1. **Downloading data:**
    a. Read the website HTML
    b. Extract all the downloadable links from the website, and download them with multiprocessing module.

    Hack: Since internet connection here is not reliable and poor, I added some resume abilities to the code, to solve this problem.
2. **Extracting the tar files:**
    a. Extracting the tar files and save the folders on disk.
3. **Make CSV dataset from the folders:**
    b. For each folder, I find the list of .wav files
    c. after that try to find 'readme' file and extract the name and gender of that Speaker.

    Hack: We have so many typos in gender, name or even readme name file. Some file doesn't even have the readme. I handle most of them in the code.
4. **Feature Extraction:**
    d. I made the class to read the .wav sounds, compute FFT and low pass filter for frequencies.
    e. I used suggested features form the list of overall frequencies and frequencies who are less than 280 Hz.

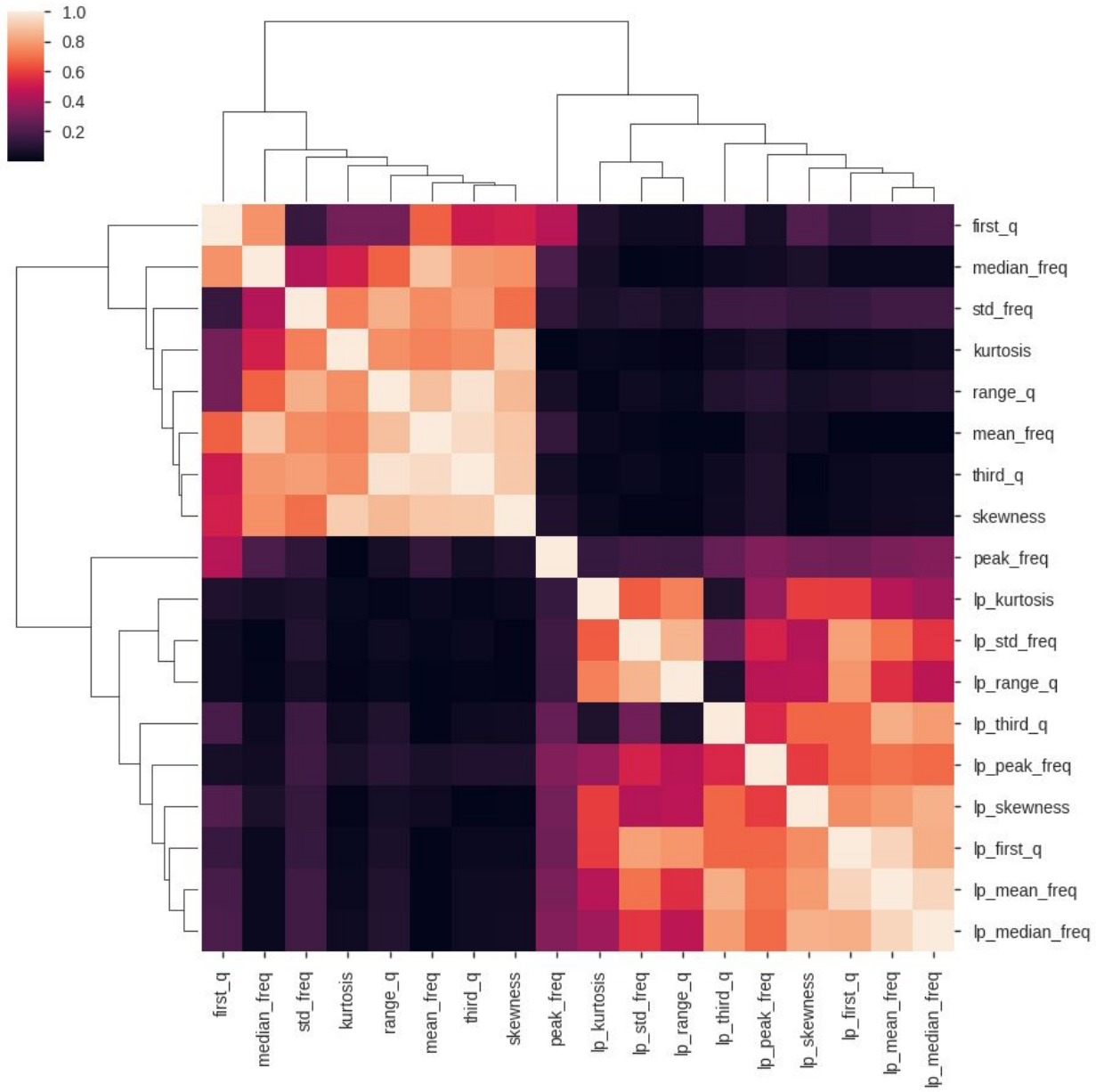
    Problem: Since the soundtracks are not in the same length, we may need to cut the 5 seconds of them, to make features without low pass filter more robust to anomalies.

