Cutting Through the Clutter: A Study on Audio Pre-Processing and TB Detection

Ashwin Salampuria

Department of Computer Science, Ashoka University

CS-IS-3032-1-Deployable Acoustic ML/DL for Public Health

Professor Rintu Kutum

May 19, 2023

Abstract

In this paper, I discuss and review a couple of data pre-processing measures, the performance of which are essential on input audios for them to be made usable to a machine learning model directed at disease classification. These include noise reduction and data augmentation techniques. Then, I move on to Mel-Frequency Cepstral Coefficients as a method for classifying the noise reduced and augmented dataset for Tuberculosis diagnosis using Cough Sounds. Finally, a Gradient Descent algorithm is implemented from scratch and the paper concludes through a discussion on a few strategies to escape saddle points which can occur in non-convex problems. Gradient Descent being one of the most common optimization methods used on cost functions helps us unpack the differential calculus needed for another most commonly implemented ML model- linear regression.

Introduction

This paper starts by focusing on Per Channel Energy Optimization or PCEN for Noise Reduction. There is a method available in Python's Librosa Library which we make use of for this purpose. Librosa is used extensively also for generating spectrograms throughout this report. Further, we look at the Audiomentations library for understanding the concept of data augmentation. The functions in the Compose Class of the library are used to explain the kind of modifications usually made to the input wav file. For this I also reference a paper on "fall event detection" where the augmentation decisions made help build the logic for how not every augmentation technique could be used on every kind of audio and that we need specific functions for the TB positive and negative audios which preserve their essence and yet enhance the size of the dataset.

As far as MFCC goes, the focus is not just on the algorithm but also on trying to integrate it with some key ideas related to frequency of the sound generated by our vocal cords and that are used for speaker recognition. These include ideas related to glottal pulse and vocal tract length response. Finally, the idea of partial differentiation is used to explain the calculus behind minimization techniques. We also suggest the idea of a second order derivative for the problem posed by inflection points, also called saddle points, that specifically occur in non-convex problems.

The observations made through the semester using weekly Google Colab reports are incorporated into this final paper.

Chapter 1 : Noise Reduction Using Per Channel Energy Normalisation

The main idea behind PCEN is the “Gaussianizing” of values present in the spectrogram of an audio that’s been generated using Mel-Frequencies (Ick & McFee, 2021). To explain in even simpler terms, it is integral we first understand what a spectrogram is. Representing an audio using an image such that along one axis we measure the time and on the other the frequency at different points in time while also capturing the amplitude through either a third axis or by using different colours for the markings of frequencies gives us a visual cue to process the audio we have (2019). This is what a spectrogram does . The 3rd Chapter of this report explains Mel-Frequency and its usage in MFCCs. In this chapter, we categorically focus on the workings and usage of PCEN.

When we speak of audios, it is integral to acknowledge that real world case studies do not present us with clean audios as the device used for recording does not only store the required sound. Everything happening in the surroundings of our main audio gets superimposed on it as well. The timbre of the various audios that usually get recorded at once is different in that background noise tends to be “Brownian” in “urban environments” and this is where the approach to “Gaussianize” an audio comes in (Ick & McFee, 2021).

There is simultaneous augmentation and attenuation that happens to different sections of the audio through PCEN (Lostanlen et al., 2018). We do not just subdue the variance that is present in the sounds of the foreground we also perform a reduction in the background noise. While the attenuation is similar to what happens when we logarithmically express a spectrogram with mel frequency distribution, such is not the case with the augmentation (Lostanlen et al., 2018). Here the augmentation refers to the amplification and not the usual data augmentation we refer to when we use the term. This is owing to the fact that both these

procedures involve something called “Dynamic Range Compression" which essentially is to say that loud sounds are made less loud while the quiet ones are made louder (Lostanlen et al., 2018). However, in PCEN we have the added benefit of “Adaptive Gain Control" in PCEN (Lostanlen et al., 2018). The latter helps in identifying stationary noise present in the audio and this identification further supports the suppression of the sound we want to remove from our main audio (Lostanlen et al., 2018) .

Figure 1

Equation governing PCEN

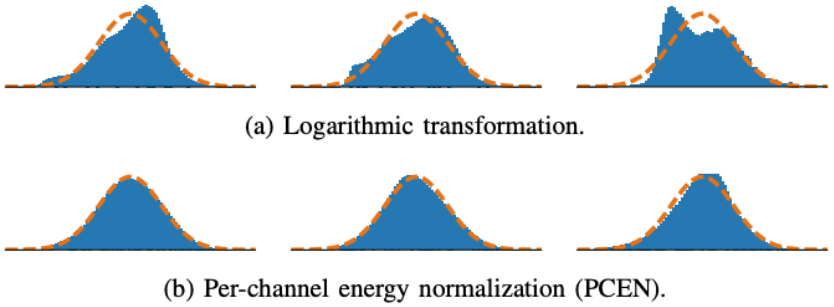
$$PCEN(t,f) = \left(\frac{E(t,f)}{(\epsilon + M(t,f))^{\alpha}} + \delta \right)^{r} - \delta^{r},$$

Note. This image was extracted from 日々、学ぶ. (2022, November 12). 【*pcen*】対数*log-mel* に代わる特徴量*pcen*【*librosa*】. 日々、学ぶ. https://take-tech-engineer.com/librosa-pcen/ .

Figure 1 shows the main equation governing the operation (Lostanlen et al., 2018; 2022). While it does involve a lot of complex mathematical terms behind it as is evident from the equation, it is easier to get the intuition when we take a look at Figure 2 to understand how a Gaussian distribution looks compared to one generated by the usual noisy log data as in the picture above (Lostanlen et al., 2018; 2022). A certain maximisation in the visual difference and contrast that is present between the background noise and the main event noise happens when we analyse the audio post PCEN using covariance matrices (Lostanlen et al., 2018) .

Figure 2

Difference between Log transform and PCEN transform



(a) Logarithmic transformation.

(b) Per-channel energy normalization (PCEN).

Note. This image was extracted from "Per-Channel Energy Normalization: Why and How" by Lostanlen et al., published in 2018.

One key assumption underlying PCEN is that the changes in amplitude as measured by the "Amplitude Modulations", are much quicker for the sound in the foreground of our audio data for every single band that the audio is broken up into than that in the background (Lostanlen et al., 2018) . The same proportionality holds for when the measurement is of the Frequency Modulations and both of these together constitute the "temporal integration" factor (Lostanlen et al., 2018) . In the article by , this is used to explain how the birds' chirps are amplified while the noise reduced and for us as the goal was identification of TB using cough sounds, the usage of this algorithm would mean it is the sound of cough which gets amplified as the AM of that part of the audio which has the cough would have higher AM and FM values compared to the noise which in our case might fit well with the assumed urban environmental noise which as per the article by (Lostanlen et al., 2018) . Thus lower AM and FM valued noise get further attenuated out of the audio and this reflects very evidently when visualised using a spectrogram of the PCEN modified audio just as we wanted (Lostanlen et al., 2018) .

