Oregon State University
College of Engineering

CS Capstone Progress Report

Winter 2020

Building More Self-Aware Everyday Robots

Prepared for
Dr. Naomi Fitter

Prepared by
Group 43
Comedy Robot Team
Timothy Bui
Yuhang (Tony) Chen
Brian Ozarowicz
Trevor Webster

Abstract

This document provides a summary of the progress made during Winter term on the 'Building More Self-Aware Everyday Robots' Capstone project. It describes the code development that has been done in relation to the established project goals and gives the plan for upcoming implementation onto hardware and project stretch goals.

CONTENTS

1 PURPOSE AND GOALS

The purpose of our project is to study how human-robot interactions can be improved by observing the ability of a robotic stand-up comedian to read and respond to cues from its audience. Our goals are to use machine learning on a dataset of recordings from previous comedy performances to train models for detecting laughter and using that response to decide whether a joke was a hit or a bomb.

This analysis can be used by the robot to make real-time adjustments to its performance based on the perceived audience preferences in order to improve the success of their interpersonal communications. The work also has potential applications outside the field of comedy in improving the experience of interactions with AI assistants and other autonomous systems.

2 TERMINOLOGY

Performance: The audio data recorded by the robot during its stand-up comedy routine. Each performance contains multiple jokes and every joke has a corresponding audience response. The audio for each joke is split into the following two files:

- **Joke Audio:** The audio in the period of time in a performance where the robot is telling a joke
- **Post-Joke Audio:** The audience's reaction to the joke; consisting of the audio from the time after the joke finished until the time when the next joke starts being told

---

Audience Response: The audience's response to a joke told during a performance. The response can be categorized in one of the three following ways:

- **Positive Response:** Laughter in chorus from the crowd
- **Neutral Response:** Laughter from an individual audience member or quiet and scattered laughter
- **Negative Response:** Silence or relative silence from the crowd

---

3-Class: The division of the jokes into negative responses (-1), neutral responses (0), and positive responses (1)

2-Class: The division of the jokes between negative responses (0) and neutral or positive responses (1)

Feature: Specific information extracted from audio that is used to train the classifier

---

Praat: Software used to extract features from the audio

Human Rating: The result from a single human rating each individual joke based on how they would perceive the audience's response if they were giving a stand-up comedy performance

---

Ground Truth Rating: The rating found by taking the median of the individual human ratings for each joke

Post-Joke Classifier: The classifier which tries to determine the audience response in the post-joke audio file. It uses the following features:

- **Pitch:** The mean frequency of the audio file, measured in Hertz
- **PitchSd:** The standard deviation of the frequency of the audio file, measured in Hertz
- **Intensity:** The mean amplitude of the audio file, measured in decibels
- **IntensitySd:** The standard deviation of the amplitude of the audio file, measured in decibels
- **MaxSound:** The maximum amplitude of the audio file, measured in Sinc70
- **MinSound:** The minimum amplitude of the audio file, measured in Sinc70

Mid-Joke Classifier: The classifier which tries to determine the audience response in the joke audio file. It uses the following features:

- **Min_Pitch:** The minimum pitch of the the audio file, measured in Hertz
- **Max_Pitch:** The maximum pitch of the the audio file, measured in Hertz
- **Mean_Pitch:** The average pitch of the the audio file, measured in Hertz
- **SD_Pitch:** The standard deviation of the pitch of the the audio file, measured in Hertz
- **Min_Intensity:** The minimum amplitude of the audio file, measured in decibels
- **Max_Intensity:** The maximum amplitude of the audio file, measured in decibels
- **Mean_Intensity:** The average amplitude of the audio file, measured in decibels
- **SD_Intensity:** The standard deviation of the amplitude of the audio file, measured in decibels

3 Current Status

We had three team members assign human ratings to the jokes in our dataset and averaged those ratings to produce our initial ground truth ratings for use training our classifiers. The results were later found to be invalid for use in the machine learning phase of the project, for reasons detailed in the Problems section of this report, however they were still able to be used to calculate the human rater accuracy level which could be used as a benchmark for our classifier results. A single team member then rerated all the jokes to produce a valid ground truth file for use with the classifiers.

We have implemented separate classifiers for the mid-joke and post-joke analysis which determine whether laughter occurred or not while the joke was being told and after the joke concluded respectively. The classifiers can use several different learning models; the accuracy rates achieved by each model are shown in the Results section of this report. The post-joke classifier can operate as both 3-Class - categorizing the audience response as positive, negative, or neutral - or 2-Class - categorizing as either laughter present or not present. The mid-joke classifier is only 2-Class - simply deciding whether laughter, of any type, was detected during the telling of the joke.

4 Work Remaining

In regard to the official project deliverables established in our documentation last term what remains is to combine the mid-joke and post-joke analysis pieces to produce one overall joke rating from our classifier, then implement the classifier onto the robot hardware and test its use in real-time operation. After this is achieved we have some potential

stretch goals to begin work on which are beyond the scope of the course grading criteria, but will be extra production to come out of this project to assist our client in future development.

One goal is to migrate the robot's programming from the current use of a program called Choregraphe to using the Python SDK. This will allow for better adaptability in future programming of the robot's performance as Choregraphe has become difficult to use as the routine becomes gradually more complex.

Other potential stretch goals include conducting human trials of live performances using the new classifier to observe its decision making ability in real-world operation compared to over the previously recorded dataset, as well as preparing the presentation of our results to be the foundation of a new research paper.

5 Problems and Solutions

The original classifier program provided by the client from a previous team's work assumed the jokes were being read in from the ratings csv in the order they had been told during the performance. Our ratings were generated using an annotator program that the previous team did not use, so our jokes were actually randomly ordered. This resulted in the classifier incorrectly matching the featurue data from the Praat files to the joke names, producing invalid results. Once this issue was identified we added new functionality to the classifier to ensure the randomly ordered jokes were assigned their correct Praat data before the machine learning was performed.

A bug was found in the annotator program that caused it to skip the audio files for jokes that were follow-up tags, causing our ratings to be incomplete. We notified the client and the issue was quickly fixed. We then reran the annotator to produce ratings for the audio that had been skipped before to complete the human ratings csv.

It was later noticed that the annotator was not correctly matching some joke names to the ratings they should have. Investigation found that this was happening because it was not actually playing all of the audio files from a performance, but was replaying half of the files twice and giving their ratings to joke names that never had their corresponding audio played. The result was that the ground truths we had been using were invalid as there was no way to determine which ratings were matched to their correct jokes. After discussion with the client it was determined that the cause of this bug would be too difficult to track down to be worth the time it would take away from proceeding with the rest of the project. The solution arrived at was for one human to redo all the joke ratings by hand to produce a ground truths csv that we knew was complete and correct.

6 Results

This section describes the preliminary results from our work on the post-joke and mid-joke classification and includes some data visualization graphs produced from both parts of the project.

6.1 Post-Joke Analysis

For a 3-Class division of the data we have two datasets where three human raters have listened to the post-joke audio and rated the audience response then these three human ratings were combined to form a ground truth rating. In the first dataset the individual human ratings matched the ground truths with accuracy rates of 82%, 84%, and 87%. In the second dataset the human ratings matched the ground truths with accuracy rates of 69%, 70%, and 89%. Based on

these results we determined that our original goal of an accuracy rating of 85% for the post-joke classifier compared to the ground truth ratings would be reasonable, since this is about equivalent to what we would expect from a human classifying the data.

We used two validation techniques with our classifiers. The first approach is to leave the data from one performance out for validation then train the classifier using the data from the remaining performances. We repeated this for each performance and took the mean accuracy across the classifiers as our final accuracy value. The second approach is holding out 20% of the data, randomly selected from across the entire dataset, then training the classifiers on the remaining 80% of the data. We repeated this process 100 times and took the average accuracy across all the trials as our final value.

At our current standing in the project we have achieved our goal with an accuracy rating of around 85% using leave one performance out validation and around 88% using hold out 20% validation with our best classifier and around 83-86% accuracy with all classifiers. The individual accuracy results for each classifier are shown in the images provided later in this section. These results seem to indicate that the classifier is able to detect laughter from an audience in a stand-up comedy environment with about the same level of accuracy as a human.

There are some limitations to the current results and scope of the project. For example, we were working with a limited sample size of the number of human raters, the number of performances, and the number of performance environments. We have also been unable to test the classifier in a real world environment. As we implement the classifier onto the robot hardware and conduct live performance tests we can determine if there is a significant difference in the real-time operation compared to operating over the dataset of pre-recorded audio.

6.2 Mid-Joke Analysis

For the mid-joke 2-Class division of data we had one dataset where one human rated all of the audio clips. We chose to do it this way because at the time we were unsure if our method would prove to be successful and wanted to keep this task as cheap as possible in the time investment required. When annotating the audio we simply looked for if there was laughter heard or not, regardless of its qualities, hence the 2-Class division being sufficient.

We used leave-one-out validation to verify the accuracy of our classifiers. Similarly to the post-joke validation, we would leave the data for one full performance out and train the classifier using the data from the other performances. This was then repeated for all the performances in the dataset and the mean accuracy across all performances was used to determine our final accuracy.

Currently we have around 73% accuracy with the Gaussian Radial Basis Function (RBF) kernel in our support vector machine (SVM) model. While this is not completely accurate by human standards, it should give us a good enough reading on whether the audience laughed mid-joke or not. When combined with the post-joke analysis the laughter detection should be more than sufficient to make an overall decision for the joke.