5. **EDA**:
    a. I reject this features for their high correlations with others:
        i. kurtosis is highly correlated with skewness (ρ = 0.91504) Rejected
        ii. lp_first_q is highly correlated with lp_mean_freq (ρ = 0.93778) Rejected
        iii. lp_median_freq is highly correlated with lp_mean_freq (ρ = 0.94356) Rejected
        iv. range_q is highly correlated with third_q (ρ = 0.97493) Rejected
        v. third_q is highly correlated with mean_freq (ρ = 0.95656) Rejected

    Hack: The bottleneck was reading the files, so I made it parallel. For each file that we read from disk, I computed all the features that I needed and added them to my csv. I couldn't find a good library that computes suggested features for me, so I developed them in numpy.

    Hack: I decided to isolate the above task from each other, so we do not need to run the whole process from start in a case of failure.

For the Logistic Regression, I also drop skewness and lp_range_q, because they made some multicollinearity problems based on blow plot.



6. Splitting Data:

The data was unbalanced in so many ways. Men made 85% of the soundtracks. We have speakers who have only 2 soundtracks vs the one who has 2607 soundtracks and I was not sure if these features are able to distinguish between the genders or they will only represent the speaker. If they will only represent the speakers and I randomly split data between train and test, we will face the data leak phenomena. To avoids this, first of all, I randomly sample 15 sound clip from each of users who have more than 15 soundtracks and merge them with others. After that, I randomly select 90% of each gender and put them in train data, but none of them are in the test data.

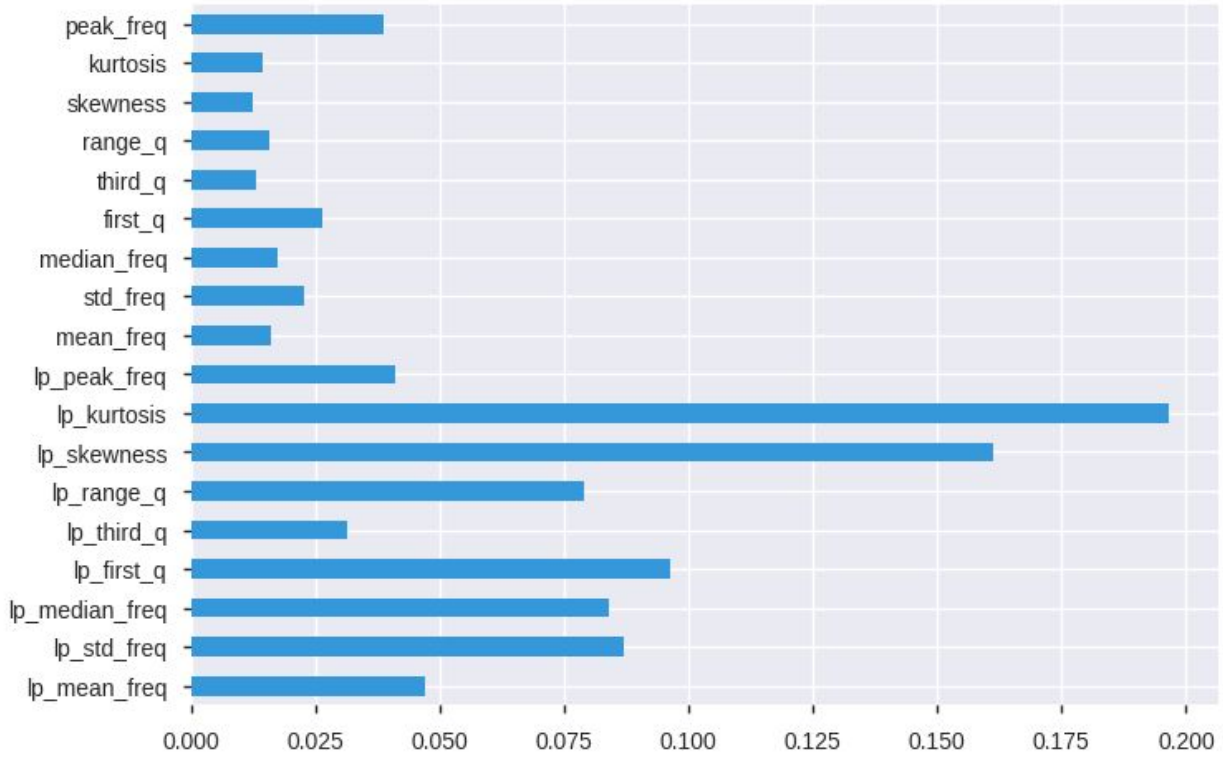
7. Modeling:

First of all, I build a dummy classifier to check what will be the random guess. As I check the data is highly unbalanced, so I need a more robust metric than accuracy. I chose to go with a The Area Under a ROC Curve. With this metric, our dummy classifier has 50% accuracy.

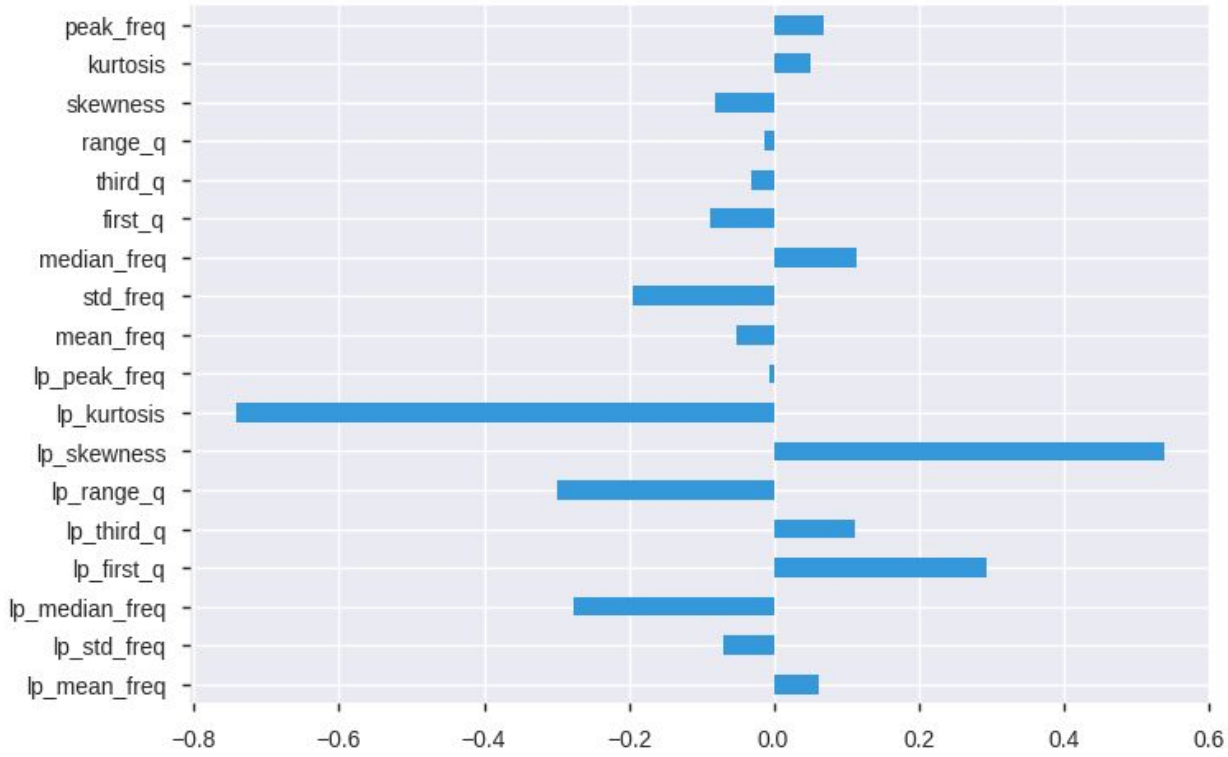
I tried Logistic regression, Random forest and Gradient Boosting with grid search and 5-fold cross-validation to find optimal parameters:

Model	Train	Test
LogisticRegression	0.8028	0.8110
RandomForest	0.9797	0.9340
GradientBoosting	0.9954	0.9227

So Random Forest did better job overall.
Here is the features plot for random forest:



And also I want to add logistic regression features to compare them:



I expect more from lp_mean_freq, It seems none of models find it supper important feature. To improve the model, I think I need to make better features that can better characterise gender, more data cleaning on sounds, and for sure make collect more data from womens can help.