Figure 3

Colab Report on Noise Reduction performed using PCEN

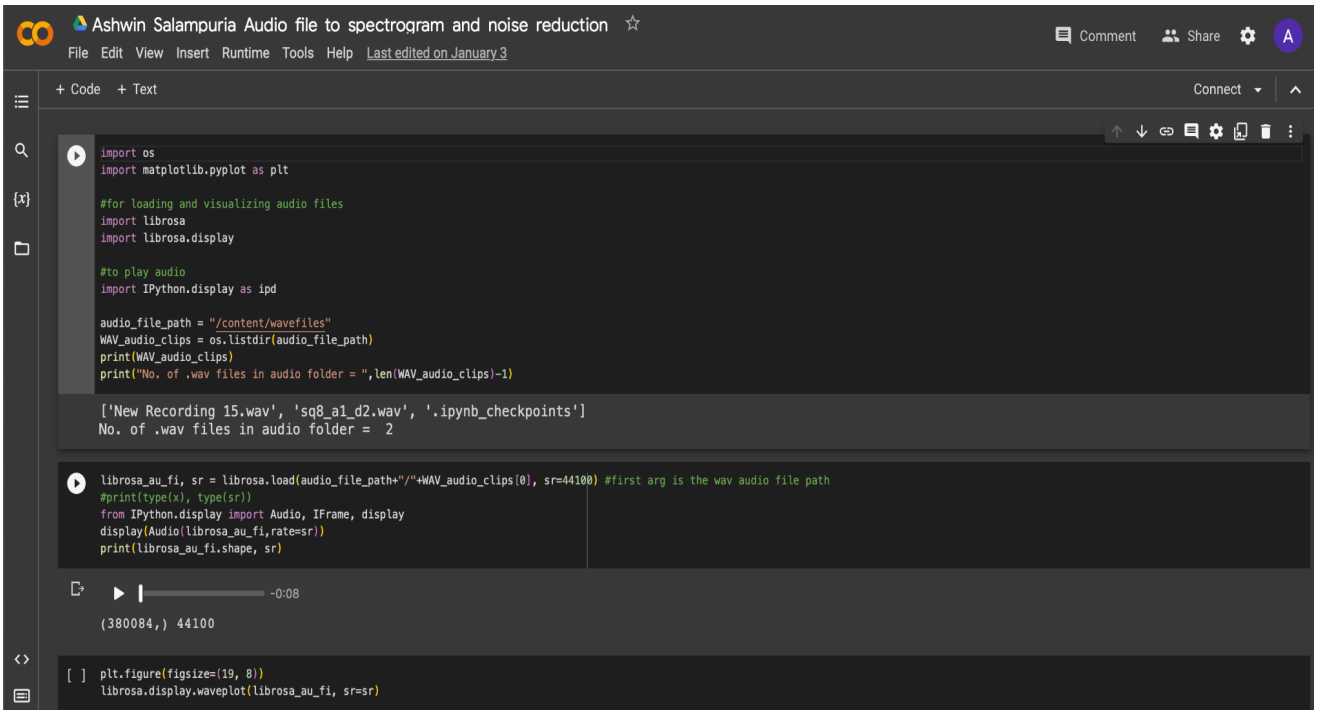


Figure 4

Colab Report on Noise Reduction performed using PCEN

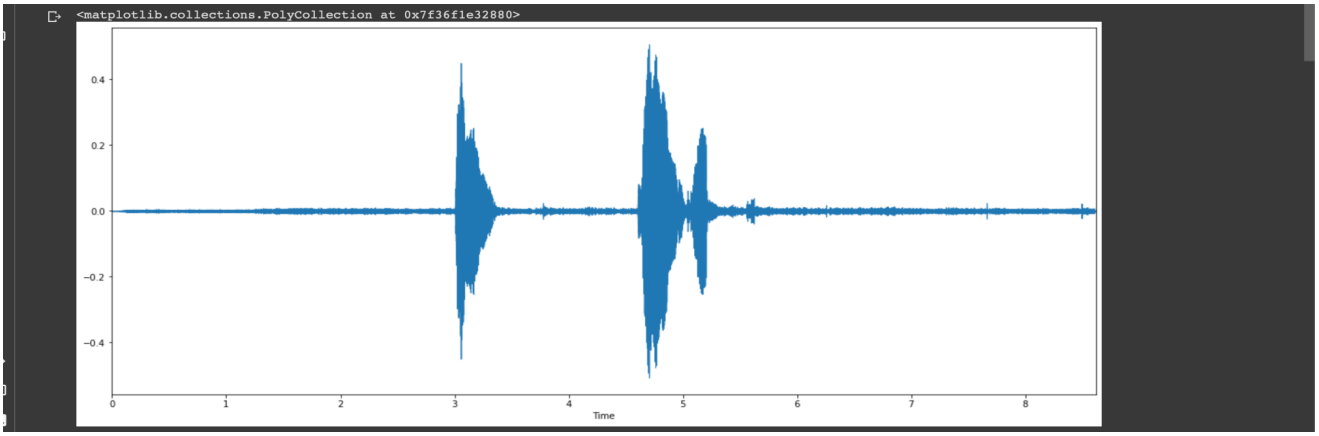
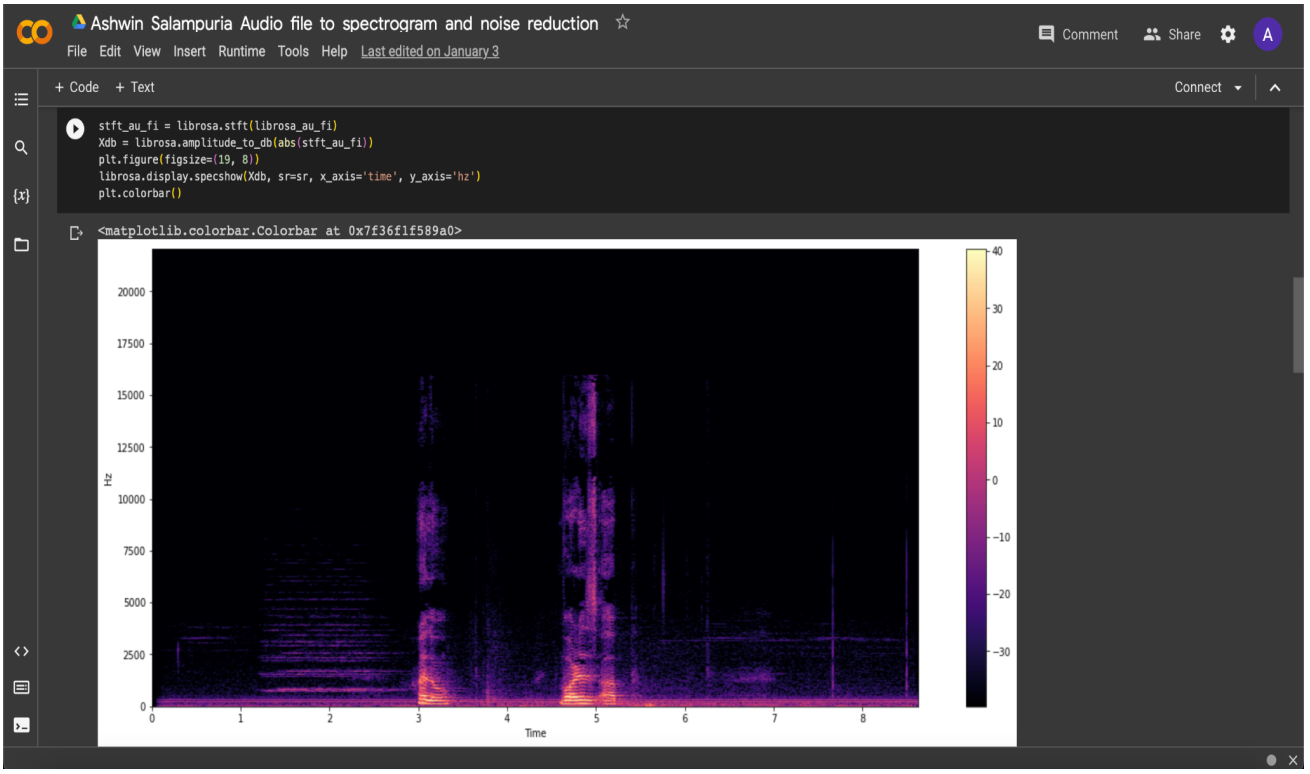


Figure 4 shows the initial waveplot we get of the input audio.

Figure 5

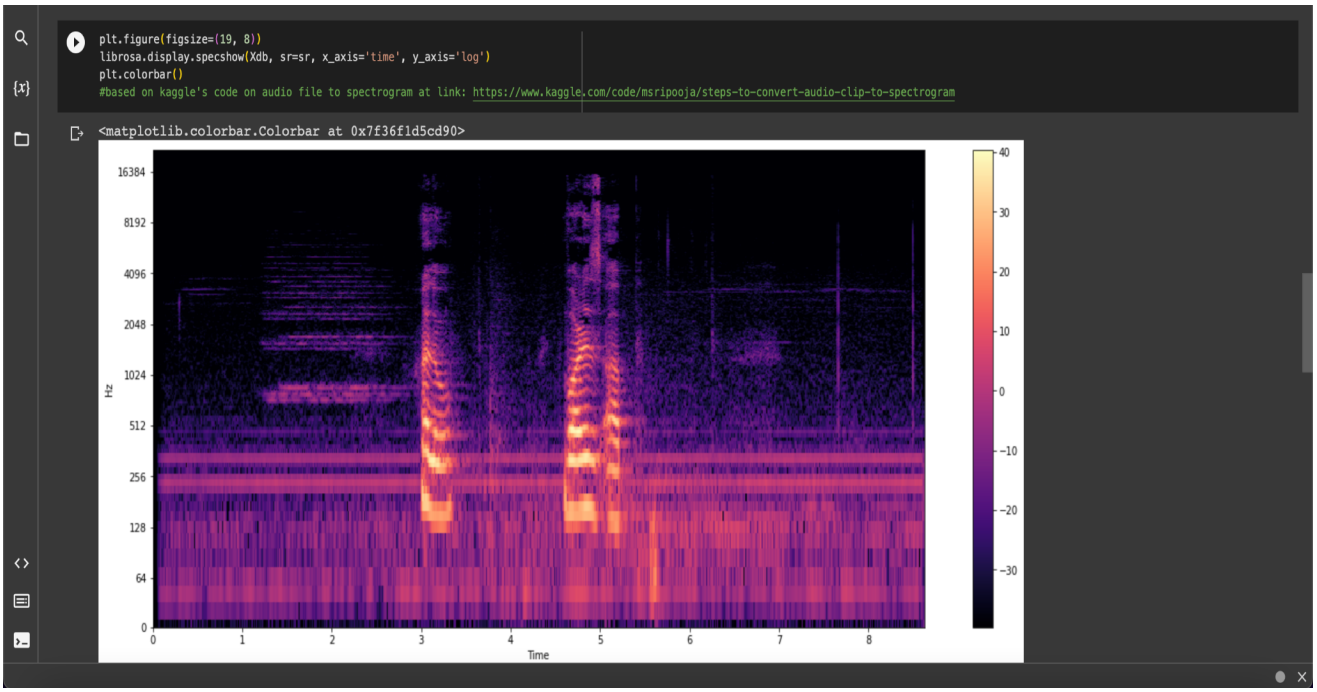
Colab Report on Noise Reduction performed using PCEN



Then, we obtain the frequency and log scaled version of our amplitude spectrogram.