6.3 Data Visualization

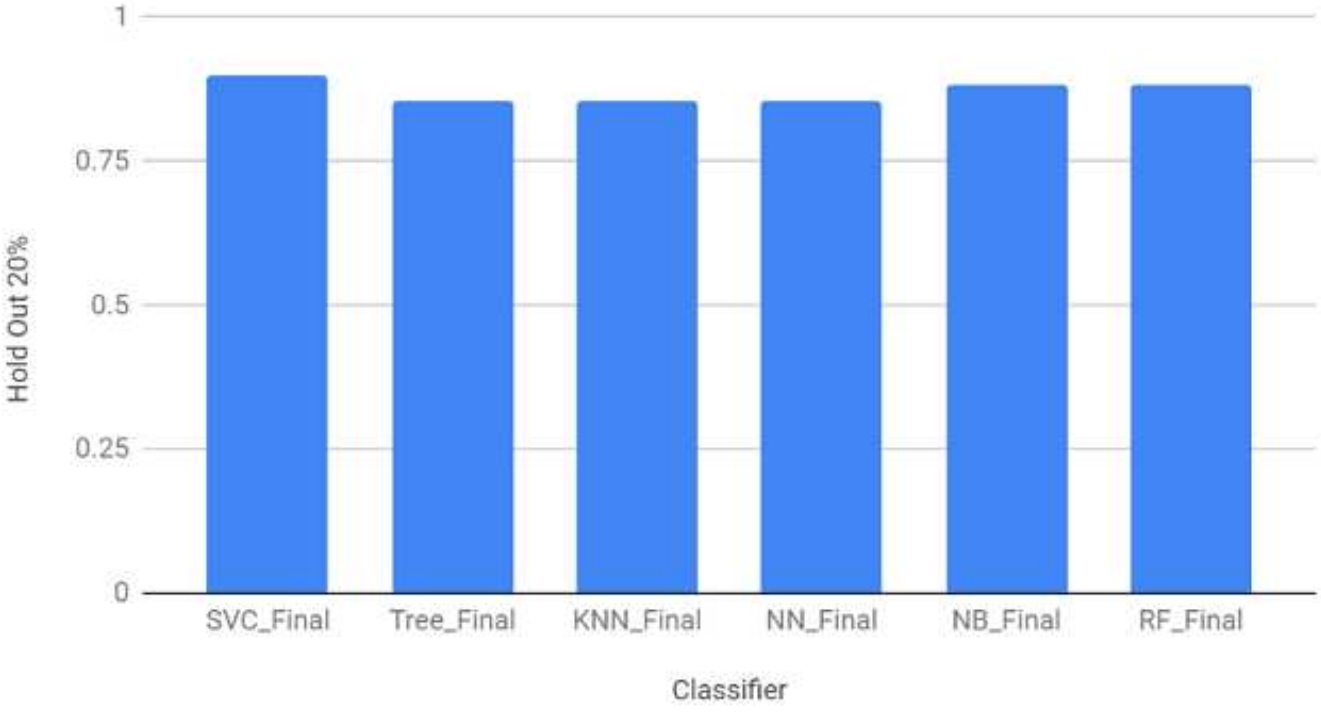

Fig. 1. Classifier Results

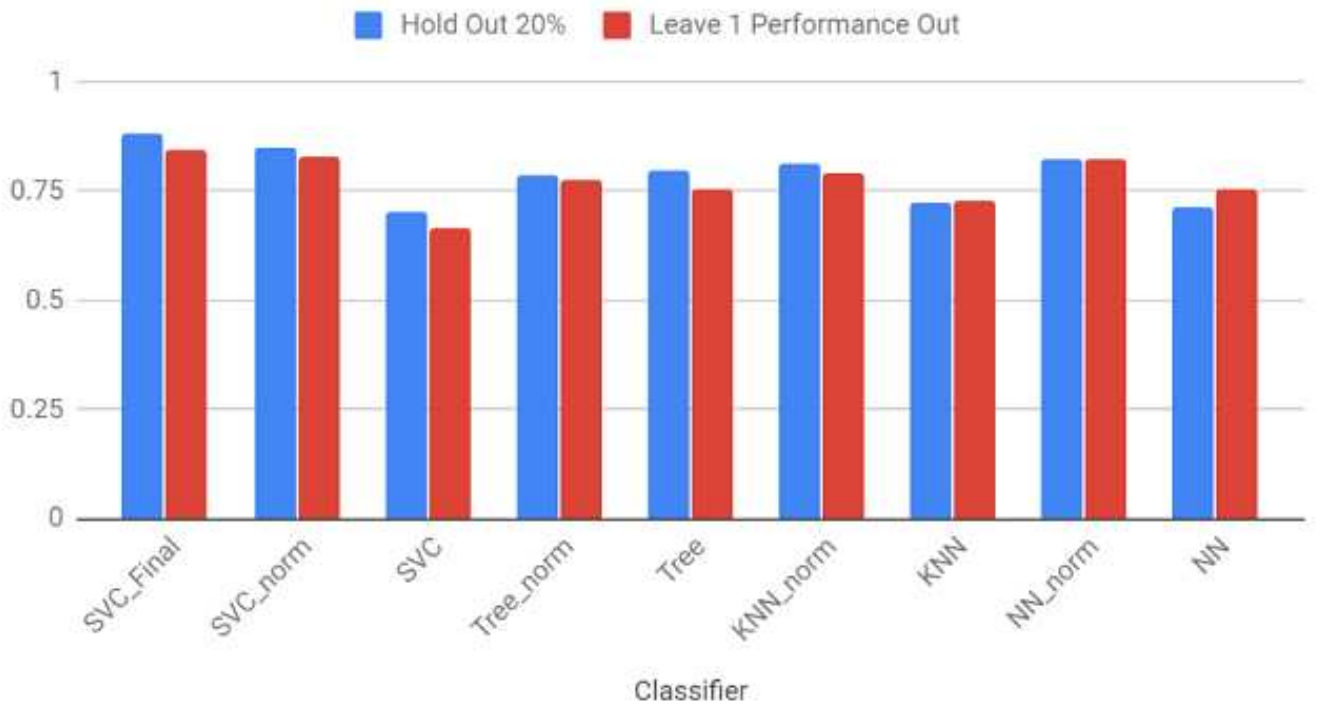

Fig. 2. Classifier Accuracy

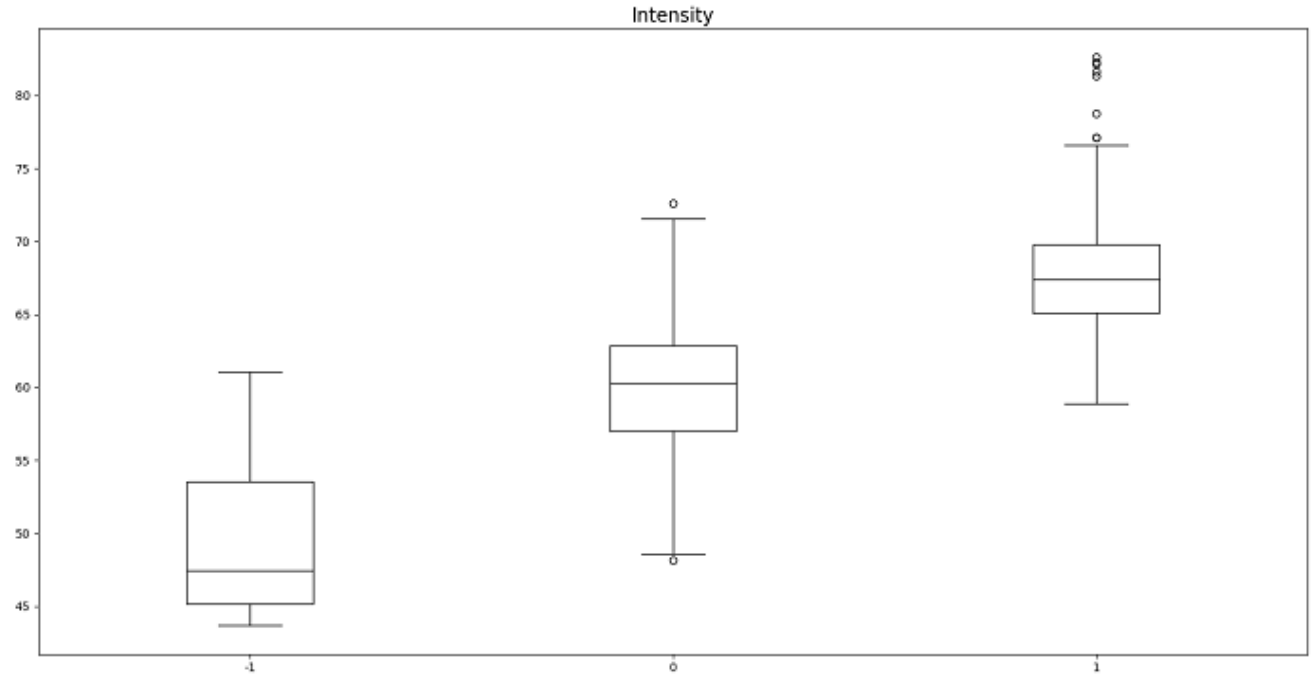

Fig. 3. Distribution of Intensity based on Ground Truth Ratings for Post-Joke Audio

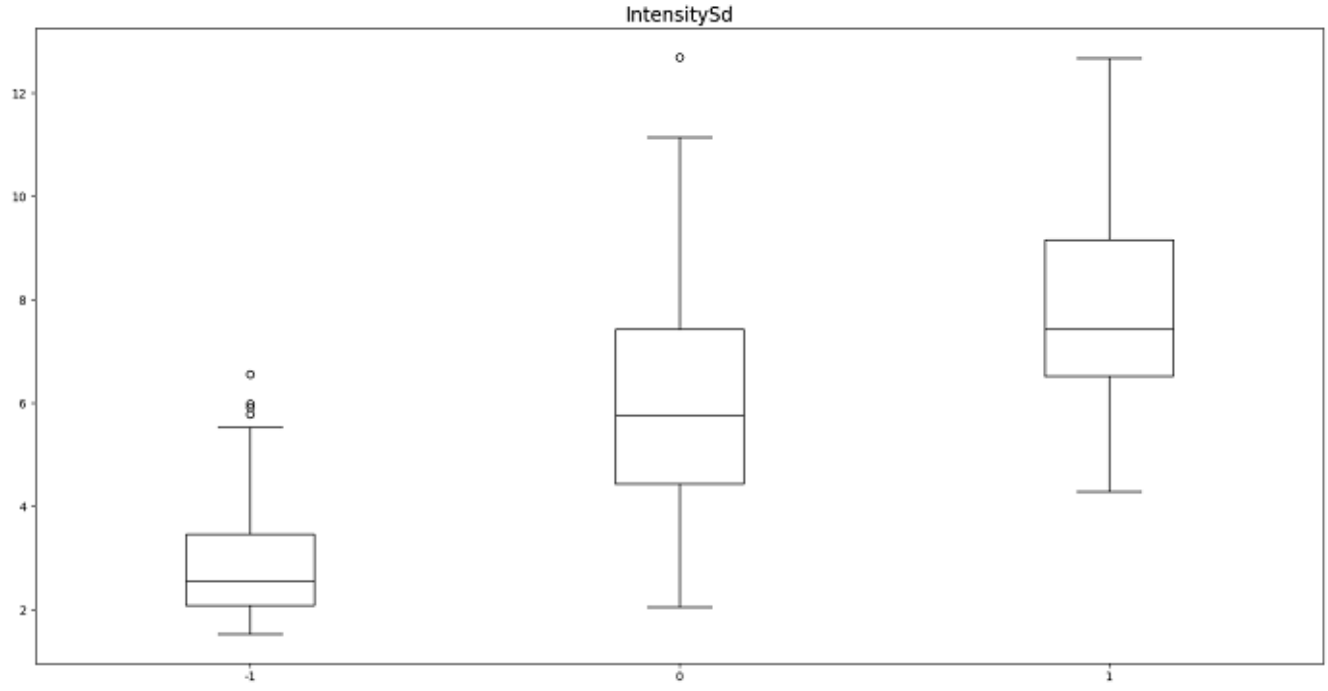

Fig. 4. Distribution of Intensity Standard Deviation based on Ground Truth Ratings for Post-Joke Audio

