

Bailey Wellen

DS & CS Capstone

Happy Birthday Thesis

Summary

I will develop a Machine Learning model that can predict how enjoyable a performance of “Happy Birthday” will sound to other people. To do this, I will ask many people to play or sing “Happy Birthday” and upload a recording of it. Once I have sufficient data, I will ask volunteers to listen to a few recordings each and rate if each one is enjoyable, tolerable, or terrible. Once I have this labeled data, I will train a machine learning algorithm to predict a given recording’s enjoyability. Since the human ear cannot hear when a note is slightly out of tune, we do not need to check the frequency of the note, but instead can use this trained model to consider if people might enjoy the music. Finally, I will turn this into a web app that will warn an individual when they should consider opting out of singing at a family birthday party.

I plan to write most of this project in Python, and will utilize multiple Python libraries.

Steps

1. Collect LOTS of data
2. Label the data
3. Research intricacies of digital audio
4. Research Machine Learning best practices
5. Develop the Machine Learning Algorithm

6. Build the web app

Step 1: Collect LOTS of data

In order to collect the data, I plan to make a Google Form where volunteers can upload their recordings. I will post the link to this Google Form on all forms of my social media (Facebook, LinkedIn, etc). In addition to this, I will ask members of the clubs and classes that I am in to participate as well. I plan to allow recordings to roll in through the end of October in order to collect as many data points as possible. For best audio quality, I will ask the participants to submit recordings in WAV format if possible. If they submit a MP3 or M4A, I will take it and convert it to a WAV for processing, but this will lose audio quality (eliminated frequencies, decreased sample rate, etc). With the analysis that I am doing, it isn't imperative that the audio be uncompressed, but it still will result in the extracted data being more accurate. [References 1 - 2]

Just how much data is “LOTS”?

I have researched how much training data is considered a reasonable, representative dataset in machine learning. While most people answer “it depends”, one common rule of thumb is to collect a certain percentage (10% seems standard) of the amount of features your data has [Reference 3] . I will only have one feature — the single recording — to predict from, which makes this 10% rule not apply directly to this dataset.

However, this single feature has a lot of details about it that could affect the rating, so I need to decide on an arbitrary cutoff of how much data I will need. There are 25 notes in the song “Happy Birthday”, so I imagine that my data would have at least 25 details, meaning I

would need at least 250 recordings for the training data if those 25 notes were each a feature. I will need to split my data into train and test data, and I plan to use an 80/20 split. This means that I will need approximately 313 ($250 / 0.8 = 312.5$) total recordings for the data input.

Step 2: Label the data

After I have collected the data, I will ask people to rate the data on a Likert scale from 1 - 10, 1 being “my ears! My ears!” 5 being “Hm, I’m not offended”, and 10 being “they must be a professional singer!”. [Reference 4]

I plan to do this by creating a very simple UI that will give a participant 10 recordings and 10 scales. The user will input any number between 1 and 10 for each recording and submit. Once they submit, it will ask the participant if they are willing to rate more data, and if so, it will go back to the main page where they can start again. Note - the recordings will be anonymized before this rating step. I will encourage participants’ honesty with the rating as the singer/performer will never hear the ratings of their own recording.

On the backend, this UI will be built with an R Shiny dashboard, and hosted on an RShiny server. Each time the participant submits, the code will update back to a Google Sheet where I will save the data. In order to do this, I will need to authenticate my google account with R and ensure that I do not max out the Google Sheets API limit.

The UI will look something like this, but with an audio clip replacing the “Rating id” text box and 10 Likert scales instead of just the one. Then, I will ask the participant if they are willing to rate more. If so, it will take them back to Screen 1 again. There is no max to how many recordings a single participant can rate, but each screen will give them exactly 10.

Screen 1:

← → ↻ ⓘ 127.0.0.1:5202

Name

Rating id

What do you rate this recording

1 10

1 2 3 4 5 6 7 8 9 10

Submit

Screen 2:

← → ↻ ⓘ 127.0.0.1:5202

Thank you for participating! We have collected your answers. Unless you want to rate more, you can close this window now.

Want to Rate More Recordings?

I plan to use the “googlesheets4” library and gargle authentication methods in R. [References 5-8] Once I have the details of my authentication and connection to Google Sheets, I will publish this site at shinyapps.io. R Shiny server has a limit of 25 active hours per month on the free account. I do not expect to hit this limit with this project, but if I do, I will pay \$9 for a month of the “starter” account, which allows for 100 active hours per month - a rate I am certain I will not exceed.

