**Song vs non-song screening procedure**

**Intro**

The purpose of this document is to enable the reader to use certain custom-made matlab software for distinguishing songs and non-songs in raw data acquired in song recording sessions. This document also contains description of the data structures that are created in this process.

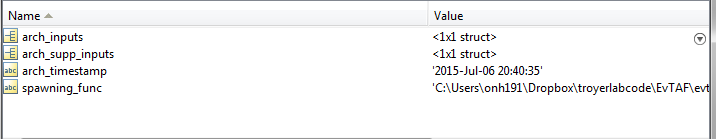
**Make a batch file**

You start with a bunch of audio files. In our case, these files are either in the .wav format or in the .cbin format. If they are in .cbin format, they are accompanied by corresponding .rec and .tmp files as well. These accompanying files do not play any role in screening. As a first step, you should make a batch file listing all the audio files you want to screen. This is achieved by a function called write\_new\_batch.m.

**>> write\_new\_batch**

You should create two batch files. One of them should contain all the audio files and the other one should contain 10-15% of the audio files. The latter will be used for manual classification of song vs non-song. We will refer to the first file as ‘batch\_all’ and to the second one as ‘batch\_hand\_classification’.

A batch file is simply a text file (without an extension) containing a list of files. You can chose to include a fraction of number of files or an exact number of them in the batch by using the function above. In addition to the batch file, the above function also writes a BATCHNAME\_input\_params.mat file. This file contains the following variables:



**Fig 1**

These variables contain information about the input parameters used for making the batch file and the timestamp of the moment of the batch’s creation. The variable ‘spawning\_func’ lists the full path of the function that triggered the writing of the batch. In the current case the triggering function is write\_new\_batch.m. This information is crucial because there are various other functions which also write batch files for various purposes. This information helps you determine the original purpose for which the batch file was written.

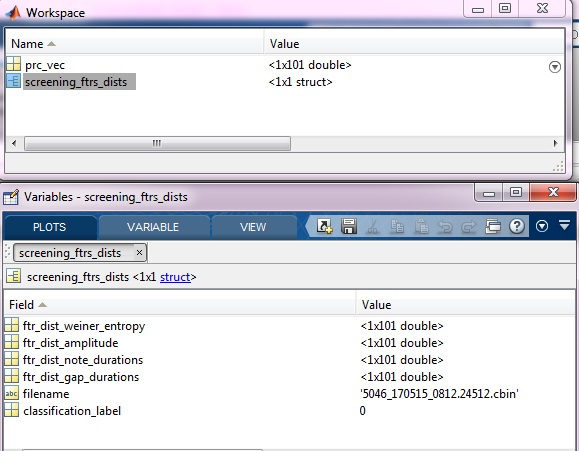
**Calculate screening features distributions**

The next step is to calculate the distribution of screening features for all the audio files. This is achieved using the following function:

**>> calc\_screening\_ftrs\_dists**

Use the batch\_all batch file for this function. This function offers you the list of features it can calculate. You have to choose which ones you want it to compute. Unless there is a specific reason, you should ask it to compute all the features it offers. It will also offer you an opportunity to change some of its default supplementary inputs. Unless you have a specific reason, you should not change these defaults[[1]](#footnote-1).

This function writes an AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file for each file listed in the batch file supplied to it. This file contains the following variables:



**Fig 2**

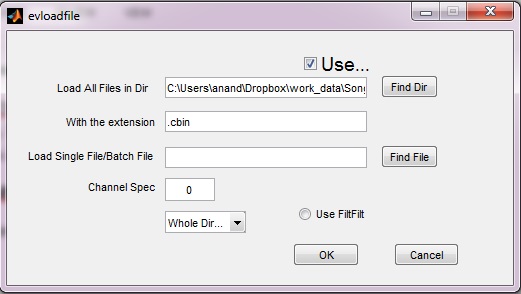
The variable prc\_vec stores the percentiles used for specifying the feature distributions. The variable screening\_ftrs\_dists is a structure that contains the name of the parent audio file, its classification label, and the feature distributions (fields that start with ftr\_dist\_\*FTRNAME). Each file is assigned classification label ‘0’ (corresponding to non-song) while creating the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file.

**Manually classify a random sample of files**

After creating the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file, you should manually classify a random sample of files. Use the batch\_hand\_classifcation batch file for this purpose. The function used to do this is:

**>> evsonganaly\_classify**

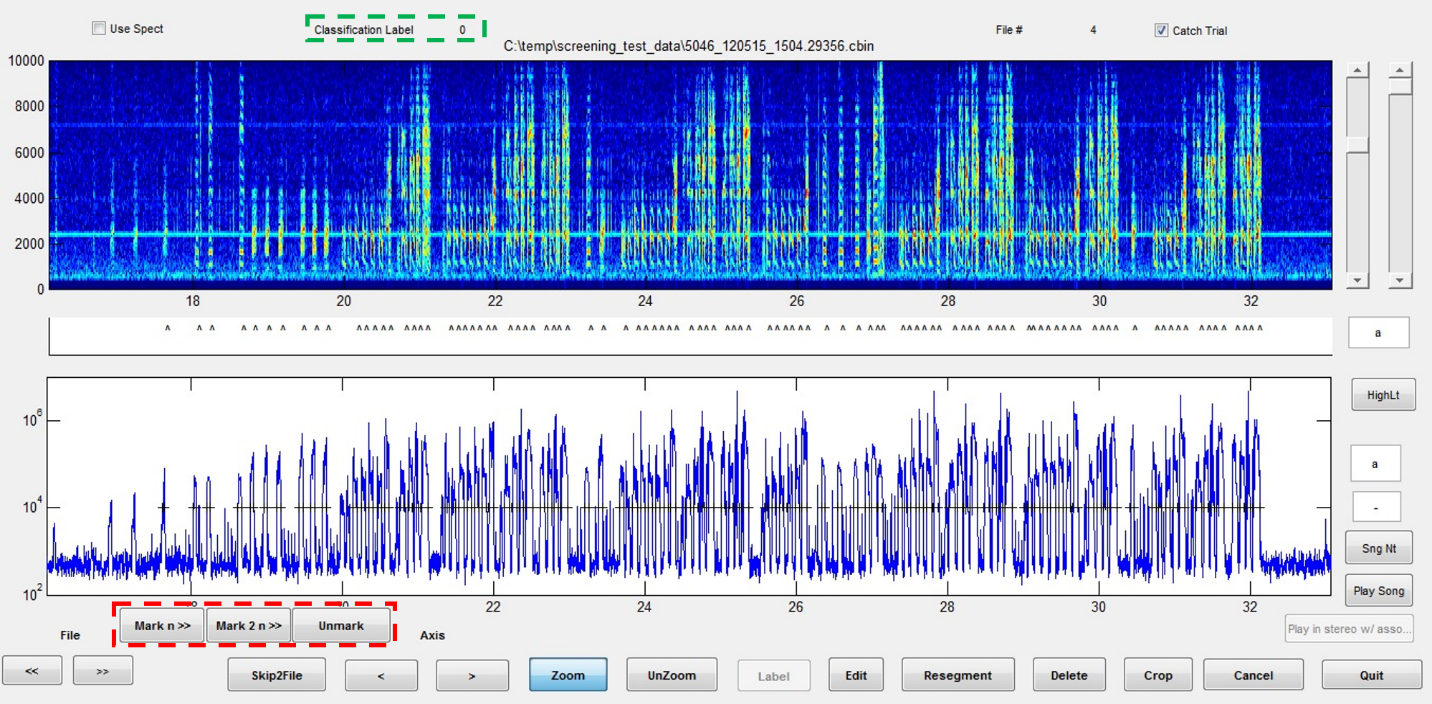
It will bring up the following dialog box:



**Fig 3**

Click on ‘**Find File**’ and select the batch\_hand\_classifcation batch file. Enable the radio button ‘**Use FiltFilt**’ and then click ‘**OK**.’

Next, this function will display the audio file along with a lot of other information and buttons. Three of those buttons are of particular relevance in the manual classification task. These buttons mark the displayed file with the appropriate classification label. These classification labels are stored in a variable called classifier. This variable is written in a file on the disk when you quit the evsonganaly\_classify function. These buttons are highlighted in red in the screenshot below.

 **Fig 4**

These are:

**Mark n >>** - Clicking this button will enter the classification label as ‘1’ and the next file will be displayed.

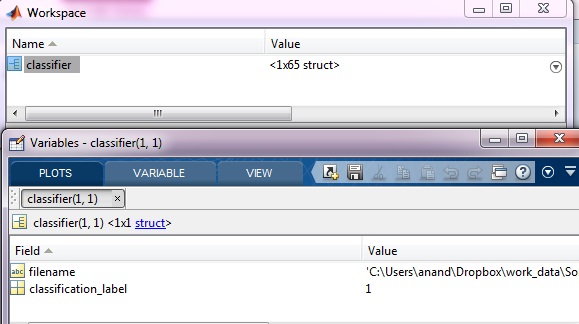
**Mark 2 n >>** - Clicking this button will enter the classification label as ‘2’ and the next file will be displayed.

**Unmark** - Clicking this button will enter the classification label as ‘0’.

Remember that when you write the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file, the default classification label is ‘0’.

The gui also displays the current classification label for the file (highlighted in green).

When you quit the evsonganaly\_classify gui, it writes a BATCHNAME\_classifier.mat file containing the following variables:



**Fig 5**

The variable classifier is a structure containing information about the filename and the classification label of that file. The number of elements in the classifier will be (should be!) equal to the number of files in the batch.