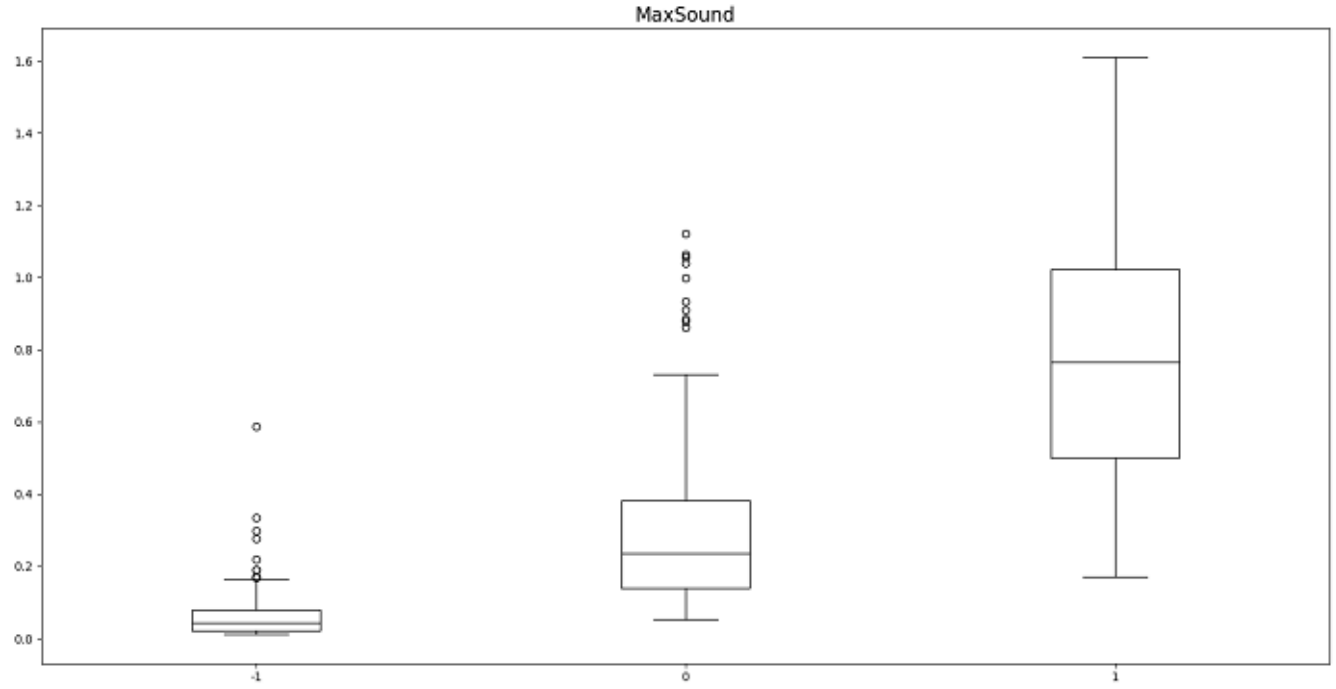

Fig. 5. Distribution of Max Sound based on Ground Truth Ratings for Post-Joke Audio

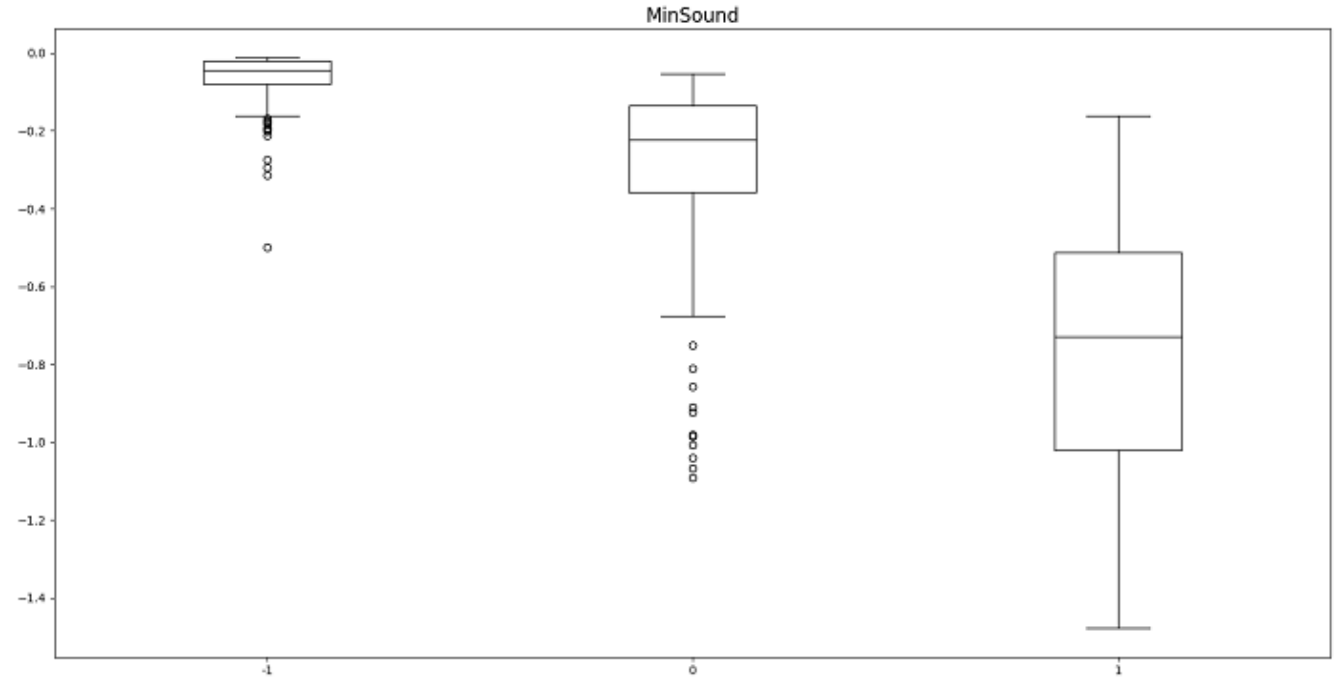

Fig. 6. Distribution of Min Sound based on Ground Truth Ratings for Post-Joke Audio