The Google Sheet will look like this, where each participant's name will be repeated in multiple rows, the rating_id will be chosen randomly by my script and correlate to the recording that the participant heard, and the rating will reflect what the participant gave that particular recording:

A	B	C
name	rating_id	rating
Bailey	42	8
Bailey	23	1
joe schmoe	246	8
joe schmoe	123	4

I would like to collect at least 3 ratings of each recording to gain confidence in the ratings, which means that each rating_id (filled from the randomized recording selection) should be repeated at least 3 times.

Once I have confirmed the R to Google API connection, I plan to look into Google database options instead of storing this redundant data in a spreadsheet. My choice of Google Sheets over a database for the time being is due to Google Sheets being free, and the remote databases that I have found all charge a fee.

Experimental Design Plan

I will consider “The Secretary Problem” in my experimental design. This mathematical theory advises optimal stopping time for something like choosing a secretary, a spouse, or, in my case, a threshold for what a good recording is. [Reference 9]

The idea of an “optimal stopping problem” rests on the need to gauge the variability and quality of the pool before you can judge how one candidate, person, or song compares. In the case of my thesis, reviewers need to develop a sense of what the range of “Happy Birthday” recordings sound like before they begin assigning numbers. While our problem differs from the “Secretary Problem” in that our reviewers do not need to choose a single recording, I believe that this same theory applies and will help the reviewers to get a benchmark from which they can begin rating the music. In order to allow the reviewers to have an idea of the scope of the recordings, I will allow them to go back and change their ratings after they have heard the later recordings. While this does not exactly follow the advice of the secretary problem, I believe that this strategy will help the reviewers to better estimate how each recording compares to the population.

Step 3: Research intricacies of digital audio

I will spend this time learning how digital audio is saved, researching the best audio processing packages, and ultimately will pick one. This step will occur while I am collecting and labeling data.

While I still have a long way to go to deepen my understanding of digital audio, I have found a Python module that will be useful. I will use the “wave” module [Reference 8], which can return the sample width, frame rate, features of an individual frame, and many more details of a recording. I will encourage the participants to record and upload wav files, but if they do not, I will use the pydub library [Reference 11 - 12] to first change it into a wav file and then proceed with the wave module.

Step 4: Research Machine Learning Best Practices

I will spend this time learning more about how machine learning works and taking note of best practices and advice by experts. I will then research the best machine learning packages and pick the one that best suits this project. From my research so far, it looks like TensorFlow, Keras, PyTorch, and Scikit-learn are some of the best ML libraries, so I will start my research there. [Reference 13] This step will occur while I am collecting and labeling data.

Step 5: Develop the Machine Learning Algorithm

Using the package that I determined to be best in step 4, I will train a machine learning model to predict how good a singing of happy birthday is. Because I will have labeled data, I will use supervised learning. More specifically, I believe that classification is the best ML method to solve this problem [Reference 14] .

Data Analysis and Statistical Methods

Before building my machine learning model, it is necessary to consider the variability between participants who rated data. Some people may be classically trained musicians and will judge recordings critically... others may themselves be tone deaf and think everything sounds great... some may have deep-seeded foul memories of the “Happy Birthday” song and take their anger out on the recordings... others may be the mother of someone who sent in a recording and believe that each recording — which could potentially be their precious child’s — deserves a 10 / 10. Knowing that each individual will rate differently, I must find a way to bring the ratings that my ML model will predict to the same scale.

The best way to consider the variability in the reviewers is to have my model predict the average z-score of a given recording. [Reference 15] To do this, I will standardize the ratings by the person who reviewed it using a simple z-score calculation, $z = \frac{x - \mu_p}{\sigma_p}$, where μ_p is the mean of all ratings given participant, p, and σ_p is the standard deviation of that same set.

Once I have calculated these z scores for each participant, I will have a new set of values correlating to each recording: the three or more z scores. Next I will determine a summary value to assign to each recording - in this case, the average is a reasonable measure. Once I have a list of recordings with a label (the mean of z scores), I am ready to build the model.

Machine Learning Model Details

- Input = many byte objects returned from wave module
- Prediction = average z score of ratings

Step 6: Build the web app

Finally, I will build a web app to display my work. This app will include a tab with information about the project, a link to github, and ways to contact me for any recommendations or bug reports. The main feature will be the ability to record a short audio into the web app and then have it tell you how your recording compares to the 300 + recordings that we used to train the data.