**Copy classification label information into the screening features dists file**

Now that the information about manually assigned classification labels is in the classifier file, you should move that information to the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file as well. That way, the programs that we use for automated screening will have access to that information. This is done using the program:

**>> copy\_classifier\_info\_screening.m**

This function asks you to select the classifier file. It will also ask you if the screening info is stored in a single consolidated file or in separate files for each audio file. Since we have written separate files, you should select the option for separate files. It will ask you if you have screening\_ftrs\_dists file or screening\_ftrs file. You should choose the option corresponding to screening\_ftrs\_dists file, since that is what we have. It then updates the classification\_label field in the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file corresponding to each file in the classifier file.

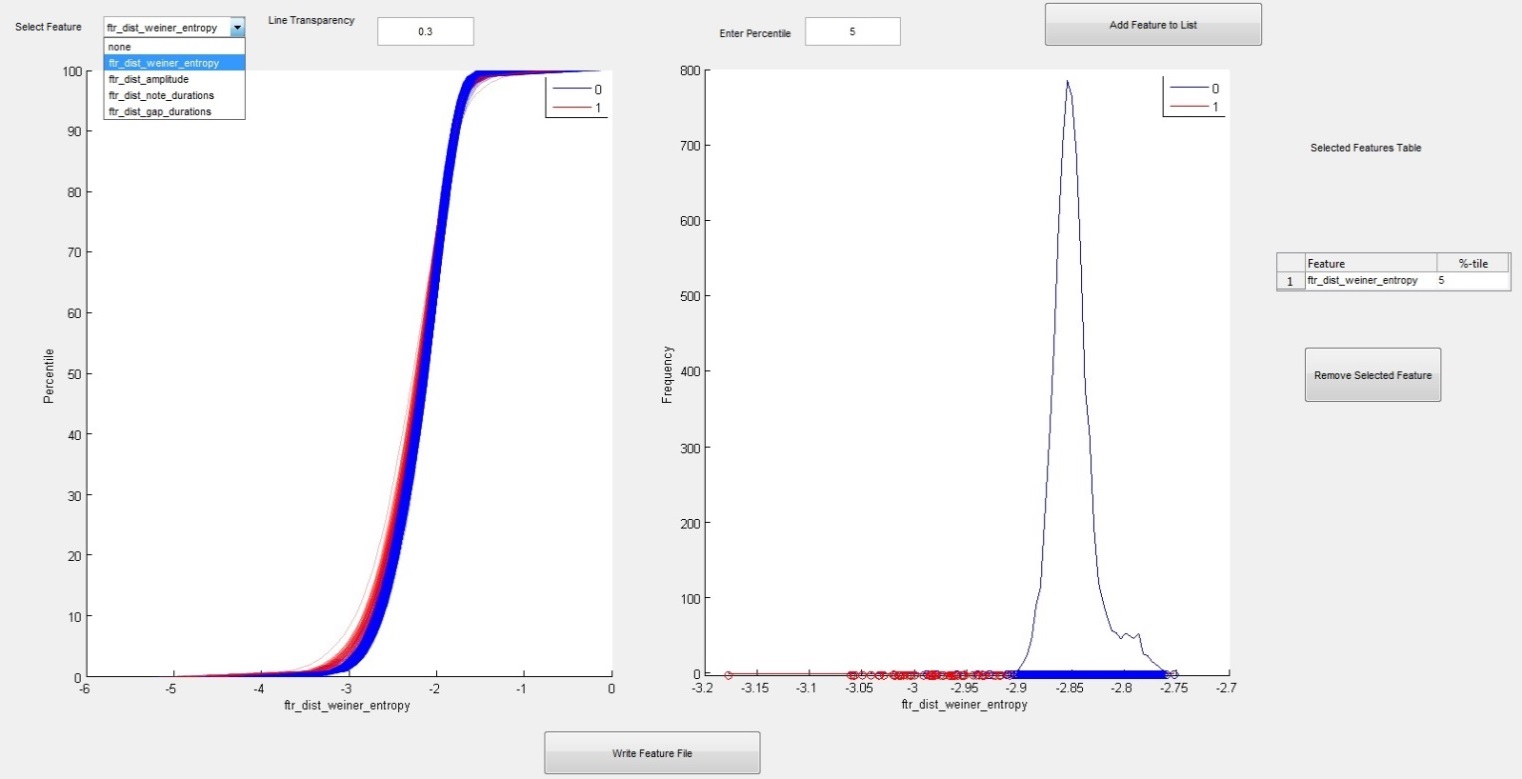
**Select screening features**

The next step is to select the specific percentiles within the previously calculated feature distributions that distinguish between songs and non-songs. The function that does this is:

**>> select\_screening\_features\_2**

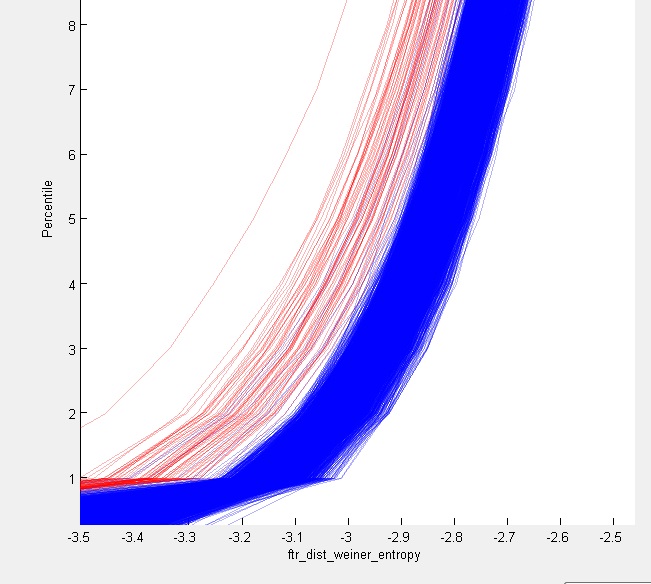
This function will ask you if you have a batch file or a consolidated file containing the screening info for all the files. You should choose the option corresponding to the batch file.

The reason why we call this ‘selecting features’ is because the specific percentiles of the feature distributions will serve as ‘true’ features. A feature, in this sense, is defined as a single number that contains information about a specific aspect of the thing (in this case, an audio file) under consideration. Here is what the interface of this gui looks like.



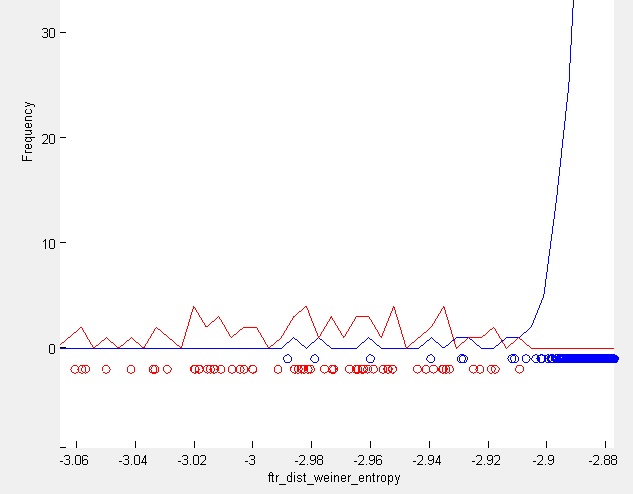
**Fig 6**

The first graph (left) plots the distribution of the feature selected from the drop-down menu for all the files in the batch you selected. In our case, there are files with two different kinds of labels. Those two kinds of files show up in different colors. This plot tells you what part of the distribution is great at distinguishing between the two types of files. In this case it seems clear that percentiles <10 are great at distinguishing between files of type ‘0’ and ‘1’. This will be clearer in the blown-up image below.



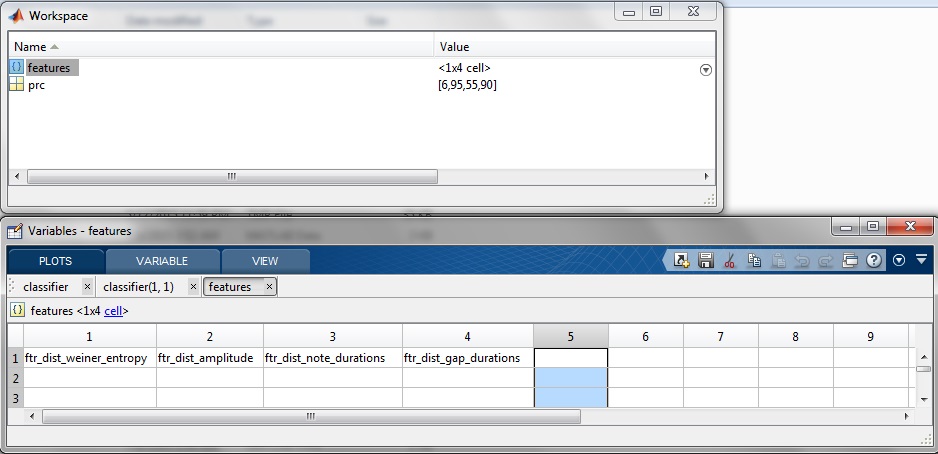
**Fig 7**

You can go with 5 as the percentile value you want to look at. When you enter this value in the ‘**Enter Percentile**’ box, the second graph (right) will show the histogram as well as discrete distribution of all the files for the specific percentile of the selected feature. This is how that looks when blown up.