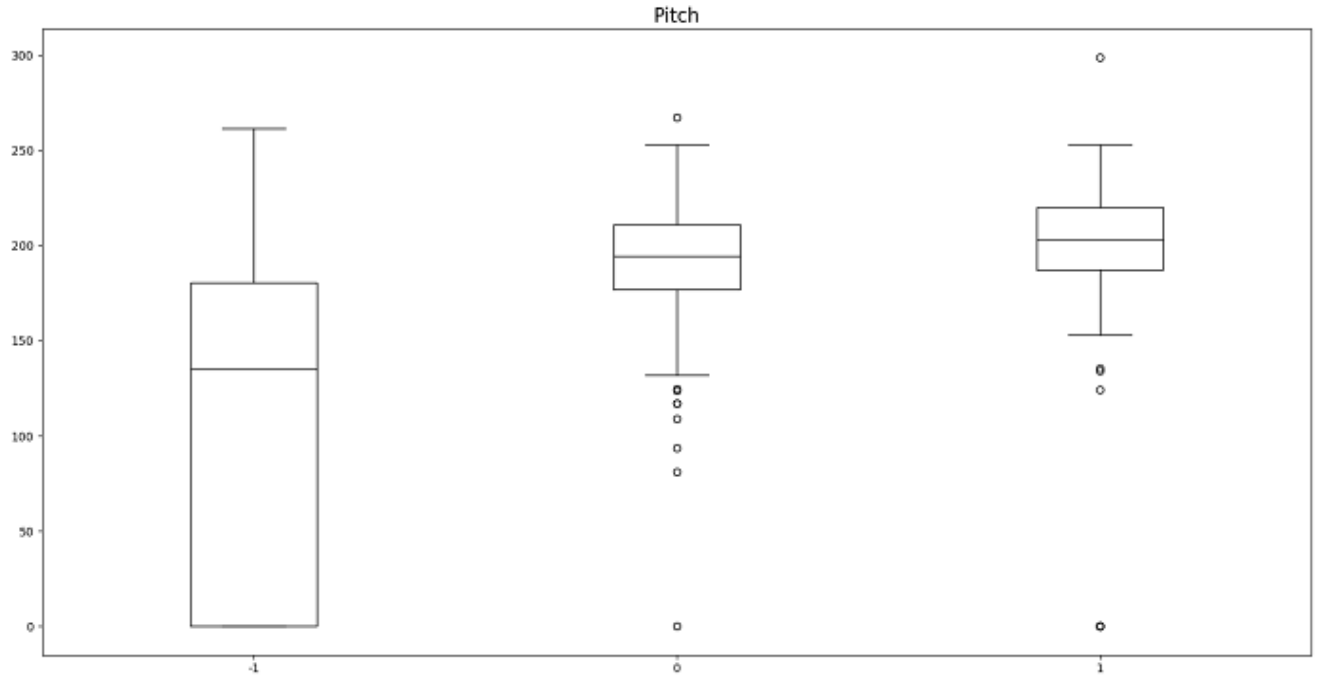

Fig. 7. Distribution of Pitch based on Ground Truth Ratings for Post-Joke Audio

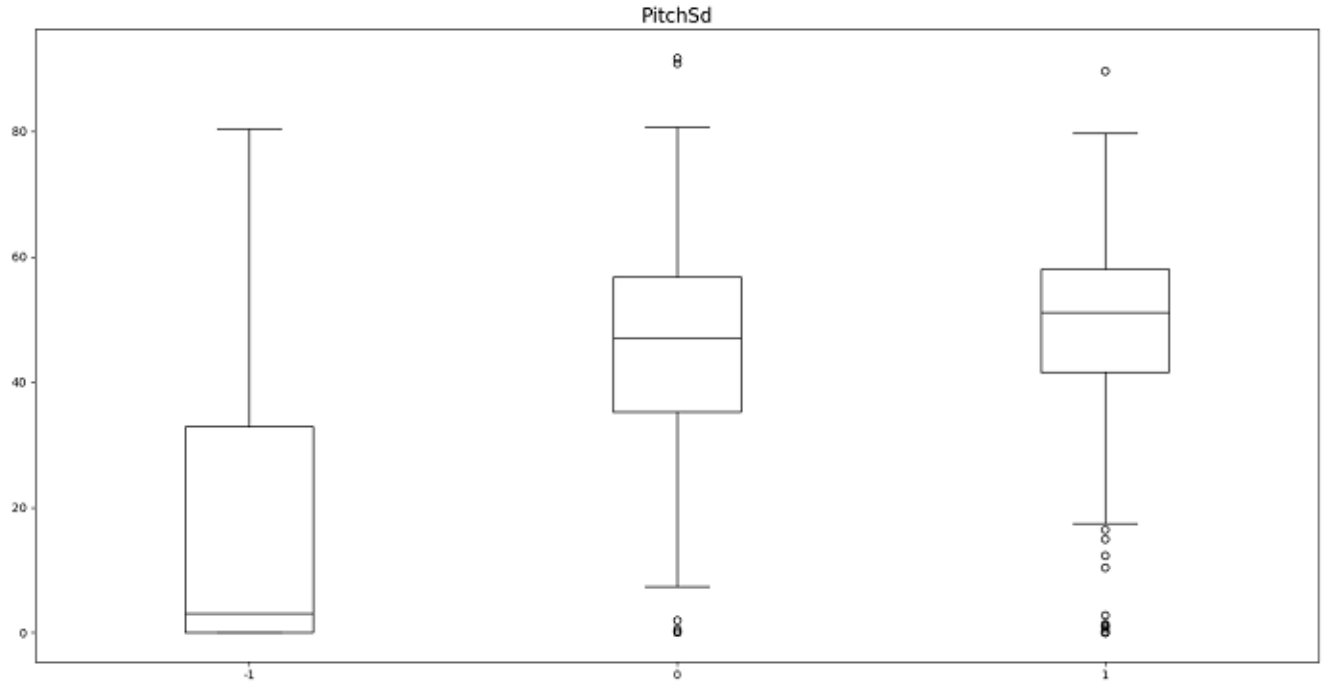

Fig. 8. Distribution of Pitch Standard Deviation based on Ground Truth Ratings for Post-Joke Audio

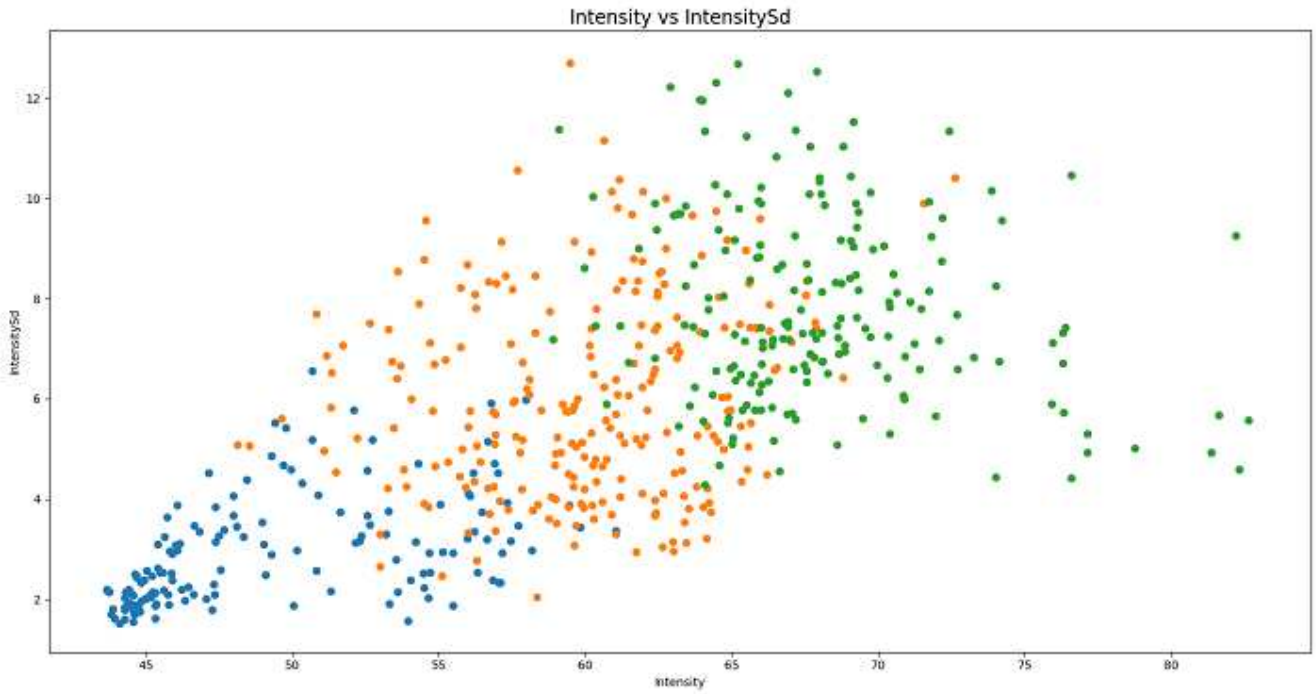

Fig. 9. Ground Truth Ratings Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Audio

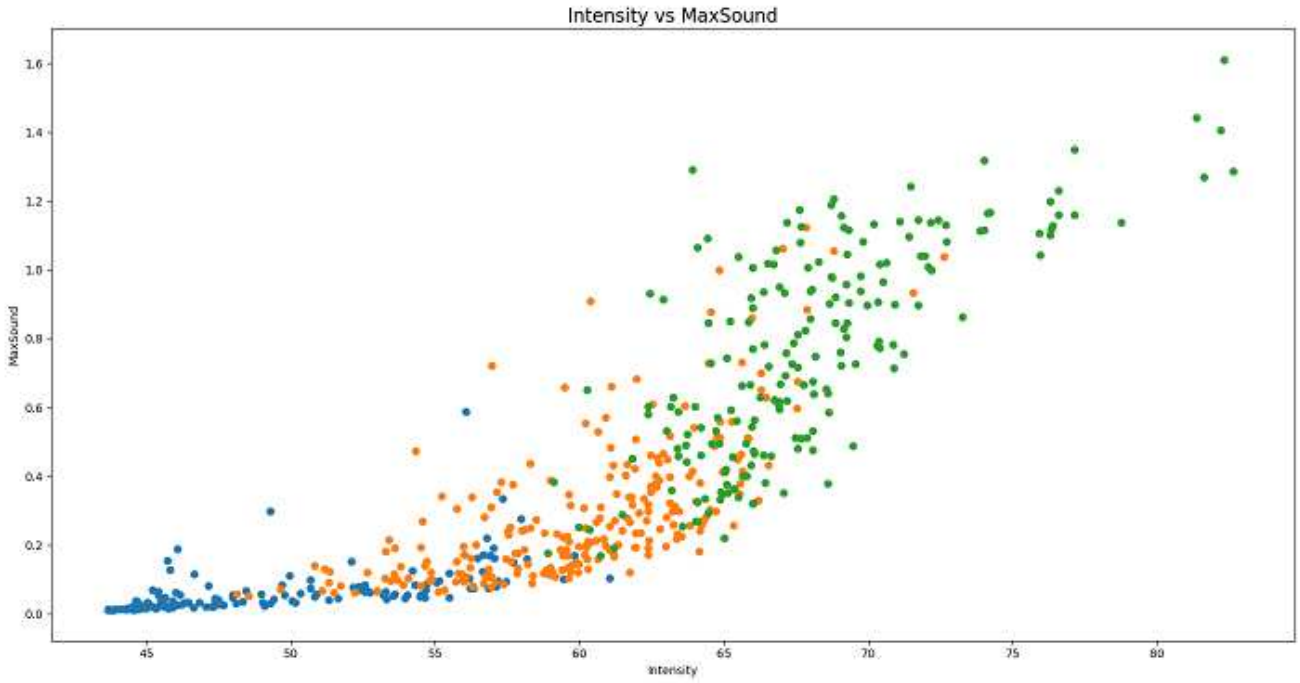

Fig. 10. Ground Truth Ratings Visualized by Intensity vs Max Sound for Post-Joke Audio

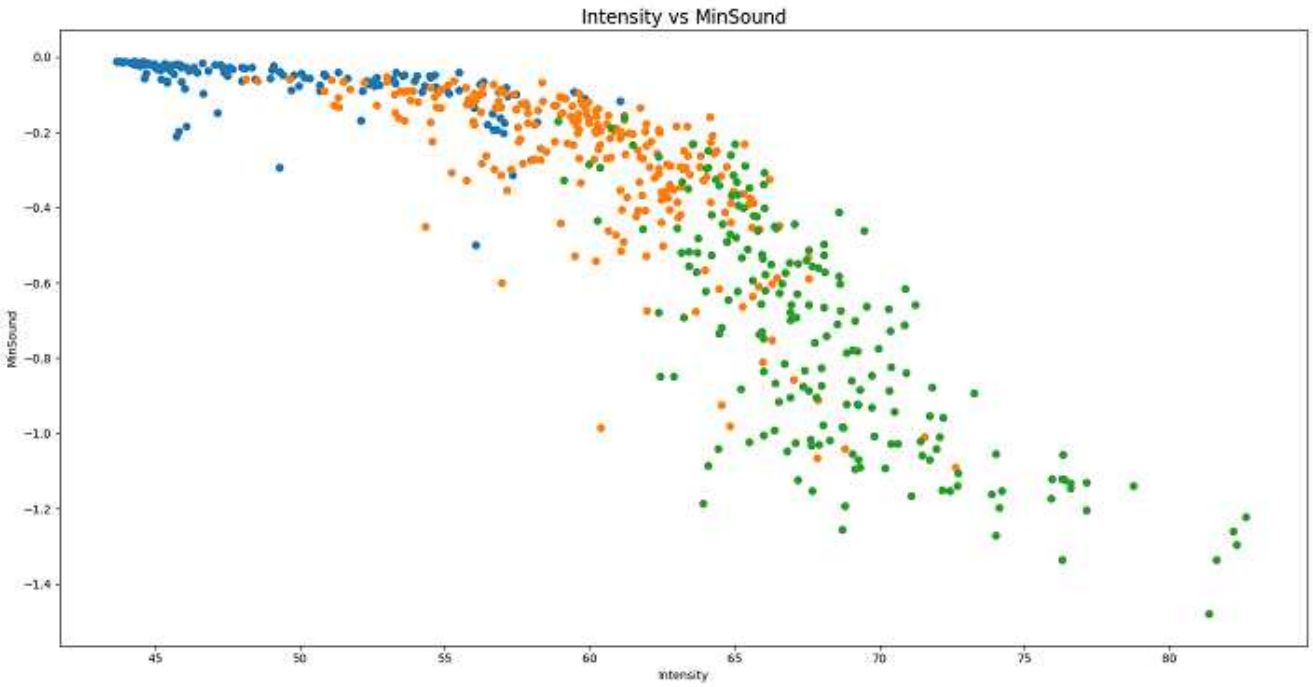

Fig. 11. Ground Truth Ratings Visualized by Intensity vs Min Sound for Post-Joke Audio

