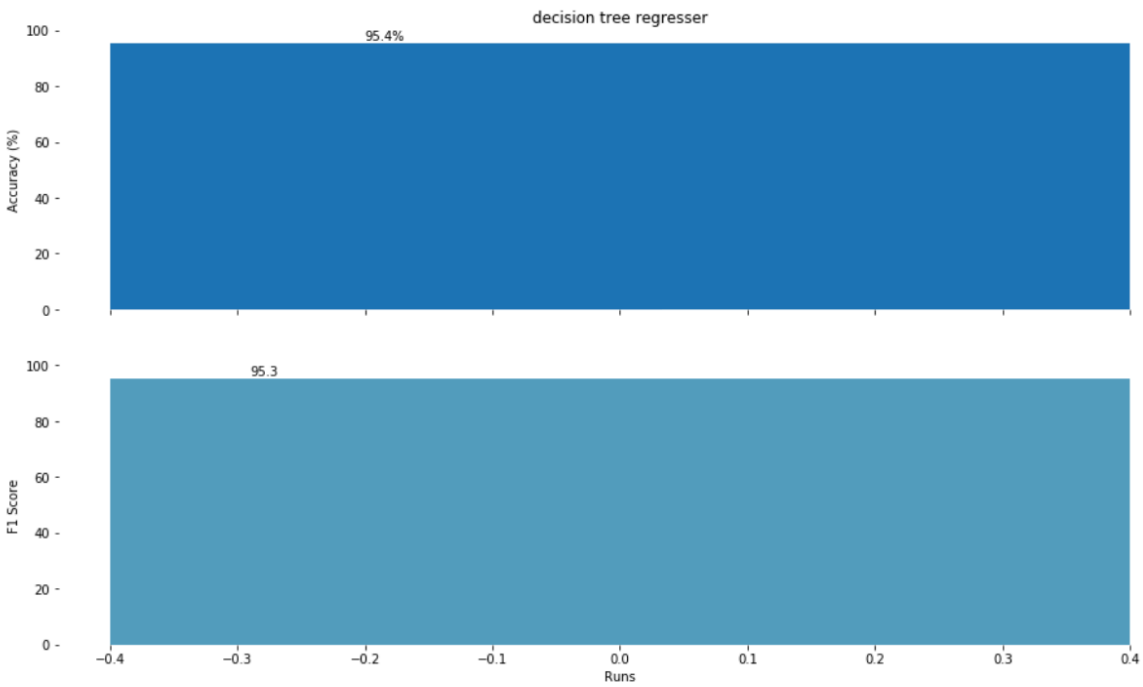
I normalized the confusion matrix fitted from the random forest classifier and the results were satisfying.

```
import matplotlib.pyplot as plt
plt.plot(tpr,fpr)
plt.plot((0,max(tpr)), "r--" ,label = "Random guess")
plt.legend();
```



Then giving the human frequency threshold we have taken true positives and true negatives and plotted a 2-D graph which looks like this.

Later I used seaboard to plot accuracy of decision tree regressor and the result was something like this.



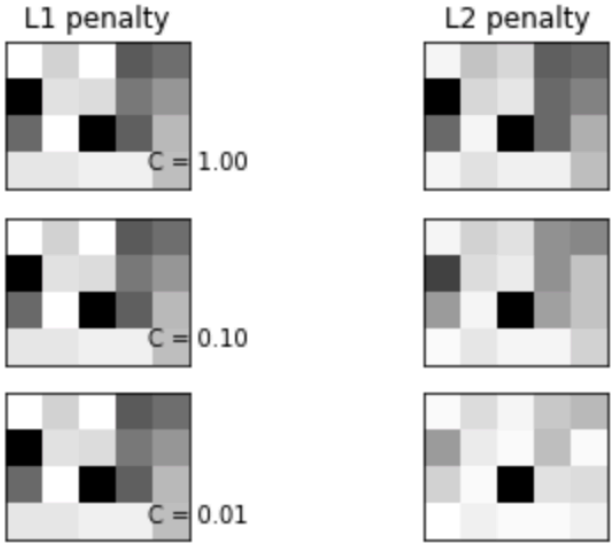
L1,L2 Penalty(Generalization)

We have seen normalization now we see generalization of logistic Regression known as L2 and L1 penalty the Gaussian and Laplace transformation respectively.

Regularization can be used to train models that generalize better on unseen data, by preventing the algorithm from overfitting the training dataset.
Common approaches I found are Gauss, Laplace, L1 and L2. KNIME Analytics Platform supports Gauss and Laplace and indirectly L2 and L1.

- Regularization can lead to better model performance
- Different prior options impact the coefficients differently. Where Gauss generally leads to smaller coefficients, Laplace results in sparse coefficient vectors with just a few higher value coefficients.
-

```
C=1.00
Sparsity with L1 penalty: 15.00%
score with L1 penalty: 0.9785
Sparsity with L2 penalty: 0.00%
score with L2 penalty: 0.9773
C=0.10
Sparsity with L1 penalty: 60.00%
score with L1 penalty: 0.9785
Sparsity with L2 penalty: 0.00%
score with L2 penalty: 0.9735
C=0.01
Sparsity with L1 penalty: 90.00%
score with L1 penalty: 0.9785
Sparsity with L2 penalty: 0.00%
score with L2 penalty: 0.9533
```



Conclusion

- I have observed that there is a lot of overfitting of data.There are many unwanted or non important features due to which the prediction may vary if taken the dataset as a whole with out feature selection.
- The accuracies however showed results to be better performed there is one too many male voices recorded and may efficiently work when there is a high vocal range of an androgynous voice is examined and record.
- Random Forest Classifier works perfectly with this dataset however given the n_estimaters more than 100 the data is deemed to run slowly.