Below are some pictures of what the web app might look like. Full mockup available at

<https://mockingbot.com/app/rggy29p8z1kft2egm6imby2zyqlp#screen=skft2eh4ato2bc4pa>

About this Site	Acknowledgments
-----------------	-----------------

PleaseDontSing.com

Before you belt it out at the next birthday celebration...

Sing Happy Birthday and See how you compare with over 900 ratings of recordings from REAL PEOPLE

[Test My Skills](#)


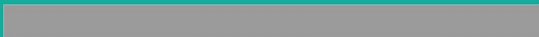

disclaimer - this project is for academic purposes and is not the all-encompassing, all-knowing guide of who can and can't sing. Take it with a grain of salt - don't dismiss your life plans of being an opera performer, but also consider that you may not be as good as you think you are :)

pleasedontsing.com


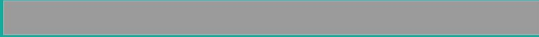

About this Site	Acknowledgments
-----------------	-----------------

Record your performance,
listen back to it if you dare,
and submit when you are ready!

Record

Playback


  

Confident and ready to see how you measure up?

[Submit Now!](#)

pleasedontsing.com

About this Site Acknowledgments



You Rock!


Why are you spending time on this website, shouldn't you be on stage?

Go again Contribute to this Project by Rating

☒ It is okay to use my recording to further improve your algorithm

pleasedontsing.com

About this Site Acknowledgments

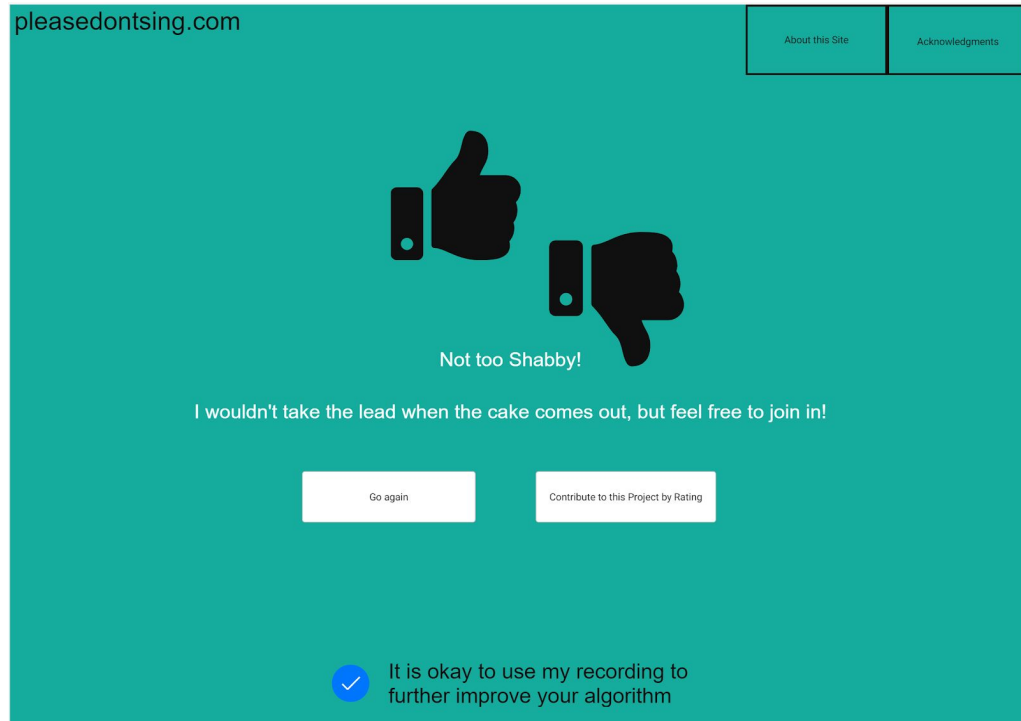


Yikes....

You know that just mouthing the words is okay too, right?