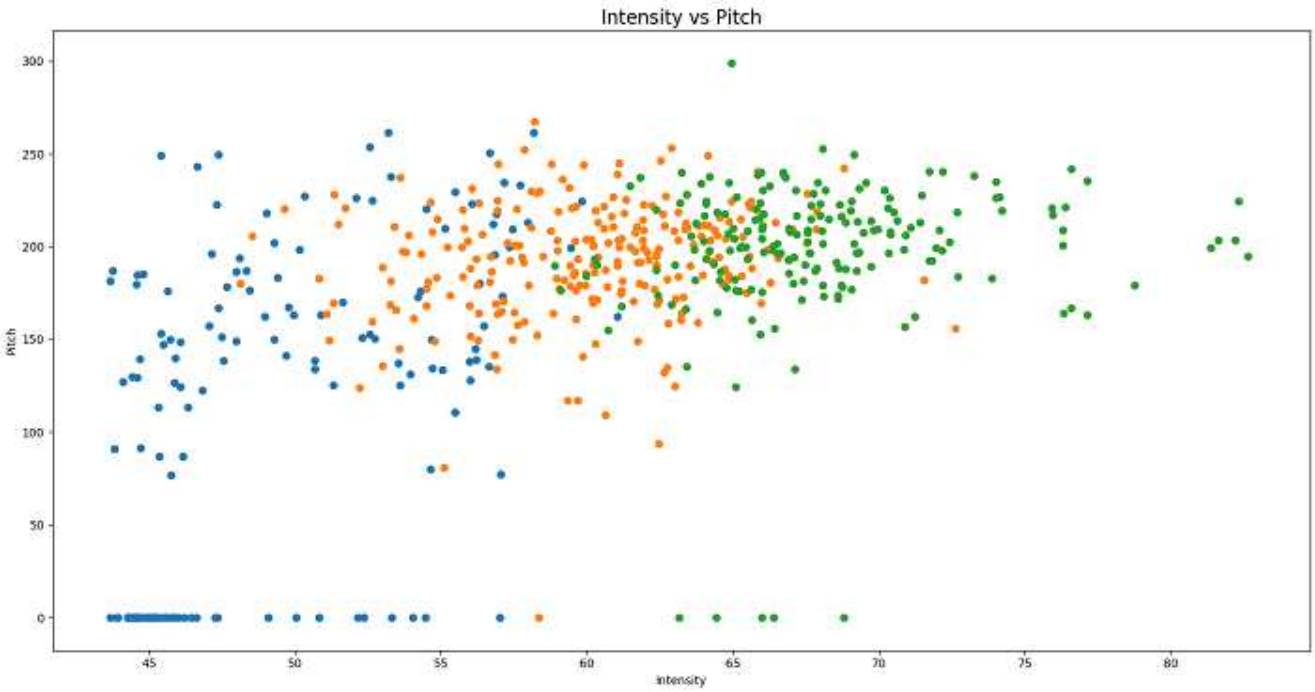

Fig. 12. Ground Truth Ratings Visualized by Intensity vs Pitch for Post-Joke Audio

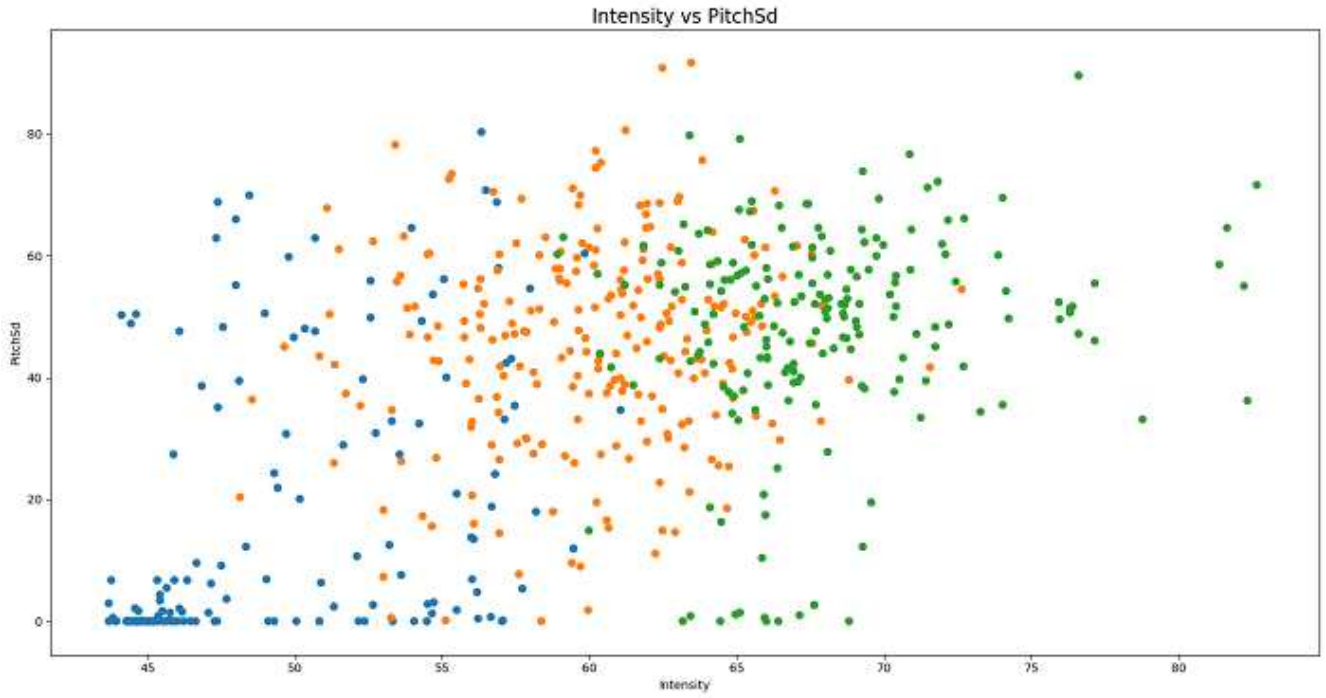

Fig. 13. Ground Truth Ratings Visualized by Intensity vs Pitch Standard Deviation for Post-Joke Audio

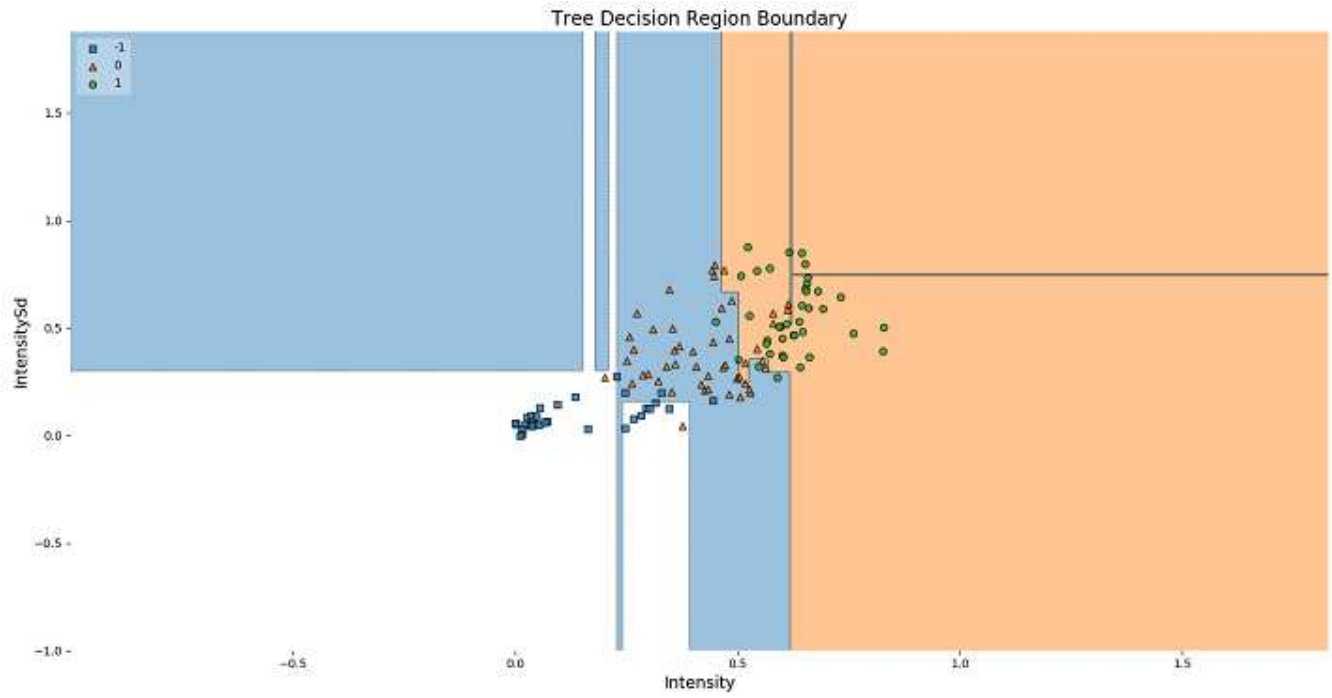

Fig. 14. Decision Boundaries for Decision Tree Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