**Fig 8**

Once you are convinced that a given percentile of a given selected feature is useful for distinguishing the two types of files, you can go ahead and add that feature to the selected features table by clicking on ‘**Add feature to List**.’ Once you are done selecting features and are happy with the ‘**Selected Features Table**’, you can go ahead and click on the ‘**Write Feature File**.’ This will write a selected\_features\_BATHCNAME.mat (or any other name that you choose to give it) file. This file will contain the following variables:



**Fig 9**

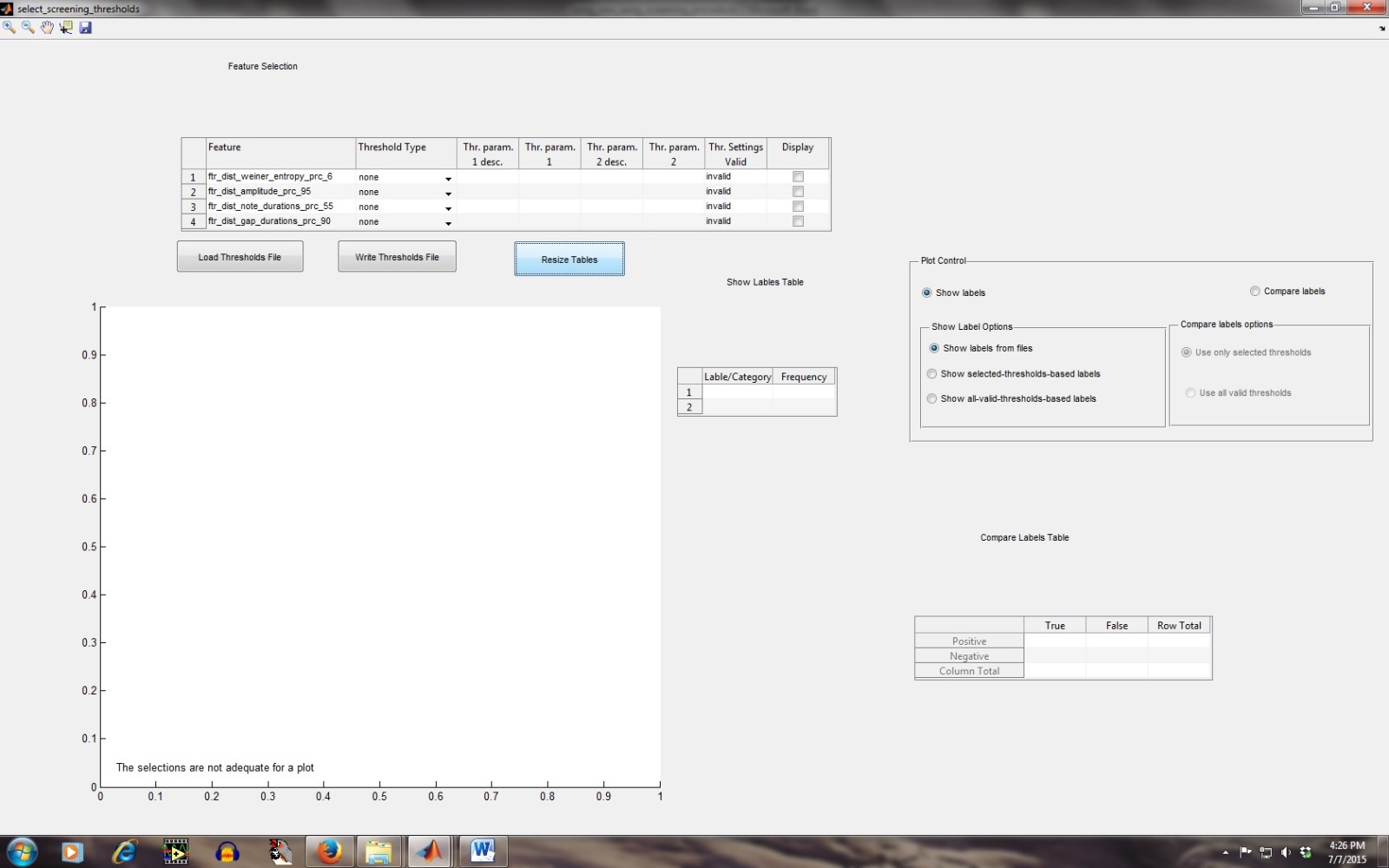
The variable ‘features’ is a cell array containing the names of the feature distributions chosen by the user. The variable prc simply contains the percentiles of those distributions selected by the user.

**Selecting thresholds**

Now we need to set thresholds on these ‘features’ we have selected and use those thresholds to classify files in an automated manner. The function which does that is:

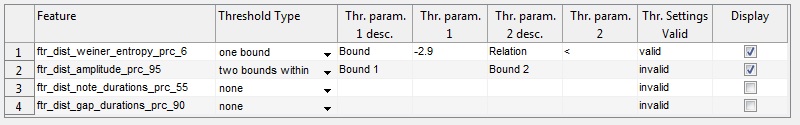
**>> select\_screening\_thresholds**

This function will ask you if you have a batch file or a consolidated file containing the screening info for all the files. You should choose the option corresponding to the batch file. It will also ask you to select the selected\_features\_BATHCNAME.mat file you created previously. This how its interface looks like:



**Fig 10**

The ‘**Feature Selection**’ table lets you set the features that would be plotted on the graph and also lets you set the thresholds on the features for classifying files. Given below is a closer look at the table:



**Fig 11**

Checking a given row in the ‘**Display’** column, displays the distribution of that feature among all the files in the batch. You cannot select more than two features (it will throw a warning dialog box if you do) since it’s only a 2-d plot.