Figure 6

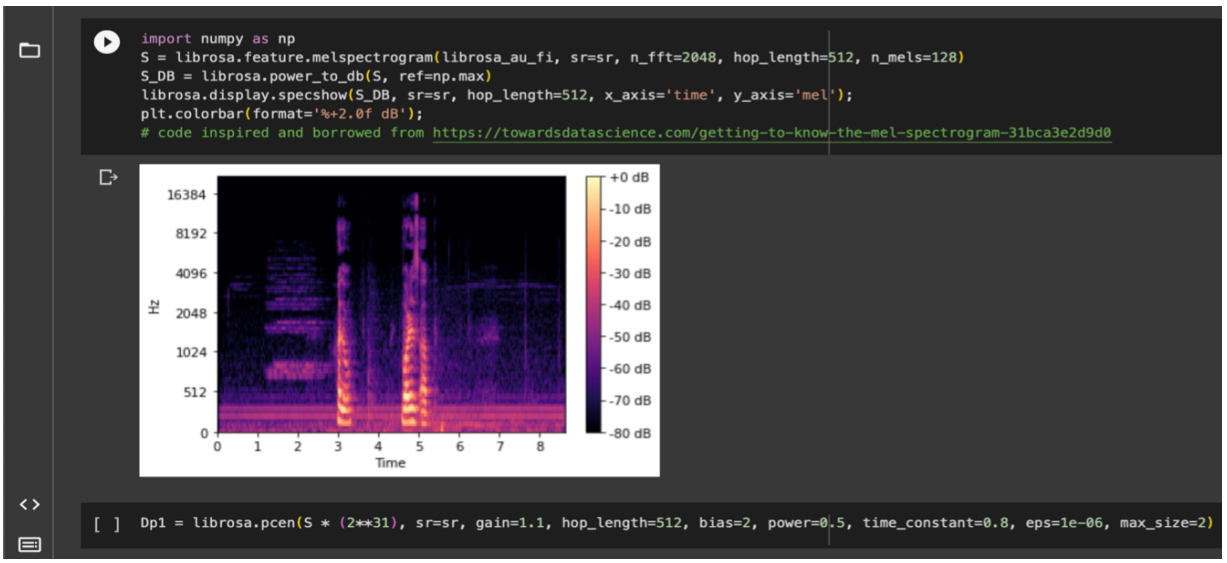
Colab Report on Noise Reduction performed using PCEN



Then we perform log-scaling of frequencies and obtain a new spectrogram. Finally, we obtain a mel-frequency spectrogram.

Figure 7:

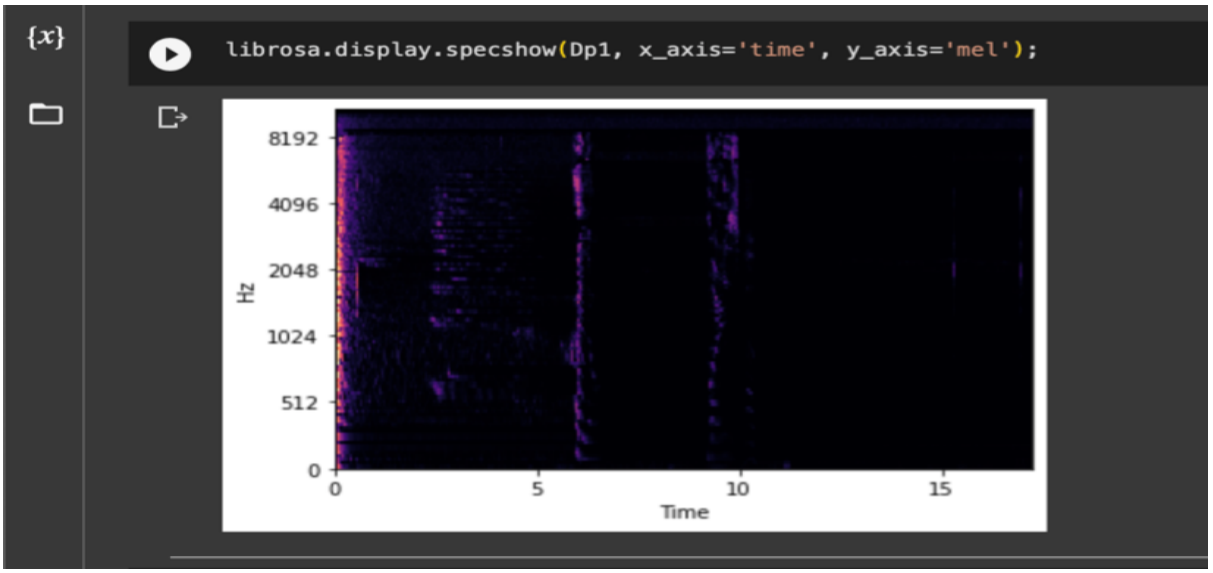
Colab Report on Noise Reduction performed using PCEN



We then apply the PCEN function on the spectrogram.

Figure 8

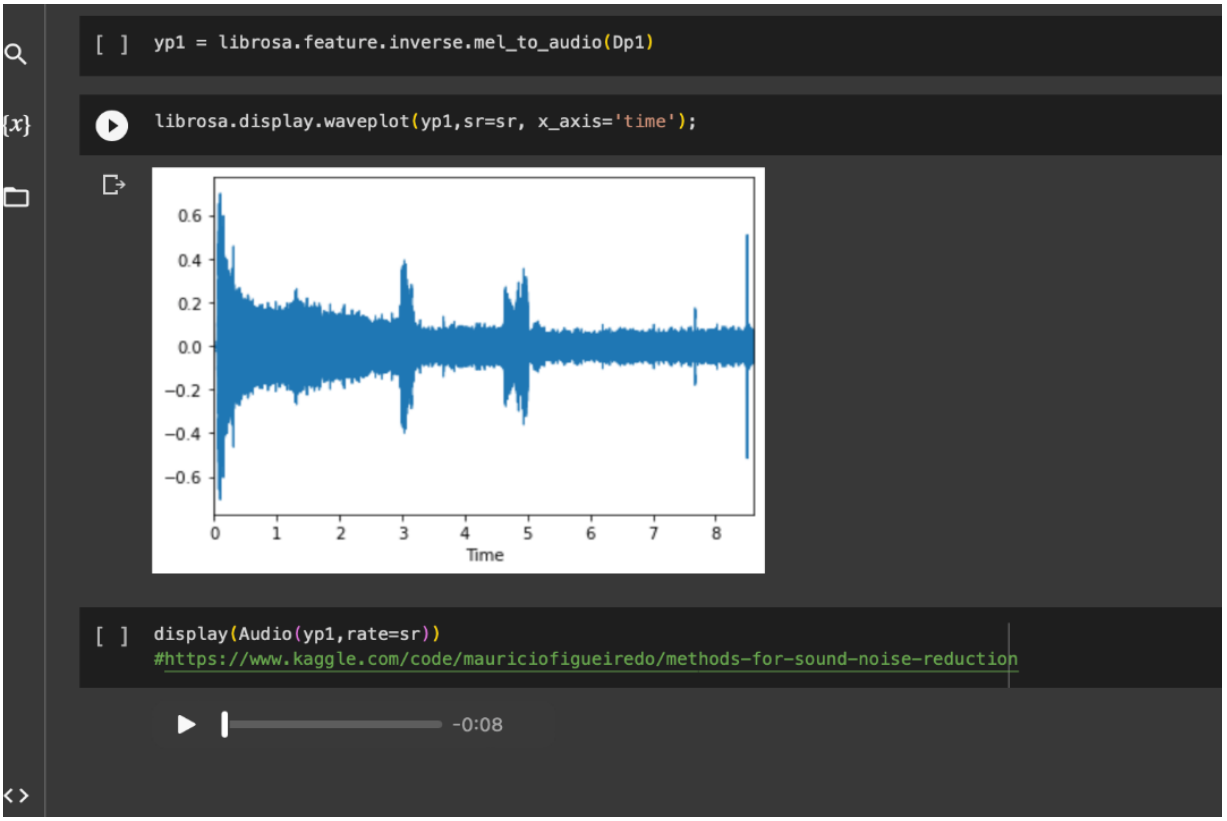
Colab Report on Noise Reduction performed using PCEN



Thus, it is clear how even for the implemented version of our code, the final spectrogram has a much higher contrast post PCEN deployment as compared to what we had initially. That spectrogram when converted back to an audio thus yields the noise reduced version of the input audio and this is the audio we then feed to the ML model for sound classification.

Figure 9

Colab Report on Noise Reduction performed using PCEN



Figures 3-9 are the weekly code reports on noise reduction.

A few questions we can further look into when dealing with PCEN as a way to reduce noise for an audio which caters especially to a model as important as one where a disease classification is at stake include looking into what thresholds for amplitude and frequency modulations the algorithm zeroes in on as it is possible that once a cough begins, that there is some steadiness in the following channels' modulation measures and that it remains fairly stationary or constant much like our background noise would and so in those particular channels, once the cough has begun until it subsides a minimal threshold might be necessary so that is not considered as noise owing to the aforementioned regularity (Lostanlen et al., 2018). However, with that suggestion, I would also like to present the caveat that a cough which is labelled TB positive could possibly have specific ups and downs in its frequency and amplitude modulations which could all perhaps be stripped off by such a noise cancellation algorithm and this might make it really difficult for the ML model which the

final noise reduced audio is fed to to differentiate the coughs if those modulations were the key differentiators between a positive case and a negative one.

Chapter 2: Data Augmentation using Audiomentations library

Now that we have addressed noise reduction, we can come to inflating the size of our dataset so that whichever model we decide upon for classification has enough data points to make accurate generalisations (Kaur et al., 2022). Too few data points could lead to better accuracy on the test data while the results which really matter to us, the predictions for unlabelled data might be way off than it should be for the model to make a better prediction than the flip of a coin would inform us about it. As such, we have size inflation techniques termed "data-augmentation" techniques in the pre-processing jargon (Kaur et al., 2022).