Go again Contribute to this Project by Rating

☒ It is okay to use my recording to further improve the algorithm



Timeline

Now - End of October:

- Collect recordings
- Develop UI for participants to rate recordings
- Research digital audio
- Research ML best practices

End of October - Mid November:

- Deploy UI - solicit participants to rate recordings
- Solidify packages that I will be using for ML

Mid November - End of Fall Semester:

- Develop original ML model, optimize and evaluate the accuracy of it

 SEMESTER BREAK

Beginning of J Term - end of Spring Semester:

- Continue improving ML model
- Build and deploy web app
- Write a summary of the project and what I learned

Summary of Commitments

<u>Must Haves</u>	<u>Would Like to Haves</u>
Collect Recordings Data	Optimize ML model to an accuracy of between 75 - 85% [References 16 - 17]
Collect Ratings Data	Build a User Interface where they can input their own recording and see the predicted rating
Perform Statistical Analysis to Standardize Reviews	Publish the UI to a web app at a url like pleasedontsing.com [Reference 18]
Be able to extract features of audio into a digestible format	Allow the User Input to be added to the training data and users to rate other recordings. Then, set the code to continually refine its model (Ambitious)
Develop a Machine Learning Model with training data	
Improve the ML Model to obtain at least 65% accuracy on test data	

Concerns and Remaining Questions

- I am concerned that the people most likely to volunteer for this study have confidence in their singing ability. If my hunch is correct, this means that the dataset will be skewed

right, with more pleasant recordings than unpleasant. This belief rests on the assumption that they are capable of accurately assessing their singing voices, so my fears may be irrelevant due to humans' tendency towards arrogance. :)

- Another concern about willingness to volunteer for recordings- I am worried that people who know that other people will be listening to and rating their singing will feel bashful and unwilling to submit recordings. If this happens, I will need to find a way to document and clarify how the rating process will work.
- A question I have about the web app - I wonder if there is a way to utilize the users' data on the web app in order to continue improving my ML algorithm? I would like to find a way to save those inputs somewhere and then encourage people who visit the page to contribute to the research by labeling some more data.

References

- 1) “Audio Formats and File Types”

<https://soundbridge.io/audio-formats-file-types/>

- 2) “Compressed vs. Uncompressed”

<https://exchange.prx.org/help/posting-audio/compressed-versus-uncompressed#:~:text=Digital%20audio%20can%20be%20split,the%20audio%20that%20was%20recorded.&text=Compressed%20formats%20are%20those%20that,the%20file%20size%20relatively%20small.>

- 3) “How Much Training Data is Required for Machine Learning?” (Brownlee, 2017)

<https://machinelearningmastery.com/much-training-data-required-machine-learning/>

- 4) “Likert Scale” (Bhandari 2020)

<https://www.scribbr.com/methodology/likert-scale/#:~:text=To%20collect%20data%2C%20you%20present,data%20can%20be%20analyzed%20quantitatively.>

- 5) “Persistent Data Storage in Shiny apps” (Attali, 2017)

<https://shiny.rstudio.com/articles/persistent-data-storage.html>

- 6) Google Sheets - Usage Limits

<https://developers.google.com/sheets/api/limits>

- 7) GoogleSheets4 Authorization Documentation

https://googlesheets4.tidyverse.org/reference/sheets_auth.html

- 8) Gargle Non-Interactive Authentication

<https://gargle.r-lib.org/articles/non-interactive-auth.html>

- 9) “The Secretary Problem: An algorithm for deciding who to marry, and other tough choices” (Parker, 2014)

<https://slate.com/technology/2014/12/the-secretary-problem-use-this-algorithm-to-determine-exactly-how-many-people-you-should-assess-before-making-a-new-hire-or-choosing-a-life-partner.html>

- 10) Wave Documentation

<https://docs.python.org/3/library/wave.html>

- 11) Pydub Documentation

<http://pydub.com/>

- 12) “Convert MP3 to Wav”

<https://pythonbasics.org/convert-mp3-to-wav/>

- 13) “Best Python Libraries for Machine Learning and Deep Learning” (Claire D. Costa, 2020)

<https://towardsdatascience.com/best-python-libraries-for-machine-learning-and-deep-learning-b0bd40c7e8c>

- 14) “4 Types of Classification Tasks in Machine Learning” (Brownlee, 2020)

<https://machinelearningmastery.com/types-of-classification-in-machine-learning/#:~:text=In%20machine%20learning%2C%20classification%20refers,one%20of%20the%20known%20characters.>

- 15) Conversation with Professor Erlan Wheeler, 9/22/20 about necessity to normalize data

- 16) “What is a good classification accuracy in data mining?” (Saitta, 2010)

<http://www.dataminingblog.com/what-is-a-good-classification-accuracy-in-data-mining/>

- 17) “How to Know if Your Machine Learning Model has Good Performance” (Brownlee, 2018)

<https://machinelearningmastery.com/how-to-know-if-your-machine-learning-model-has-good-performance/#:~:text=If%20you%20are%20working%20on,modeling%20problems%20have%20prediction%20error.>

- 18) Email from Professor Perry Kivolowitz, 9/27/20, with domain name idea