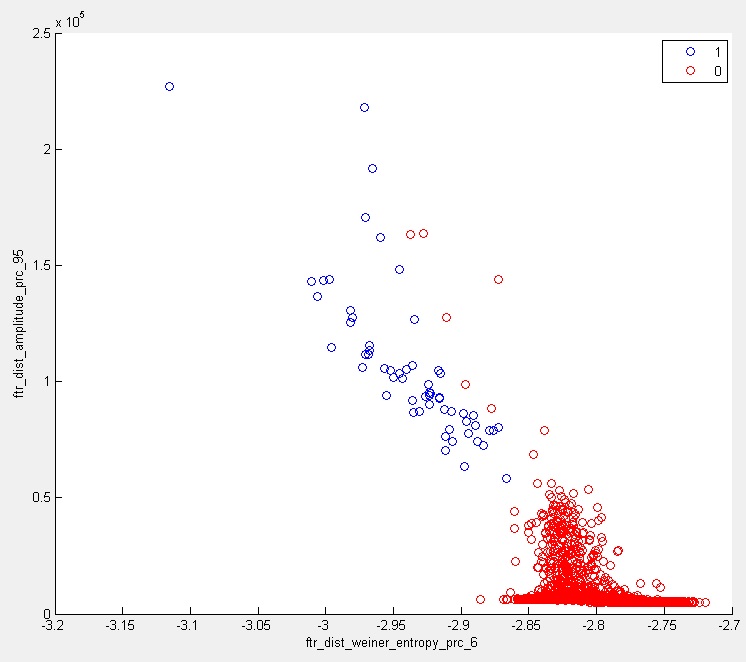
When you select a threshold type for a given feature from the dropdown menu ‘**Threshold Type**’, the columns ‘**Thr. Param. 1 desc.**’ and ‘**Thr. Param. 2 desc.**’ will show the description of the two parameters needed to fully specify the threshold for that particular threshold type. The following table shows the descriptors for all threshold types and the corresponding valid inputs.

**Table 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Threshold Type** | **Thr. Param. 1 desc** | **Thr. Param. 1 valid input** | **Thr. Param. 1 desc** | **Thr. Param. 1 valid input** |
| One bound | Bound | Any real # | Relation | Any operator out of: <,<=,>,>=,==,~= |
| Two Bounds Within | Bound 1 | Any real # | Bound 2 | Any real # |
| Two Bounds Outside | Bound 1 | Any real # | Bound 2 | Any real # |
| Distance Within | Center | Any real # | Distance | Any real # |
| Distance Outside | Center | Any real # | Distance | Any real # |

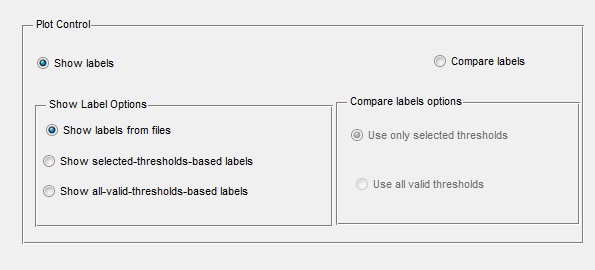
If both the threshold parameters for a given feature are valid, the corresponding cell in the ‘**Thr. Settings valid’** column will display ‘**valid**.’

For the settings shown in the previous image, this is how the plot would look like:



**Fig 12**

Next, let’s turn to the ‘**Plot Control**’ panel:



**Fig 13**

This panel controls the manner in which the data is plotted.

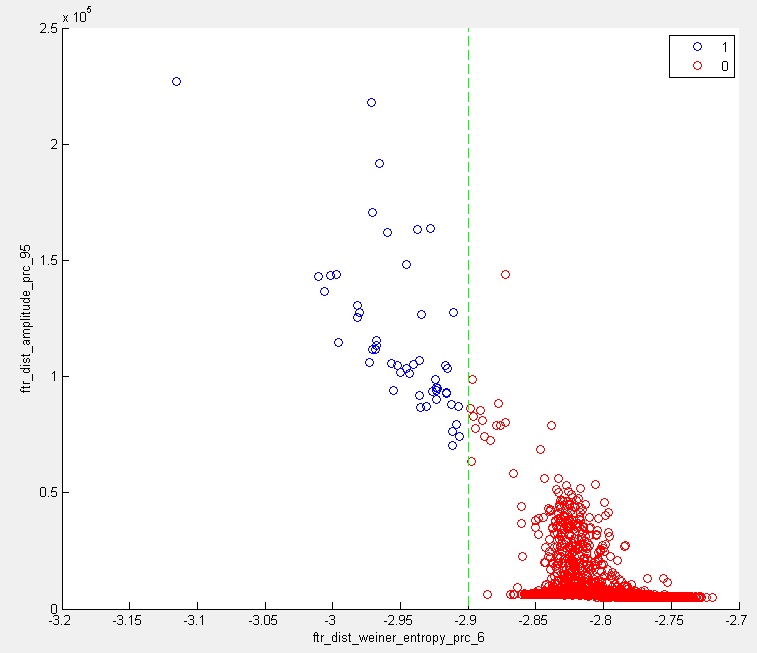
Selecting ‘**Show Labels**’ will simply show files with different classification labels in different colors. The graph shown above illustrates this. The two colors correspond to the two labels types to which the files in this dataset belong.