We use libraries with functions which can not only alter the original audio, but alter it in a way that the alteration preserves its labelled value and yet modifies it (Kaur et al., 2022). An example with our TB audio datapoint might clarify this further. Let's say we have an audio wherein a patient's cough begins on the 2nd second and goes on till the 4th second in an audio file which is 7 seconds long. Now, an augmented version of this datapoint might alter the pitch of the audio as we know that men and women could have different pitches and thus provide us with an additional datapoint which would mimic perhaps a woman's TB positive cough given the sound clip of a TB positive patient who identified as being a man and so had a lower pitch and vice versa. It could also perhaps stretch the cough to a length longer than its original time duration and make it suit those cases where TB positive patient has a cough which lasts longer than the original person's but still had the internal AM FM measures identical in ratio to those of a positive person and thus were likely to have the disease than not. However, it is important to highlight that the original label's preservation is important, else the augmentation might even make the dataset worse (Kaur et al., 2022; Lostanlen et al., 2018). Thus, we do not apply every single augmentation measure on every single datapoint

but carefully segregate and decide on specific ones for those which are TB positive and those that are TB negative (Kaur et al., 2022). This is because, if it is negative a bit of manipulation of the AM FM measures might not alter the labelling significantly even though exceptions are always possible- cases where AM FM measures of audio were manipulated into a negative audio making an audio through augmentation which is potentially positive for TB (Kaur et al., 2022; Lostanlen et al., 2018).

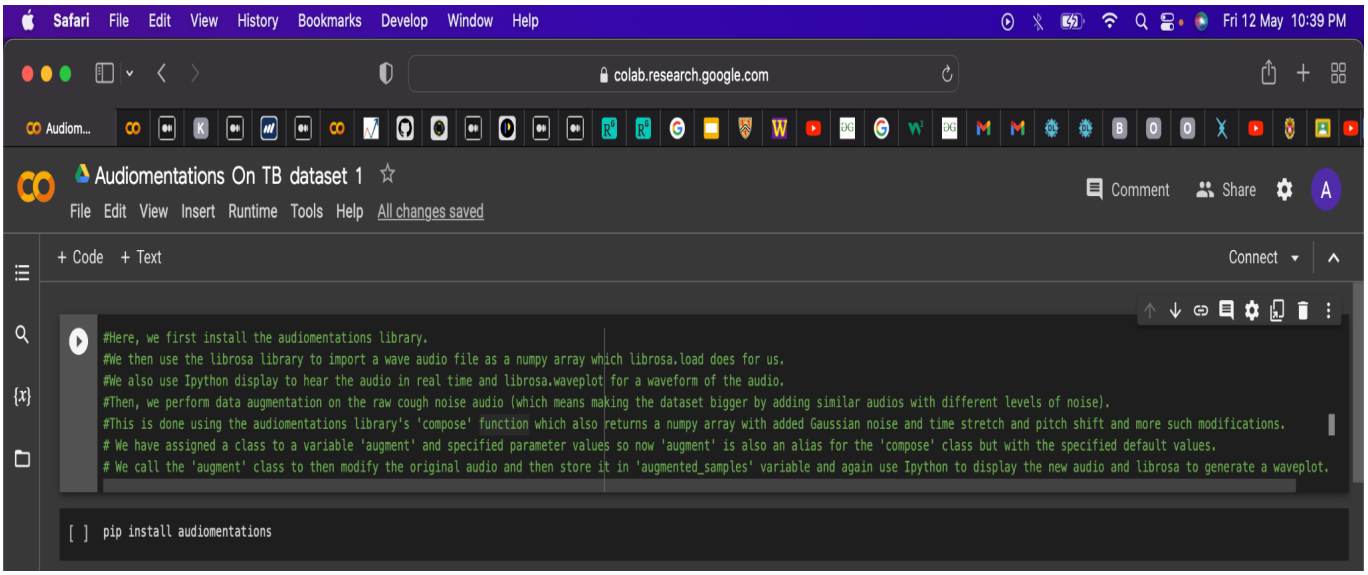
To achieve the above, we use Python's "Audiomentations" library which provides us with a host of functions to achieve the required augmentation on our dataset (Kaur et al., 2022). This is done following an approach taken by researchers working on "fall detection" in the elderly, people who are at the most risk of hitting the floor and thus requiring a device which would monitor them regularly (Kaur et al., 2022). This is so, once they fall, help could be provided at the earliest without also compromising on their privacy which certain devices might violate. The latter bit is in reference to the visual ML projects deployed in response to this problem statement which are not particularly useful in those locations inside the household where say a grandparent is most likely to trip and fall, the washroom (Kaur et al., 2022). Thus, a moving robot which just uses "ambient sound" to make a decision on whether to alert someone or not comes to be beneficial (Kaur et al., 2022).

Populating the dataset with augmented versions of the labelled data would thus give the model enough time to also adjust whichever loss function is being employed for classification. In SVM for example, this would be the hinge loss. As sampling the audio converts it to a list of numbers which can be understood as a vector, the longer the audio, the longer would be that single dimensional vector. To make the job of classification easier, before feature extraction, it is ideal for the audio to be divided into segments which can be analysed by the Transformer model used by Kaur, Wang and Shei (Kaur et al., 2022). This

yields as many segmented audios as the ratio of the length of the input audio and the duration of the segmentation decided prior to extraction (Kaur et al., 2022). Then an additional feature "diff" is calculated between each subsequent pair of vectors through an element-wise subtraction operation (Kaur et al., 2022) . This yields values which signify a certain rate of change between each subsequent vector and could prove crucial in classification as positive cases could hold a set pattern of average differences in the part of audio where the sound of cough is audible. These are ideas which could serve us very well in TB detection as the duration window could be customised based on the number of coughs made by the patients and then could further be used to not just calculate micro differences but also to calculate the differences between subsequent coughs (Kaur et al., 2022). It is, for example, possible to imagine even for a layman based on their experience of having suffered from a simple cold that theoretically coughs could get more intense towards the middle of a coughing fit and perhaps differ from the initial coughs or the concluding ones if multiple coughs are allowed in our short audio clips within a certain time period.

Figure 1

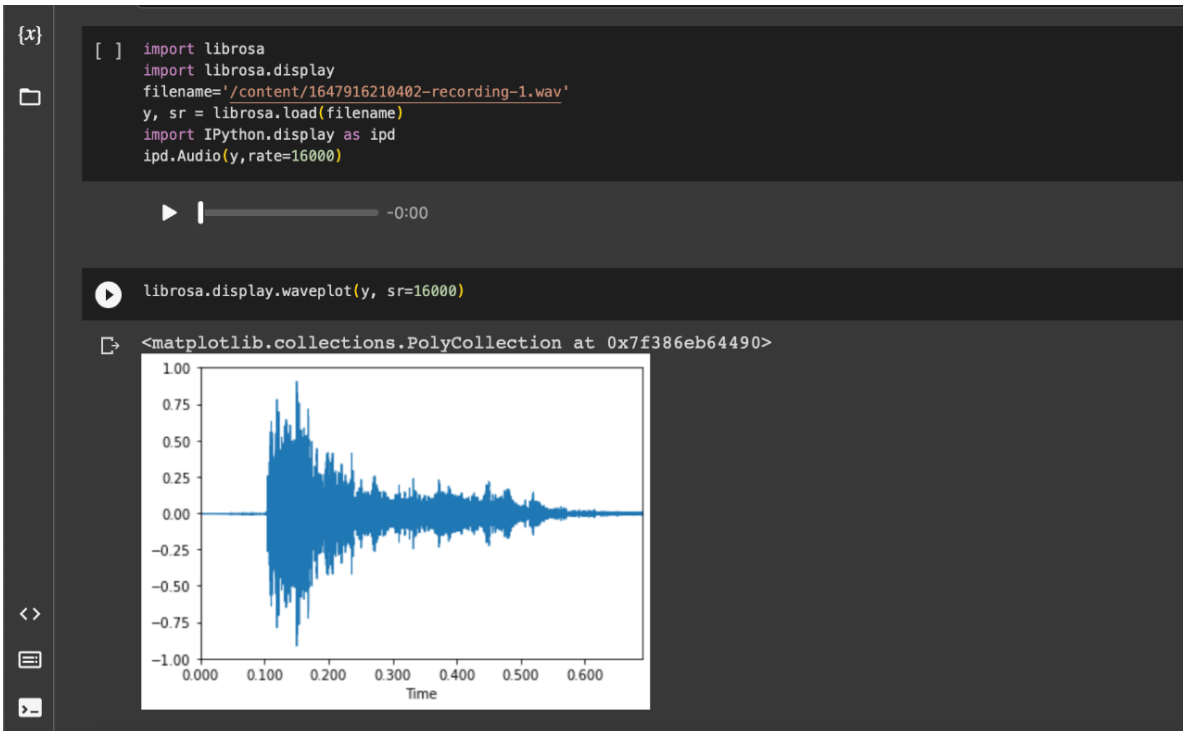
Colab Report on Data Augmentation



Here we installed the audiomentations library.