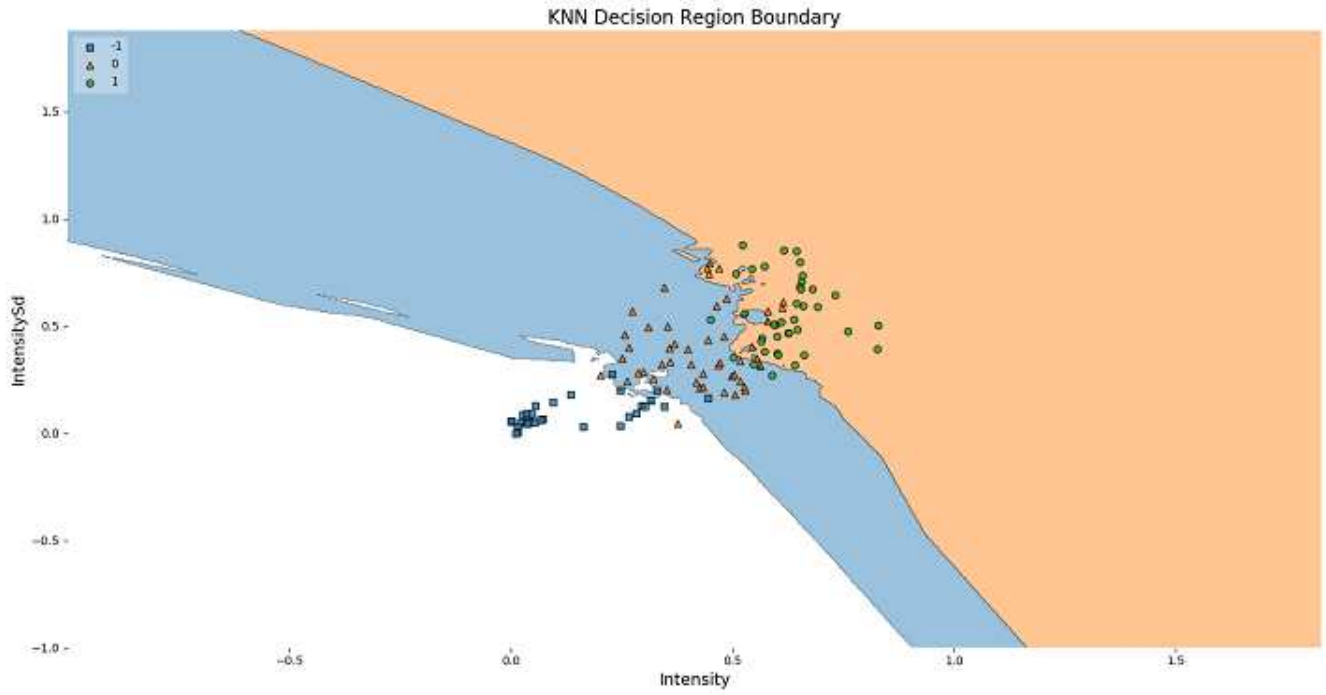

Fig. 15. Decision Boundaries for K Nearest Neighbor Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

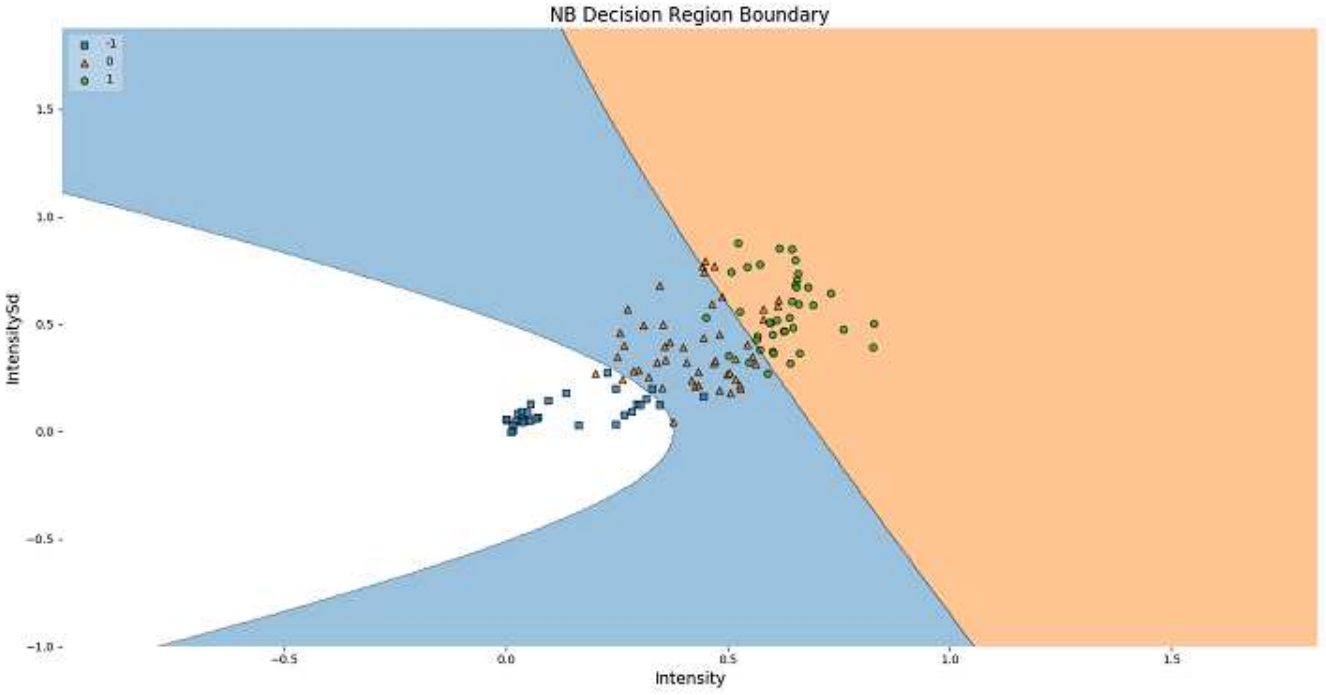

Fig. 16. Decision Boundaries for Naive Bayes Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

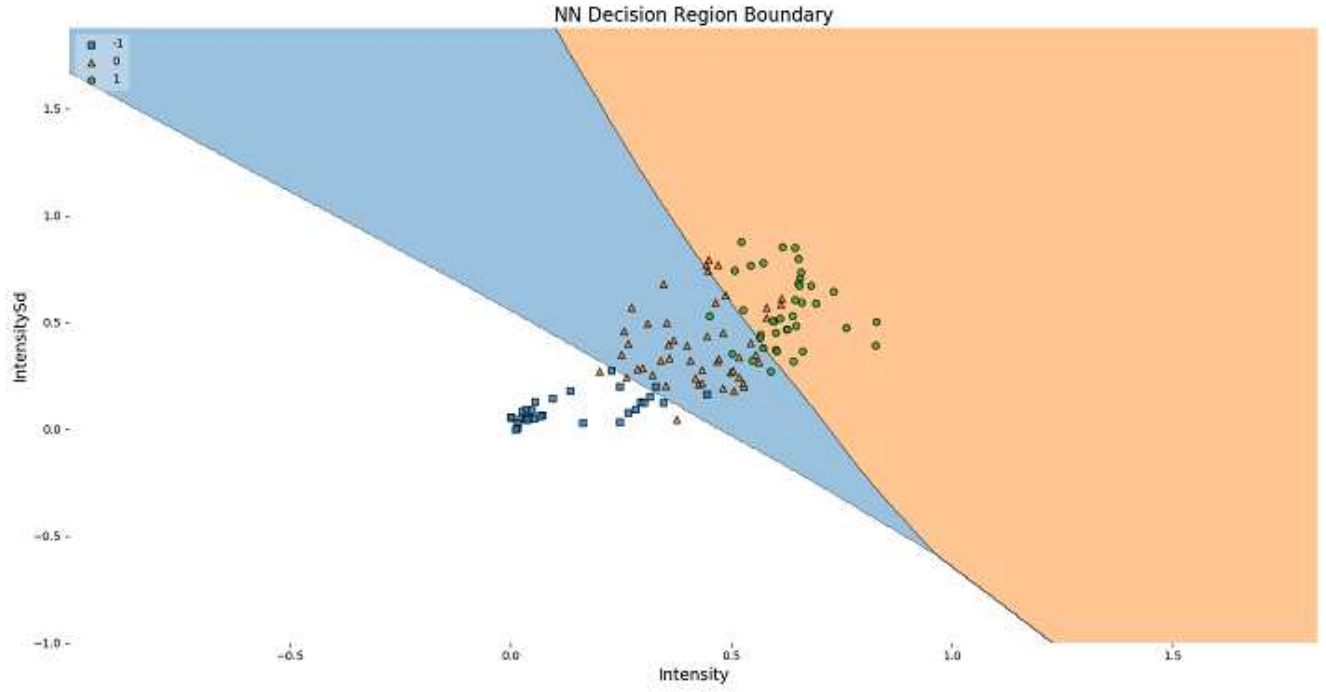

Fig. 17. Decision Boundaries for Neural Network Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

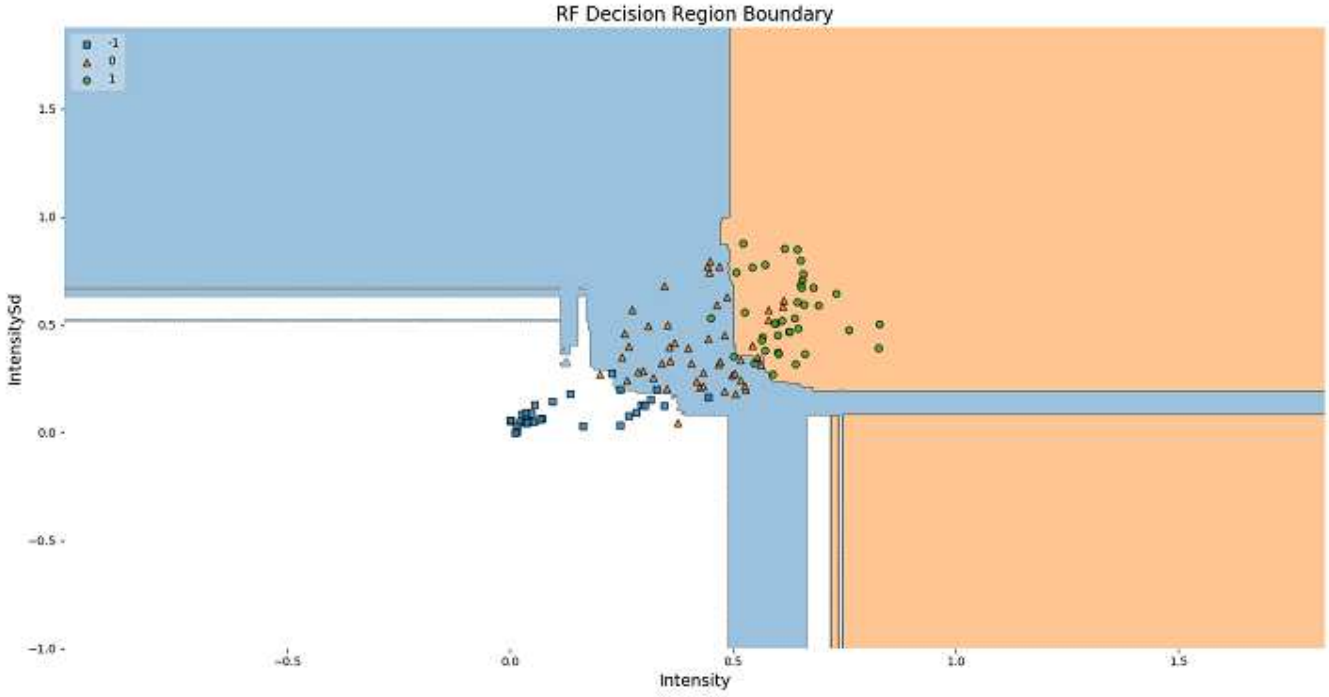

Fig. 18. Decision Boundaries for Random Forest Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