The panel ‘**Show Label Options**’ controls how these labels are determined. Keep in mind that if a file satisfies given thresholds, it is assigned a classification label 1. The table below explains how the options differ in determining labels.

**Table 2**

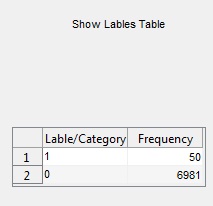
|  |  |
| --- | --- |
| Option Selected | How labels are determined |
| **‘Show labels from files’** | The value stored in the **classification\_label** field of the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat will be used to determine the classification label. |
| ‘**Show selected-thresholds-based labels**’ | Valid threshold settings from among the features selected for display are applied to each file to determine its classification label. |
| ‘**Show all-valid-thresholds-based labels**’ | All valid threshold settings (and not just from among those selected for display) are applied to each file to determine its classification label. |

The plot below shows an example of a graph that results when ‘**Show selected-thresholds-based labels**’ is selected in conjunction with the settings in **Fig 11**.



**Fig 14**

Before we go on to ‘**Compare Labels Options**’, let’s take a small detour to look at ‘**Show Labels Table**.’ This table simply shows the frequency with which each label occurs in the data set. Labels are, of course, determined by the setting in ‘**Show Label Options.**’ This is how the table looks corresponding to the graph in **Fig 14**.



**Fig 15**

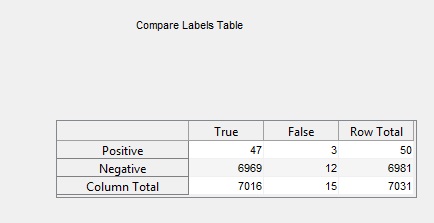
Now let’s look at ‘**Compare Labels Options**.’ Selecting ‘**Compare Labels**’ radio button will enable the ‘**Compare Labels Options**’ panel. In this mode, the plotted data shows the result of the comparison between the classification label obtained from the AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file and that obtained by checking if the given file satisfies the thresholds. Each file is classified into four types. The table given below explains how it is done.

|  |  |  |
| --- | --- | --- |
| Label from AUDIO\_FILENAME\_screening\_ftrs\_dists.mat file | Label determined by thresholds | ‘Compare labels’ - file type |
| 1 | 1 | True Positive |
| 0 | 0 | True Negative |
| 1 | 0 | False Negative |
| 0 | 1 | False Positive |

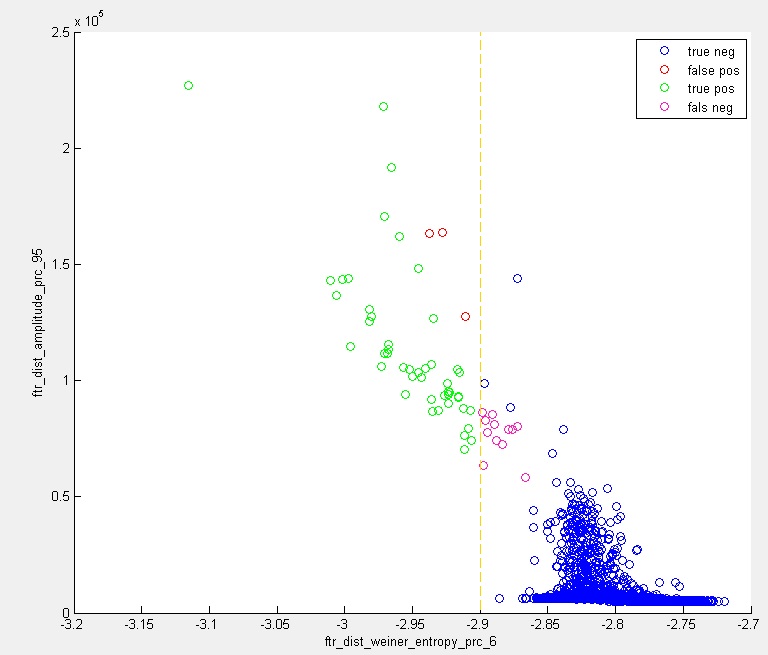
The two options within ‘**Compare Labels Options**,’ ‘**Use only selected thresholds**’ and ‘**Use all valid thresholds**’, control how the threshold-based labels are determined in a manner similar to what is described in the last two rows of **Table 2**.

The plot given below (**Fig 17**) illustrates how a plot looks when Compare labels is selected.

The table ‘**Compare Labels Table**’ shows the numbers of files belonging to each type.