Figure 2

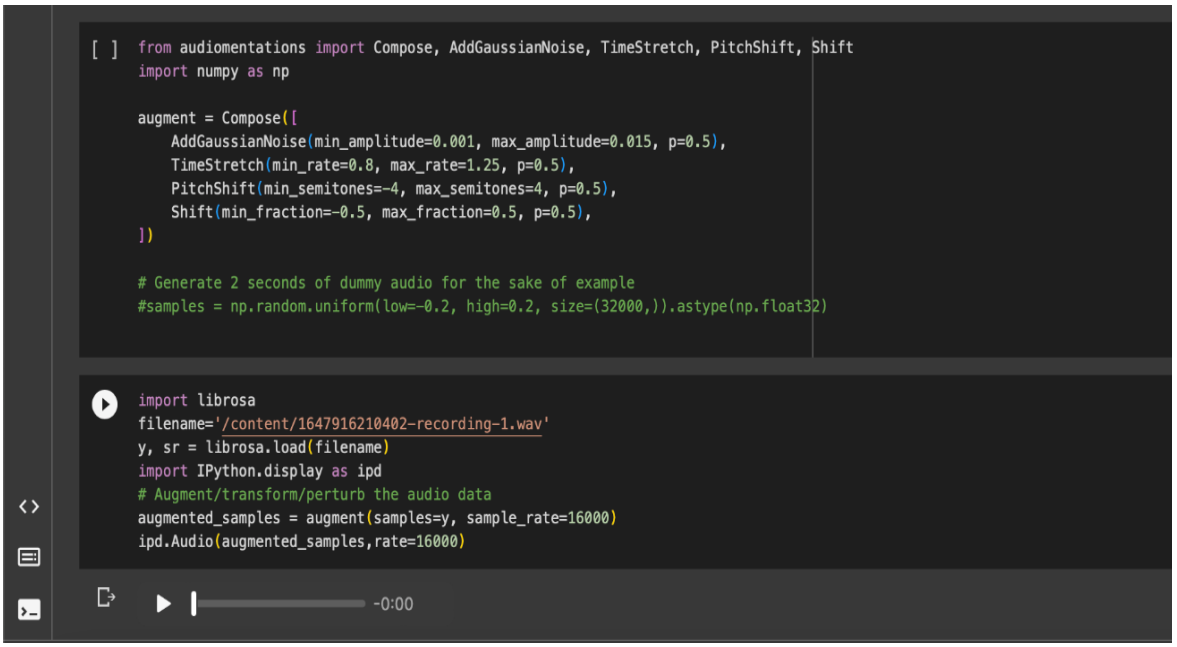
Colab Report on Data Augmentation



We then load the audio using Librosa's load function which stores it in an n-dimensional array and then we generate a waveplot as above.

Figure 3

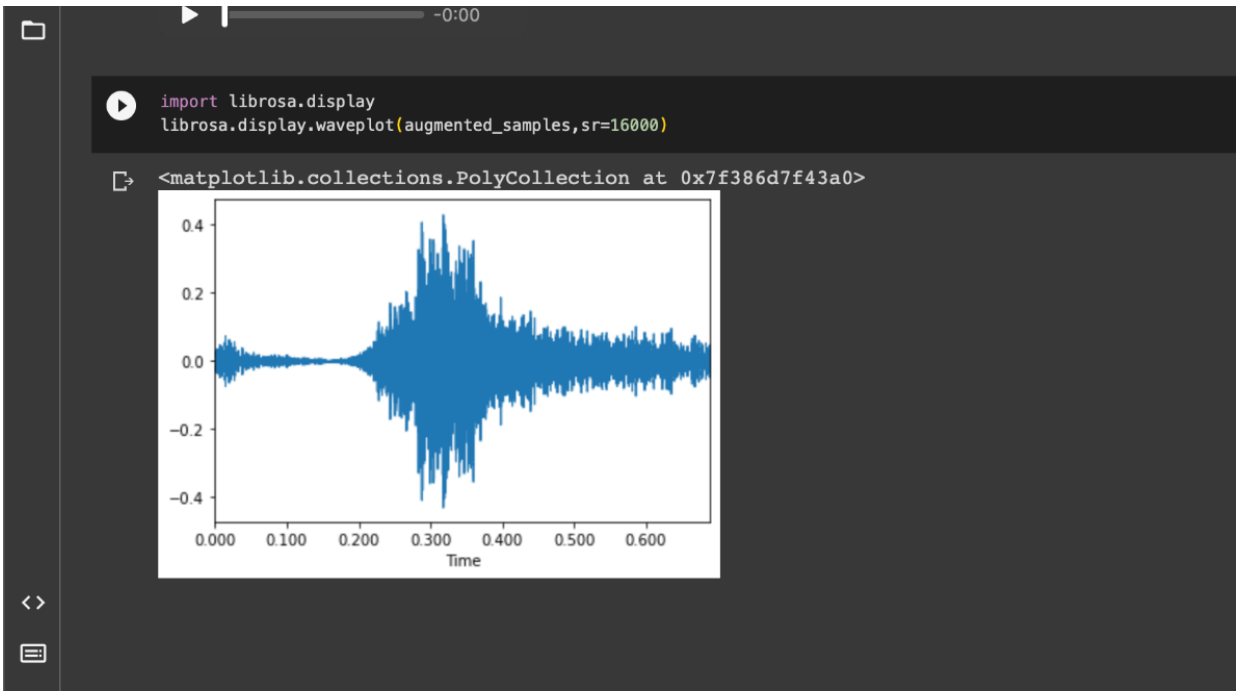
Colab Report on Data Augmentation



We then import all the necessary functions from Audiomentations library and call the Compose class on our audio file to generate new audio from our existing one (iver56, n.d.). The above code in figure 3 was inspired from the GitHub documentation on "audiomentations" by user iver56 (iver56).

Figure 4

Colab Report on Data Augmentation



Finally we generate a waveplot for the new audio generated using Compose.

Figures 1-4 are the weekly reports on Data augmentation.

Chapter 3: MFCC based approach to audio classification

MFCCs require an understanding of multiple concepts. Starting off with discrete fourier transforms and spectrograms, we need to understand the reasoning behind the log scaling of amplitude and mel scaling of frequencies and how to represent the same using a mel spectrogram (Valerio Velardo - The Sound of AI, 2020). Further, we need to understand what a cepstrum is, and how that is generated from the mel spectrogram. We can briefly look at the function which is at the heart of the conversion from mel spectrogram to a cepstrum using something called an “Inverse Fourier Transform” (Chaudhary, 2021; Valerio Velardo - The Sound of AI, 2020). Then we can look at what constitutes human speech and how “glottal pulse” and “spectral envelope” together constitute our words and how we scrape off the glottal pulse bit by using a cepstrum which gives us the major coefficients governing our speech and look at how we limit ourselves to the first few of those major coefficients so that we capture the essence of just the spectral envelope leaving much of glottal pulse data behind and are able to identify the speaker or in our case the patient’s TB results (Chaudhary, 2021; Gaudrain et al., 2009 ; Gutierrez-Osuna, n.d. -b; Valerio Velardo - The Sound of AI, 2020). A bird’s eye view of this would perhaps constitute looking at this process as extracting the main differentiators of a cough from person to person and identifying the characteristics of their cough’s spectral envelope so that similarities in the spectral envelope coefficients across positively tested patients could then perhaps be correlated by the model and hence be used on the test data points for classification into positive and negative results.

In order to analyse an audio, the more information we can squeeze out of the given input material the better. This is accomplished through something called a “Fourier Transform” (Chaudhary, 2021; Valerio Velardo - The Sound of AI, 2020). While we can always plot the amplitude of the given audio with respect to time and perhaps compare the respective values

of amplitude at different points in time, loudness which the amplitude measures is not enough information and so we decide to work with frequencies instead. This is because a soundwave is often composed of not one but multiple frequency sub waves. The resultant sum of the magnitudes of these individual frequencies is what constitutes a regular waveplot where on one axis is the amplitude and on the other time. Now, when we use a fourier transform, we extract out all the constituent frequencies and thus have even more data to compare. This when done on a discrete valued signal forms the basis of “Discrete Fourier Transform” or a “Fast Fourier Transform” (Chaudhary, 2021) . We often also decide to break the audio into multiple elements before performing FFT on it as it makes the classification model’s job easier.

Here the idea of a window comes in, which could vary from task to task. The window would basically determine how many frames or elements our audio clip is divided into. Terms like “hop length” are also common which identify that length of the window which has no intersection with the next frame window (Chaudhary, 2021; Maheshwari, 2021). But we keep the hops to values lesser than length of the window as otherwise we would end up skipping on certain sections of the audio and would have no information on their frequency distribution (Chaudhary, 2021; Maheshwari, 2021). A spectrogram is basically what helps us visualise these frequency details coupled with time data. Not only that, but we also get to understand the amplitude of each of these frequencies (Chaudhary, 2021; Maheshwari, 2021).

To understand log scaling we can take a mundane example to simplify things. Much like weight perception, wherein a dumbbell weighing 5 kg when compared to something weighing 10 Kg would have a much more perceptible difference in their comparative weights in contrast to two dumbbells which are 45 Kg and 50 Kgs respectively, our sound perception is also heavily dependent on the actual values of the two sounds comparatively (Kan

Samples, 2021; Mlearnere, 2021). Like in the dumbbell example, the weight differential is 5Kg in both cases, however it is not the differential which is the only determinant of the ease with which the weights can be told apart. It is highly probable that a human not used to lifting might falter and be unable to tell which of the two weights, 45 Kg and 50 Kg is heavier. Likewise, for amplitudes also, the loudness differential can be a deceptive indicator of how we would perceive different sounds (Kan Samples, 2021; Mlearnere, 2021). This is where log scaling helps us, as it factors in these perceptive differences for amplitude. For frequency, we use something even more advanced called the Mel scale which again has a formula which adjusts the original frequency in such a way that the differential between frequencies accounts for similar perceptual intricacies (Kan Samples, 2021; Mlearnere, 2021).