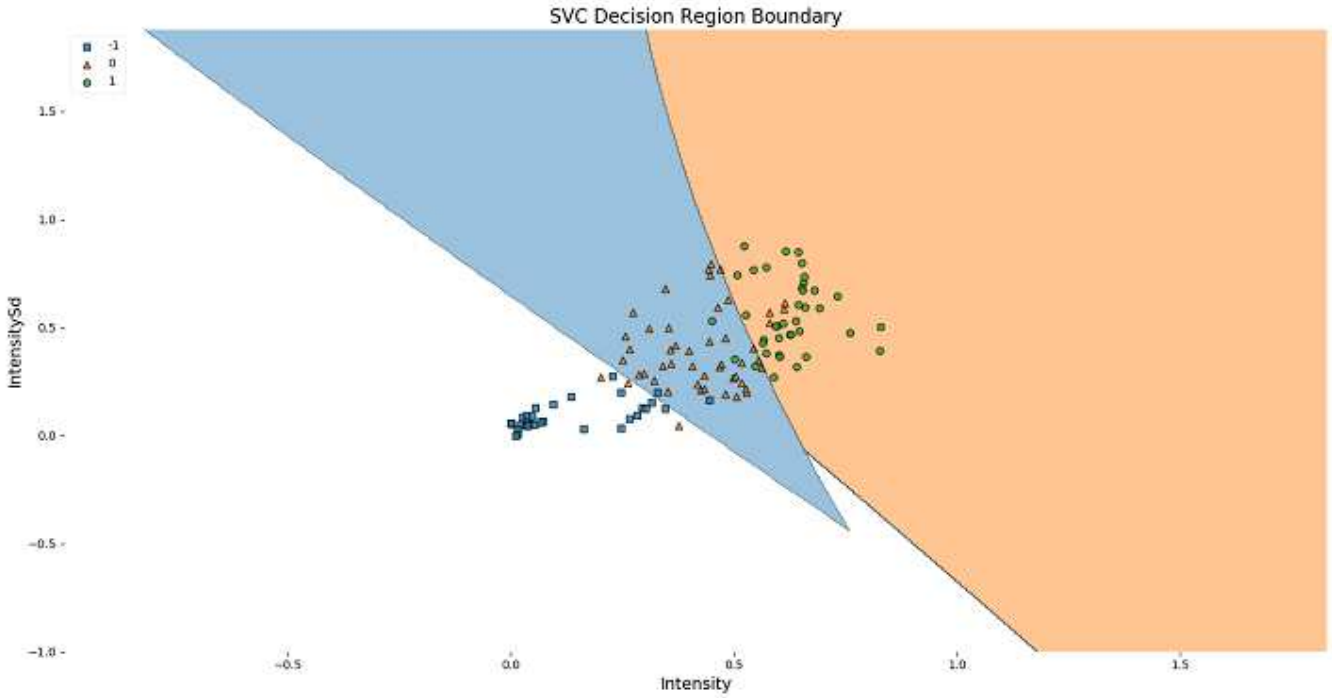

Fig. 19. Decision Boundaries for Support Vector Classifier Visualized by Intensity vs Intensity Standard Deviation for Post-Joke Classifier

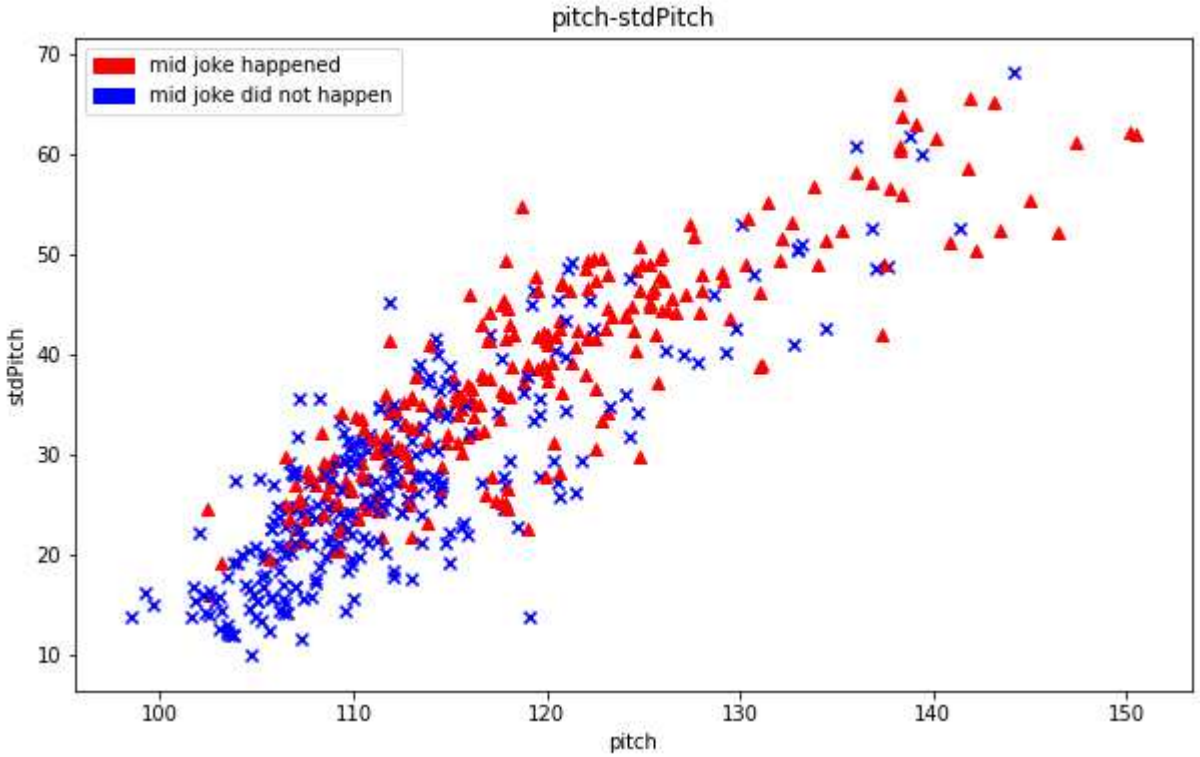

Fig. 20. Positive Correlation of Pitch and Standard Deviation Pitch in Mid-Joke Audio Overall