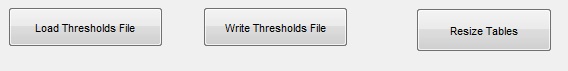


**Fig 16**



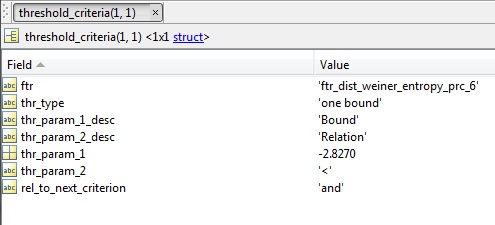
**Fig 17**

That leaves us with only three buttons left to understand. They are:



**Fig 18**

Clicking the ‘**Write Thresholds File**’ opens a dialog box asking the user to place the thresholds file in an appropriate folder and name it. All valid thresholds from the ‘**Feature Selection**’ table (**Fig 11**) are written to the thresholds file. The thresholds file contains a structure called ‘**threshold\_criteria**’ with the following fields.



**Fig 18**

It is basically the same information that is stored in the ‘**Feature Selection**’ table. The field **rel\_to\_next\_criterion** defines how the result of this threshold criterion is combined with that of the next criterion. This field is by default set to ‘**and**.’

Such threshold files can be loaded into the gui by clicking on the button ‘**Load Thresholds File**.’

Finally, the button ‘**Resize Tables**’ resizes all the tables in the gui if their appearance has gone out of whack due to resizing of the gui window.

**Writing batch files based on thresholds file**

The next and the final task is to write batch files listing all the files that satisfy thresholds and the ones that do not. This is done using the following function.

**>> write\_batch\_based\_on\_thresholds\_file**

This function will ask you if you have a batch file or a consolidated file containing the screening info for all the files. You should choose the option corresponding to the batch file. This function will also ask you to select the thresholds file you created in the previous step. Additionally it will ask you if you want to create a batch file for files that do not satisfy the thresholds. You should answer in affirmative. It will also ask you if you want to use features dists file to retrieve feature information. Answer in affirmative.

It will eventually tell you the numbers of selected and rejected files and ask you if you want to continue to writing the files. This check lets you do a sanity check. If a threshold was set inappropriately, these numbers should look weird. Once you ask it to go ahead, it will ask you to place the batch files in some directory and name them. So now you have two batch files, one containing the names of all the files that satisfied the thresholds and should contain songs (aka, for the purpose of this document, **batch\_selected**). The second one should contain files without songs (aka, for the purpose of this document, **batch\_rejected**).

**Checking the performance of the screen**

Now that you have two batch files listing the files that should be songs and non-songs, you may want to check how good the screening was. Specifically, we are interested in how many non-songs slipped into the **batch\_selected** (aka false positives) and how many songs were lost to **batch\_rejected** (aka false negatives).

You can draw a random sample of files from these batch files for hand classification. The code that you use for this purpose is the following.

**>>write\_batch\_by\_sampling\_from\_batch**

It will ask for the batch file you want to sample from. It will also ask you the proportion of or the number of files from the parent batch that you want to stuff into the new batch file. Next it will ask you to name and place the new batch file at an appropriate location.

You should do the above step for both **batch\_selected** and **batch\_rejected**.

You can use evsonganaly\_classify to classify the files in these daughter batch files as songs and non-songs. Next you can use the following code to display the results of the hand classification.

**>>show\_classifier\_results**

This code will ask you for a classifier file and show the classification labels present and the counts of those labels for the files listed in the classifier. Use these numbers to determine the proportion of false positives and false negatives.

**Archiving results of screening**

The next step is to gather all the individual screening files and storing them in an aggregate file so that less space is occupied moving data is easier. The code for doing that is:

**>>** **write\_screening\_info\_aggregate\_and\_delete**

This code will ask you to select a batch file and enter if the files to be aggregated are screenings\_ftrs\_dists files or screening\_ftrs files. It will also ask you if you want to delete the individual screening files after the aggregate has been made. For the output, it will ask to name and set location for the aggregate file.

This aggregate file, the selected\_features file, and the thresholds file are the three files that you need to store for archiving the information about screening.

**Deleting the rejected files**

This step should only be done if you are sure that your screening is accurate. You should consult the PI before you do it. Data deleted in this step may be lost forever.

The code we use for deleting the rejected files from **batch\_rejected** is the following.

**>>delete\_files\_in\_batch**

This code will ask you to select a batch file and ask you if you want to recycle the deleted files or not.

**Epilogue**

Once you have screened one day from a session (a session being a period of, possibly multiple days over which the recording conditions and the bird being recorded are the same), all you need to do for the other days is to make sure that the clusters in feature space look qualitatively similar to the ones observed for the current day. If that is the case, you can simply screen that day without any hand classification.

That’s all folks!!

1. The supplementary input ‘write\_consolidated’ (when changed to 1) gives you the ability to write a single consolidated file containing the screening features distributions for all the files in the batch. Don’t change its value to 1 for now. Later in this document, we may see a scenario where it may be useful to change its value to 1. [↑](#footnote-ref-1)