Figure 1

Formula for Conversion of frequency f to Mel Scale

$$m = 1127 \cdot \log\left(1 + \frac{f}{700}\right)$$

Note. This image was extracted from an article by Mlearnere called “Learning from Audio: The Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral Coefficients” written in 2021.

Figure 1 shows the formula which helps us convert the normal frequency, f, to the mel-frequency scale.

Finally, we can come to the idea of a cepstrum. A cepstrum is essentially the “spectrum of log spectrum” (Oppenheim & Schafer, 2004) . It has become quite clear how a view of the frequencies of the audio obtained through a transform helps us in seeing its various peeks and when we obtain the log spectrum, we essentially convert any multiplicative operation in the spectrum into sums and ease our job of removing some common unimportant factor which for us is the glottal pulse data (Chaudhary, 2021; Gaudrain et al., 2009 ; Gutierrez-Osuna, n.d.-a ; Valerio Velardo - The Sound of AI, 2020). When we remove that we are left with the essence of a cough, “the spectral envelope” (Gutierrez-Osuna, n.d.-b) . It has been found out that for speaker identification, the “ Glottal Pulse Rate” is not as good an estimator as the “Vocal Tract Length” (Gaudrain et al., 2009) and so similar logic could be extrapolated to our case of TB diagnosis as the cepstral coefficients which are representative of the VTL response could be taken into consideration for TB cough identification and hence GPR coefficients could be ignored (Gutierrez-Osuna, n.d.-a ; Valerio Velardo - The Sound of AI, 2020). As the VTL response is the main indicator for recognition, after applying the log operation, we can obtain our audio as a mixture of the two responses and then we focus on the VTL response frequency as that forms the spectral envelope. This is done through the usage of an “Inverse Fourier Transform” of the log mel spectrogram we had obtained earlier (Chaudhary, 2021; Gutierrez-Osuna, n.d.-b ; Valerio Velardo - The Sound of AI, 2020). The cepstrum helps in the “deconvolution” of our cough signal where the signal could be understood as a “convolution of glottal source and vocal tract” (Gutierrez-Osuna, n.d.-a ; Valerio Velardo - The Sound of AI, 2020).

Chapter 4 : Gradient Descent Algorithm implementation from Scratch and the Saddle Points Conundrum

If we try to formulate the algorithm of linear regression as a series of steps, then at its core is the estimation of variable values for which the error in the value predicted is the least. As long as we have an idea about the equation of a line we can infer the fact that we do have an equation of estimation to begin with for any given regression problem which is linearly solvable in its given set of variables (Miller, n.d.). For this chapter, we look at simple regression to get a hold of the differential calculus that's employed within this simple looking algorithm (Miller, n.d.). The starting estimation would also mean we understand how the linear graph of the feature variable and the final output variable's prediction compares with the scatterplot we would obtain from simply plotting the the results we do already have, the labelled dataset.

Once we have done this, we might question how to go about calculating the error. This forms an integral step as a simple addition of the differences between errors could mean we end up in a spot of bother where our error data is not representative of the actual error rate and does not encapsulate what we would have liked it to. We have two simple ways of going about solving this. One that we square the errors and then add as then we would have a final positive sum which we could then average out and have an estimate for the entire dataset (Miller, n.d.). Or we could think of the simpler way of going about this. If negative errors are the problem, then why don't we only alter them. Not only would that reduce the time complexity of our code as we would not have to go through every single data point to square it but only the negative values and make them absolute (Miller, n.d.). For a dataset that has millions of data points, such a reduction in complexity could still mean a good few

sub-seconds even though the complexity would be linear either ways. This is where our understanding of minimization helps us narrow down to one of the ways.

For minimization of the total average error, we would need to start off with random values for our variables and then increment or decrement them in such a way that when entered into the equation of the line, the output is as close as it can be to the actual final value (IBM, n.d. ; MIT 6.390, 2022). To know whether incrementing or decrementing is the right step, we would need to know what the rate of change between two variables is. However, even in simple linear regression there are two unknown variables contrary to our expectation of a single unknown. The two are- the coefficient for x and the y-intercept value. Thus we need a closer look at the impact that the change to each unknown has on the equation's final output. To understand this better, we try to find a way which makes every unknown other than the one we want to know about constant, and find the slope of the curve with respect to that particular variable. How the slope helps is that it gives us this exact information- the rate of change (IBM, n.d. ; Miller, n.d.; MIT 6.390, 2022). If the slope or what in calculus is called the derivative at a point is positive, it indicates a positive proportionality between the output and feature variable and likewise when it is negative. Going a step further we repeat this process with respect to each individual variable and in our case the two unknowns. This process called partial differentiation necessitates that the cost function we decide on is differentiable and hence we have no option but to rule out the absolute value idea as that is not differentiable and hence minimization would require some strategy beyond this (IBM, n.d ; Miller, n.d. ; MIT 6.390, 2022). But before we repeat this process of slope calculation comes another important step which is the updation in values. While we do understand the interconnection between slope value and the proportionality between the variables, we are yet to decide on an equation for updation (Turin, 2020). For this, we can think of going down the slope as the right step because we want to minimise the errors and if the slope is negative it

means that with an increase in the feature value, the error goes down. In such cases we want to keep moving to the side of the graph where the feature value is increasing. But on the contrary if the slope is positive at a point, we know that with an increase in the feature value, our cost is going to get steeper as well and so we want to move against the positive slope direction or decrease the feature value (Turin, 2020).

Next is determining by how much we would like to alter the value of our unknown features. This is called the "learning rate" and can play a very crucial role in the amount of time it takes for our algorithm to complete its course and whether it actually ever converges or not (Turin, 2020). Making alterations which are too drastic could make us miss the mark and hence we choose a small value for update (Turin, 2020). This is usually a fraction of the slope value. As positive slope means going against it and negative one means going with it, we can use an equation where we simply subtract a fraction of the slope value from the unknown variable. This way the negative slope makes the subtraction turn into an addition and hence serves the purpose for either type of slope value (Turin, 2020).

As we are minimising the cost, we are also moving towards minimal slope values and this naturally influences the size of the steps we take in the direction opposite the positive gradient. However, it is possible that we reach a value for the cost function such that slope is either exactly zero or hovers in the extreme vicinity of zero on the number line despite that cost value not being the minimal value for our slope. Such cases can happen at locations on the graph called "Saddle points" which are neither a local minimum nor a global minimum nor are they a maximum of any type (Ge, 2016; IBM, n.d.; Turin, 2020). They are just there with no specific role per se than to make our job tougher. Suppose on our climb down a hill, we reach a huge flat ground similar to a cricket ground except perhaps a lot bigger. In such a scenario for someone to figure that the stadium is still not the lowest point but just a flat

infrastructure built on some part of the same hill which is neither a valley nor a peak, would mean we travel to the edge of the ground and then see for ourselves that we need to travel more. For such a realisation to occur to our algorithm, given the small step size that we would be at because of the small gradient value might mean we take forever to escape that cricket ground. Not only would this affect our time complexity in a way which would render our algorithm run forever but also make our efforts futile (Ge, 2016; IBM, n.d.; Turin, 2020). Therefore, a few approaches have been considered to tackle this problem. An intuitive one is that we do not stop ourselves at the slope of the line with respect to the individual features but that whenever we do encounter a situation as the one described above, we calculate the second order derivative also, which is to say that we calculate the slope of the slope. As slope tells us whether the output variable's value changes with respect to a feature, the slope of the slope would tell us whether our slope itself changes with change in the feature values. If the second order derivative is zero that would mean the slope remains more or less constant and hence we might have to stop once and for all but this would not be the case otherwise- when slope of the slope is non zero (Ge, 2016; IBM, n.d.; Turin, 2020). Another approach is that we repeat the process of gradient descent by using a random sample for error calculation in each step and repeat the process a good number of times such that it converges. It has been found to work for most cases and is also often termed "Stochastic gradient descent" in technical terms (Ge, 2016; IBM, n.d.; Turin, 2020).

The weekly colab reports that follow this showcase an attempt at writing a gradient descent algorithm from scratch for a simple linear regression model.