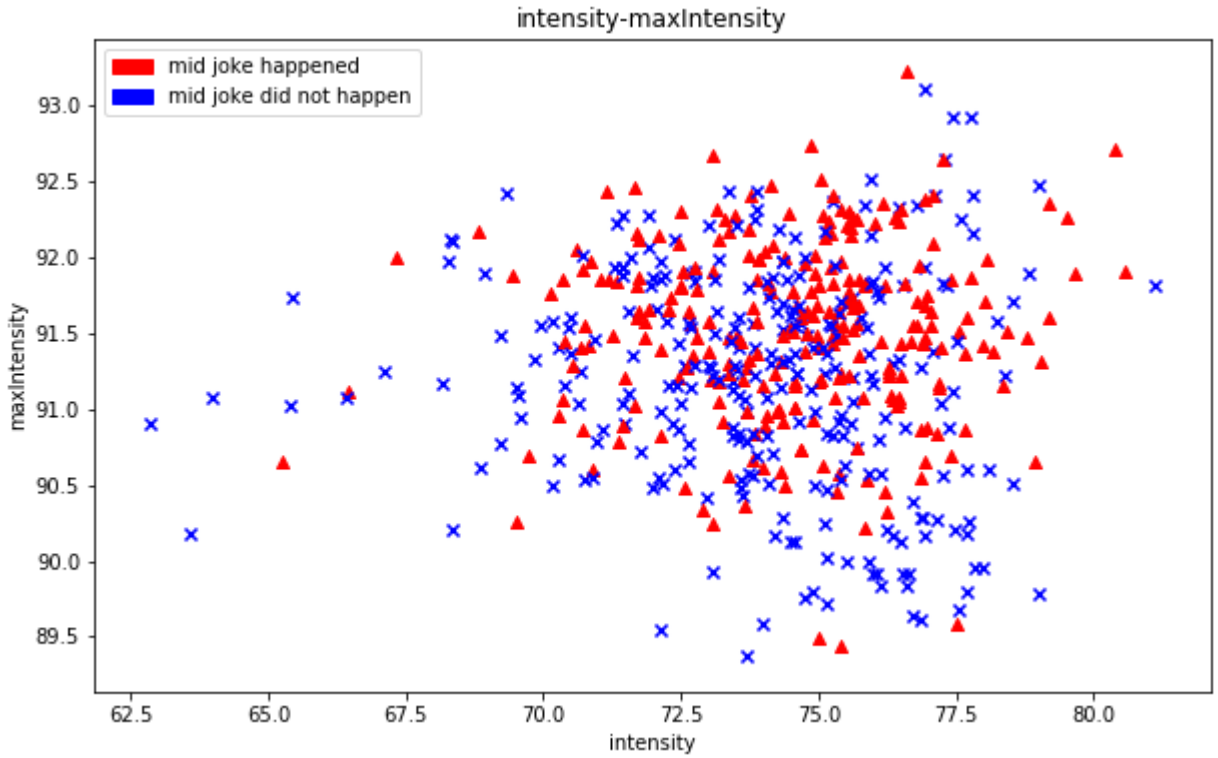

Fig. 21. No Correlation of Intensity and Max Intensity in Mid-Joke Audio Overall

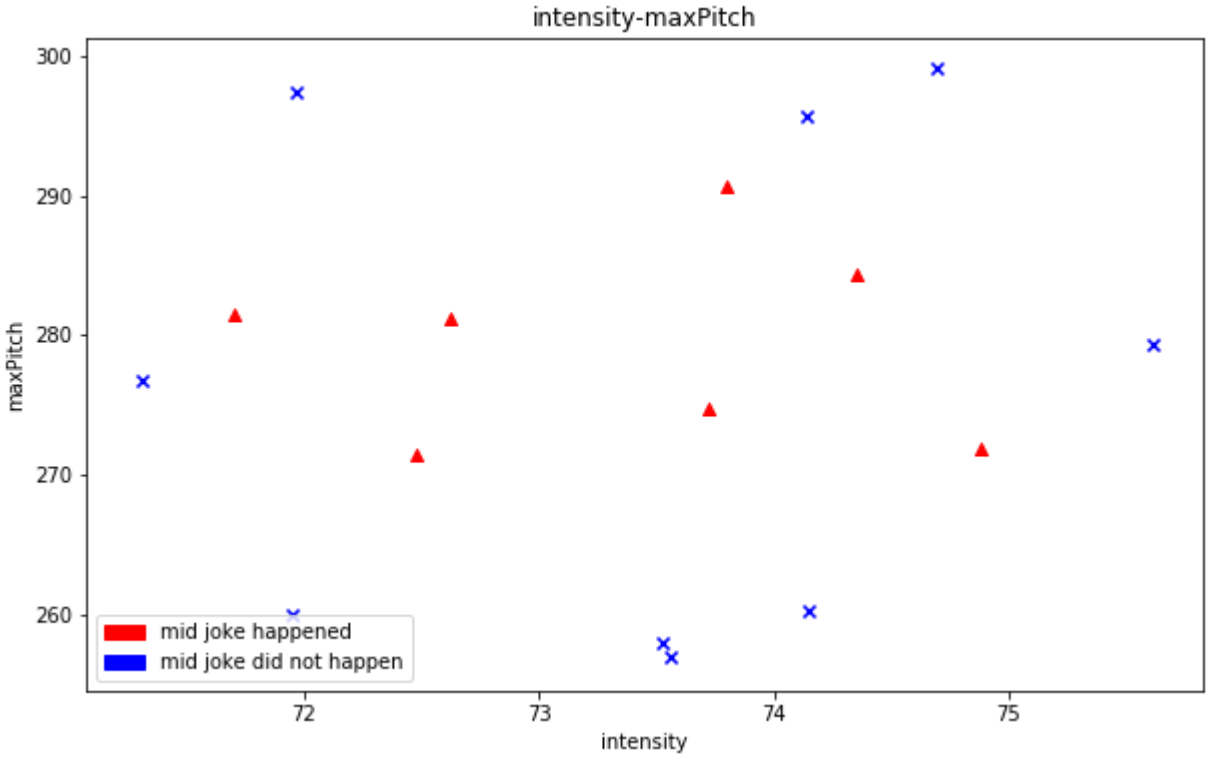

Fig. 22. No Correlation of Intensity and Max Intensity in Mid-Joke Audio Compared Joke-wise

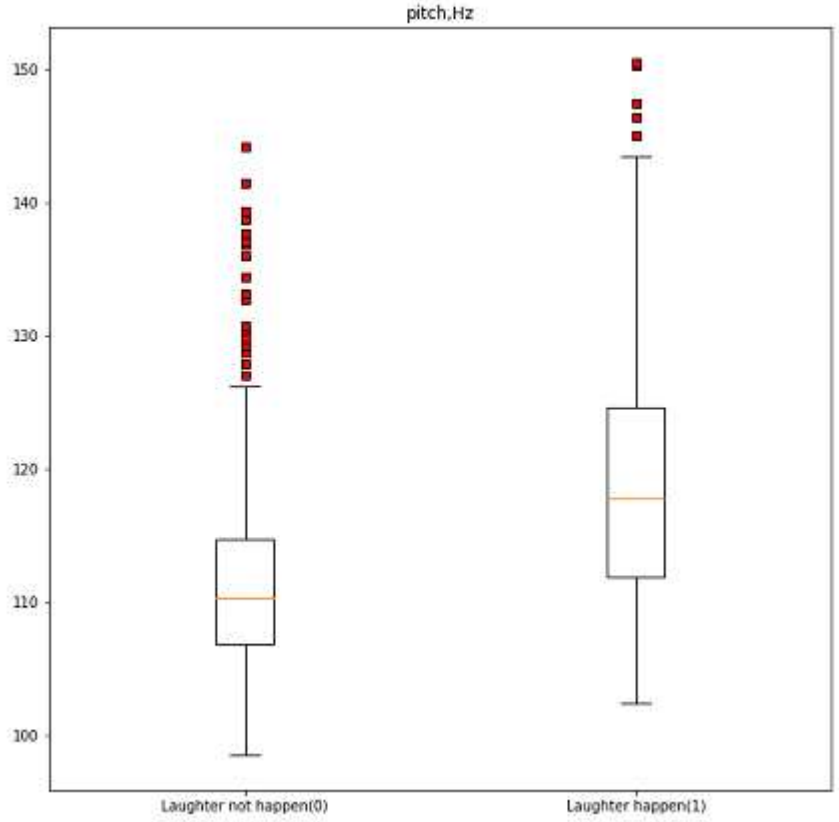

Fig. 23. Distribution of Pitch based on Ground Truth Ratings for Mid-Joke Audio

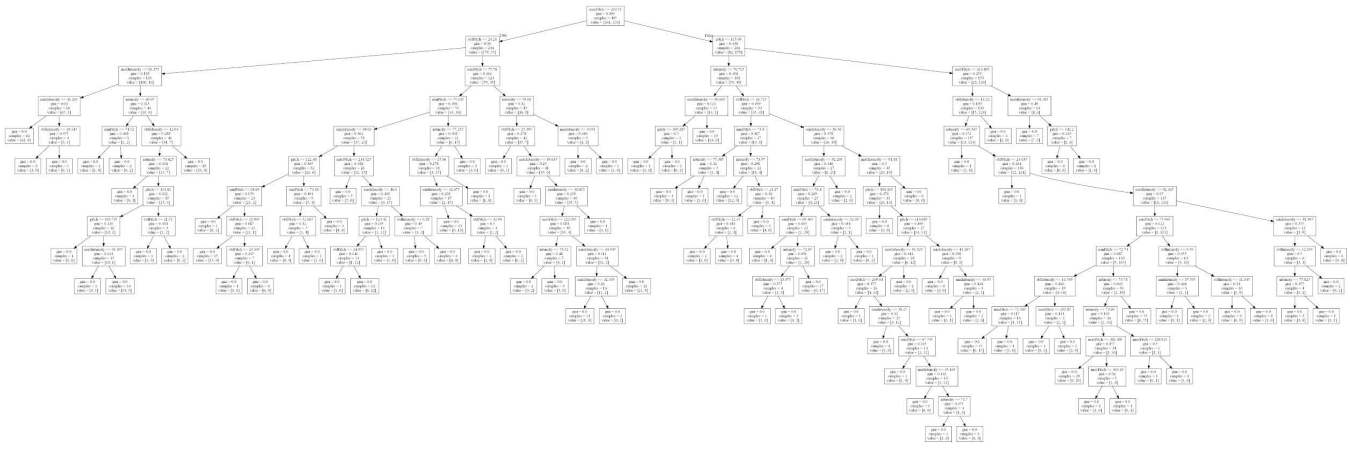

Fig. 24. Mid-Joke Decision Tree

7 Progress Summary

Our work this term was divided into two distinct parts: analysis of the post-joke audio which expanded on previous work by the robot team and introduction of entirely new functionality for mid-joke audio analysis. We held weekly progress meetings with our client to establish a series of sprints for each branch of the work with defined goals to be achieved by the next meeting.

7.1 Post-Joke Sprints

Sprint 1: Jan 8 to Jan 22

- Split performance recordings into joke and pause audio files
- Used annotator to classify post-joke data with individual human ratings
- Created feature extractions from the joke audio files
- Combined our three human ratings into a ground truth rating for each joke

Sprint 2: Jan 22 to Jan 29

- Combined classification data with feature data
- Implemented verificaion of joke order per performance in the classifier program

Sprint 3: Jan 29 to Feb 12

- Implemented basic classifiers (SVM, KNN, Decision Tree, Neural Network)
- Implemented 2-Class and 3-Class
- Calculated human rater accuracy of individual agreement with the consensus rating

Sprint 4: Feb 12 to Feb 26

- Reannotated all jokes by hand after annotator bug was discovered
- Implemented normalization in classifier (MinMax, Standardization, Running Average)
- Performed parameter tuning for SVM classifier
- Implemented new classifiers (Random Forest, Naive Bayes)
- Completed CITI training for potential future studies involving human participants (Stretch Goal)

Sprint 5: Feb 26 to Mar 11

- Data visualization
- Data analysis
- Working toward moving classifiers onto the robot hardware (Stretch Goal)
- Preparation for moving framework for the robot from Choregraphe to the Python SDK (Stretch Goal)
- Began compiling classifier results as foundation for a paper (Stretch Goal)

Sprint 6: Mar 11 to Mar 18

- Completed data visualization
- Preparation for combining mid-joke and post-joke classifiers

7.2 Mid-Joke Sprints

Sprint 1: Jan 8 to Jan 22

- Processed and annotated data from various performances
- Researched strategies for rating mid-joke laughter
- Researched how to visualize data
- Researched various machine learning methods
- Researched various noise suppression methods; attempted naive method of noise subtraction
- Learned how to use Jupyter Notebook; used this to start code for unsupervised clustering

Sprint 2: Jan 22 to Jan 29

- Determineed rating system for mid-joke audio (What constitutes a laugh)
- Decided to use automatic clustering rather than noise suppression
- Decided on ELAN as the audio annotator tool
- Extracted audio data from each joke
- Initially classified extracted data into three groups using K-Means (good, ok, bad)
- Later shifted to extracting data into two groups (laugh, no laugh)

Sprint 3: Jan 29 to Feb 12

- Annotated jokes for mid-joke laughter
- Began initial efforts in data clustering (two groups)
- Analyzed correlation of different audio features
- Created script to match joke to joke ID
- Determined which features to extract from mid-joke audio and found their distributions

Sprint 4: Feb 12 to Feb 26

- Looked into Mel-frequency cepstral coefficients (MFCC) to help with better extracting features and detecting laughter
- Started cross validation algorithm
- Determined if we needed a baseline approach for mid-joke laughter detection (i.e. look for pitch drop by taking the minimum amplitude and subtracting that from the maximum amplitude)
- Exported and parsed the annotated data
- Continued efforts in data clustering
- Completed CITI training for potential future studies involving human participants (Stretch Goal)

Sprint 5: Feb 26 to Mar 11

- Continued research into MFCC
- Looked into intensityRange and pitchRange as potential new features to analyze
- Produced graphs for data visualization
- Looked into Fourier transformations as possible aid to analysis efforts

Sprint 6: Mar 11 to Mar 18

- Added intensityRange and pitchRange as new features
- Review jokes that the program classified wrong
- Categorized errors in the machine learning model into four types: Human annotation error, Post-joke laughter caught in mid-joke audio, Robot laughter caught in audio (only happens with one specific joke), Short laugh heard at beginning of clip
- Discussed adjusting the timing of the audio segments on the robot or re-annotate the audio to address the errors
- Produced a visual of data output from simple implementation of MFCC; began interpreting the results
- Preparation for combining mid-joke and post-joke classifiers