Figure 1

Gradient Descent Implementation

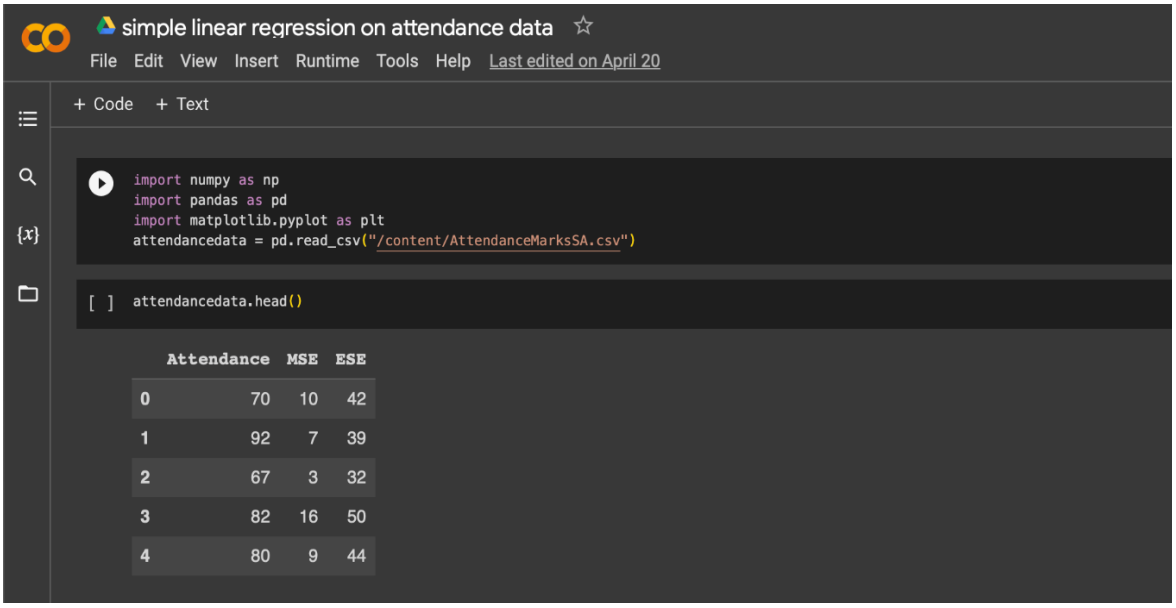


Figure 2

Gradient Descent Implementation

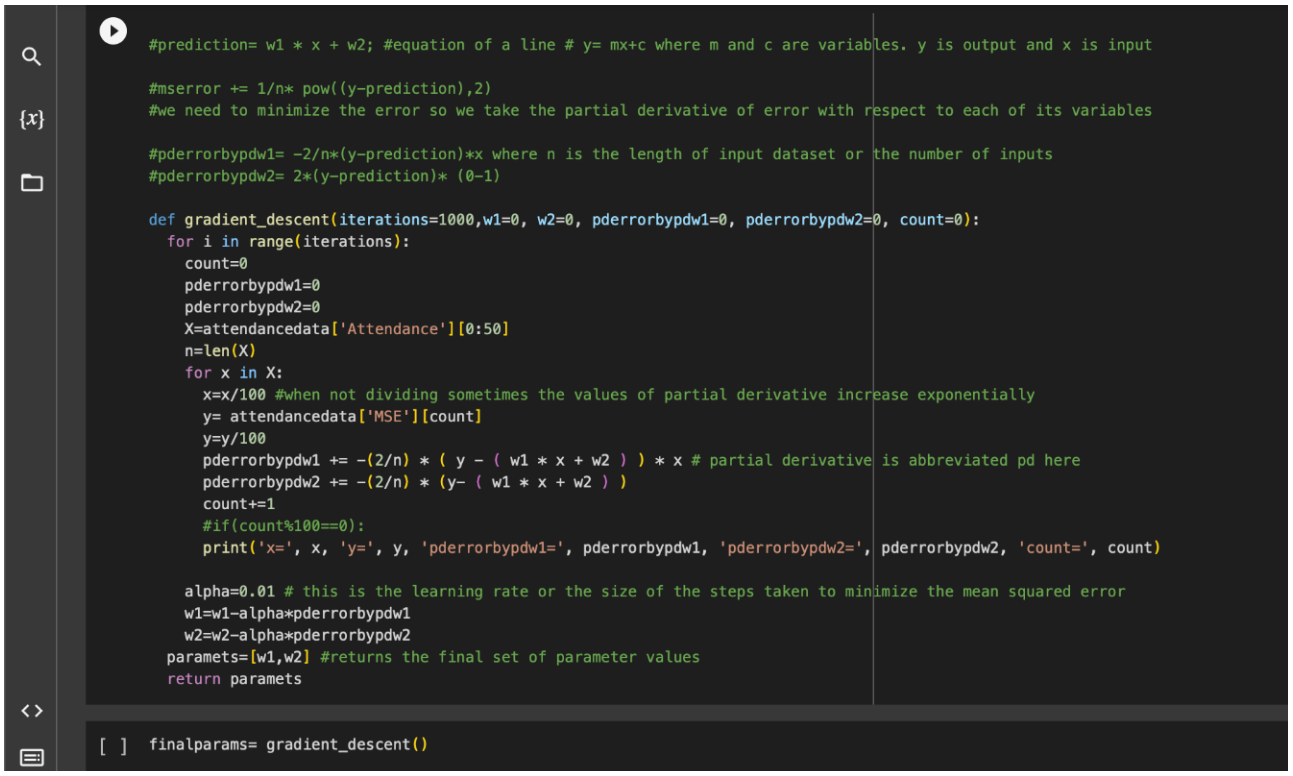


Figure 3

Gradient Descent Implementation

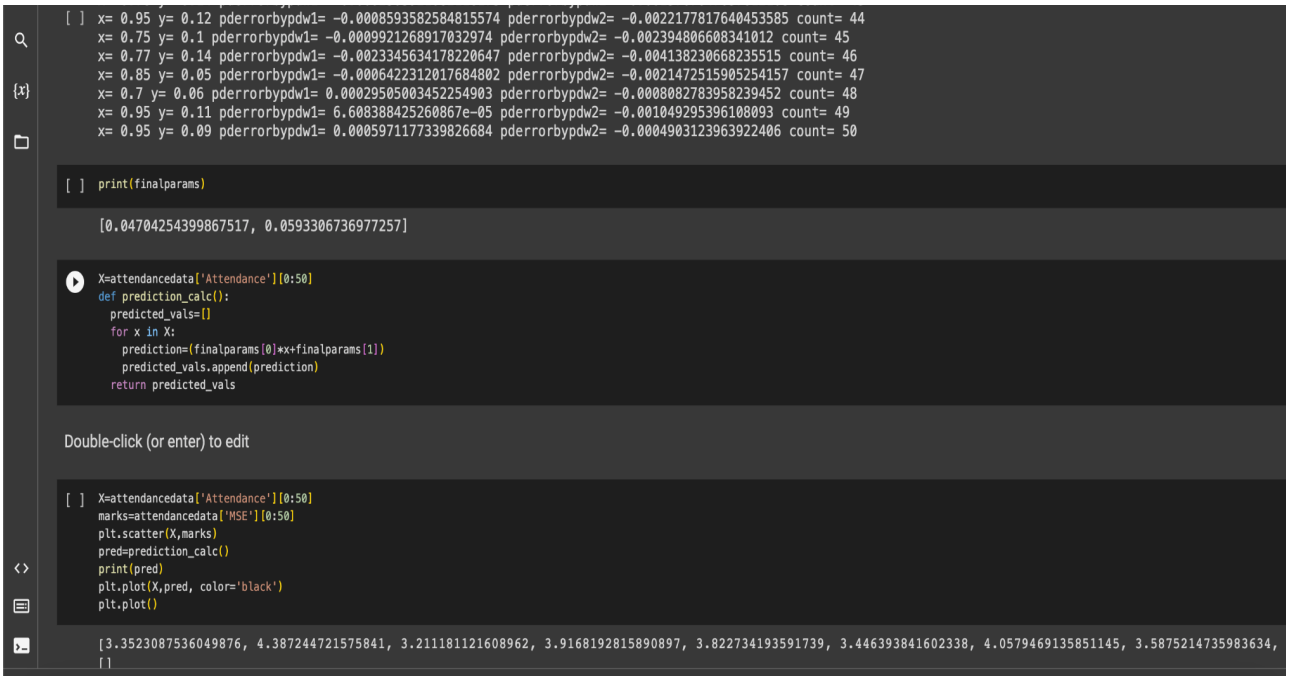
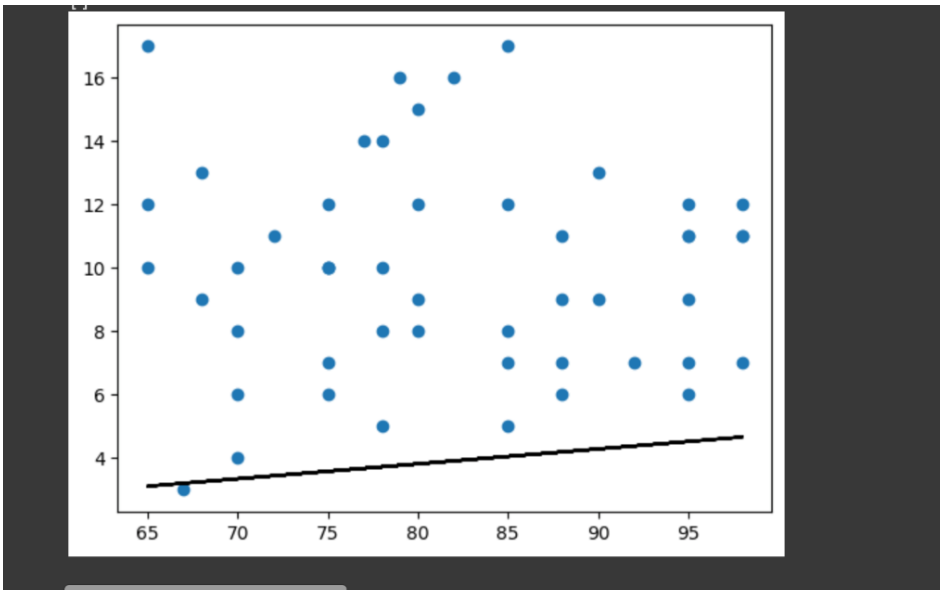


Figure 4

Gradient Descent Implementation Scatterplot



Conclusion

It can thus be concluded that any deployable acoustic ML/DL product remains incomplete without a series of operations being performed on the input audio before it gets fed into the model. These usually include data augmentation and noise reduction. It was observed that the PCEN algorithm when used on noisy audio improves the contrast in its mel-spectrogram which is an important insight if we want to implement an ML classification model such as SVM on noisy cough sounds. Likewise, Audiomentations library's compose function generated audios from the existing dataset preserving their inherent nature and final label which again is something we could incorporate in TB cough detection for better accuracy using a host of code-generated coughing and non-coughing audio files. Some essential concepts behind MFCCs, which can prove useful in feature extraction were also looked at. Finally a GD algorithm was developed from scratch which helped clarify some of the main mathematical concepts underlying minimization of cost functions in machine learning.

Appendix

Legendre and Least Squares based Optimization

Ashwin Salampuria

1020201229

Professor Rintu Kutum

ISM- Acoustic ML/DL for Public Health

April 24, 2023

The two unknowns in Simple Linear Regression : Weight and Bias

Weight (**m or w_0**) : The coefficient of input 'x' in our equation of a line.

Bias (b or c): The y-intercept value.

y = **m**.x + **b**

Or

y = **w_0**.x + **c**

We can start off with random values for 'm' and 'b' (say 0 for each) so we have an equation for our line of best fit and then we employ techniques to correct it.

Index:

1. Key suggestions made by Legendre in 1805 on finding least square.
2. Unpacking some Regression jargon:

a) Residuals.
b) SSR - Sum of Squared Residuals.
c) MSE- Mean Squared Error or Cost Function.
d) The algorithm of Gradient Descent - why does it work?

(Terms from Calculus: Partial Derivatives and their importance / role in Optimization)

e) Learning Rate.
f) Convergence.

An example scatterplot using Linear Regression

Source: https://learningstatisticswithr.com/book/lsr_files/figure-html/regression1a-1.png

Legendre's main ideas on least squares

- Legendre implores the usage of squared errors (for two possible reasons as we will see).

  Squared errors he believed **"is very well fitted to reveal that state of the system which most nearly approaches the truth".**

- **Recurrently suggests finding "the equation of the minimum with respect to one of the unknowns" (which leads us to modern calculus).**
- Uses integrals to perhaps hint at summing the changes made to unknowns.
- Uses something akin to the modern 'learning rate' to update the unknowns.

Source: https://www.york.ac.uk/depts/maths/histstat/legendre.pdf

SSR is the Sum of Squared Residuals for all samples.

number of samples, real value, predicted value

$$\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

sum of the errors of all samples

Mean, Error, Squared

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Source: Mean Squared Error (MSE)

Residuals, SSR and MSE / Cost Function

Residual is simply the error in prediction

real value, predicted value

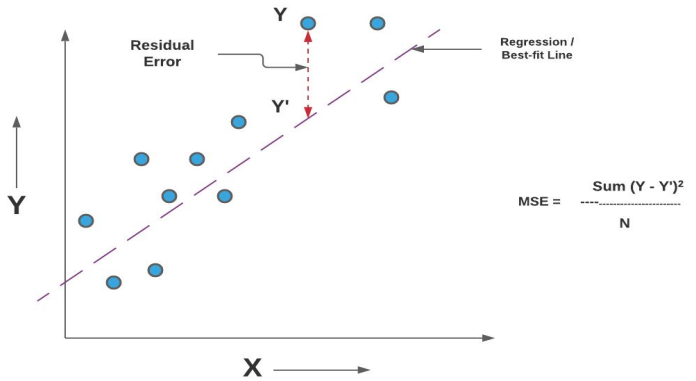
$$Y_i - \hat{Y}_i$$

error

We however cannot use this to calculate the mean of errors because:

- It might contain both positive and negative values.
- hence values might cancel out giving wrong idea about overall model error.

Source: https://suboptimal.wiki/explanation/mse/



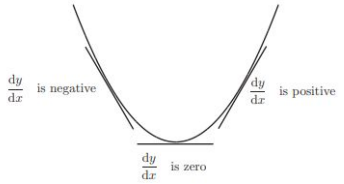
Source: https://medium.com/nerd-for-tech/linear-regression-the-history-the-theory-and-the-maths-e8333924b8a2

Gradient Descent

- An algorithm to find *Optimal Values* for weight and bias.
- Optimal Values : Minimize the error in prediction.
- **Problem Statement**: Find the values of ‘m ‘and ‘b’ for which MSE (our Cost Function) is minimum.
- **Approach**: Employing Calculus to find minimum.
- Prereqs to use Calculus: Use a differentiable function (This eliminates the choice of absolute value functions).

Why derivative?

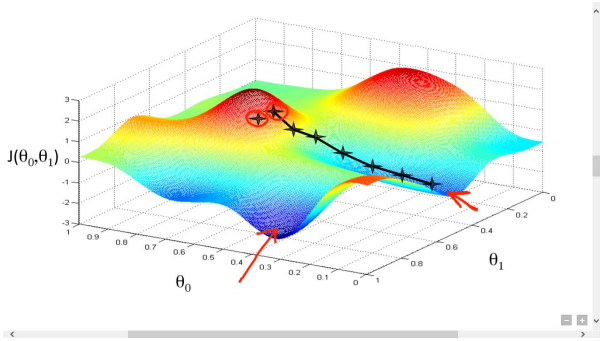
- ★ Essentially gives us the direction to move towards
- ★ Derivative gives us the slope or the instantaneous rate of change at a point.
- ★ Meaning how does one variable change with respect to a change in another.
- ★ We could use it to figure how our output variable changes with respect to each of the input variables affecting it.
- ★ But as we are dealing with more than one unknowns (the weight and the bias), we need to derive with respect to each to find how a change in either one affects the total loss and make those changes to reduce the error. And for that we need the partial derivative
- ★



Intuition behind Gradient Descent and Partial Derivatives:

The hill climb down example :

Note: This also shows why the starting point can be crucial as we can end at different Local minima



Source: https://www.youtube.com/watch?v=IH9kqpMORLM

Partial derivative

Finding the derivative with respect to each individual variable in an equation.

Gives separate equations for each variable, for example:

$$E(a,b) = \sum_{n=1}^{N} (y_n - (ax_n + b))^2 .$$

$$\frac{\partial E}{\partial a} = \sum_{n=1}^{N} 2\,(y_n - (ax_n + b)) \cdot (-x_n)$$

$$\frac{\partial E}{\partial b} = \sum_{n=1}^{N} 2\,(y_n - (ax_n + b)) \cdot 1.$$

Sources:https://web.williams.edu/Mathematics/sjmiller/public_html/BrownClasses/54/handouts/MethodLeastSquares.pdf , https://dotnettutorials.net/lesson/gradient-descent-in-artificial-neural-network/

Updation post finding the derivative

$$Repeat\ until\ convergence\ \{$$
$$\theta_j = \theta_j - \alpha \frac{dJ(\theta)}{d\theta_j}$$
$$\}$$

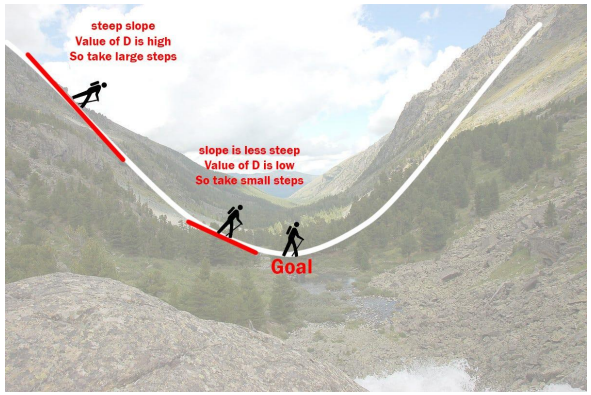
m = m - (learning rate * gradient of cost function w.r.t m)

b = b - (learning rate * gradient of cost function w.r.t b)

Note: for negative slope it automatically becomes positive and increases m or b

Source: https://dotnettutorials.net/lesson/gradient-descent-in-artificial-neural-network/

Gradient Descent



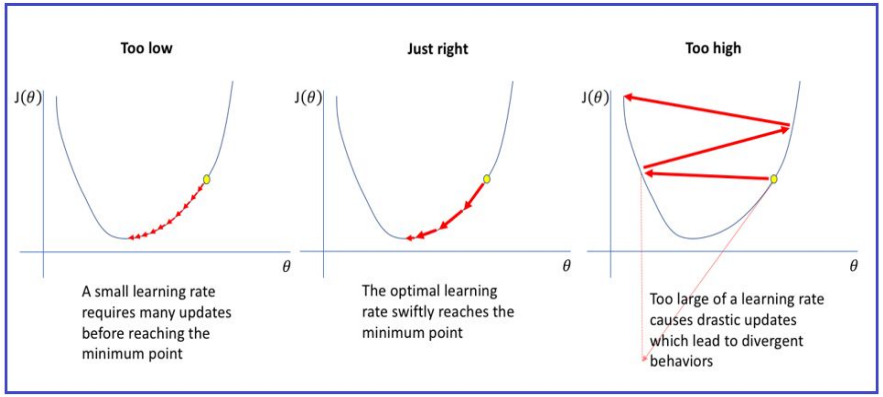
Source: https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931

Learning Rate and Convergence

Learning Rate : Basically the step size for coming down the mountain or more technically the value by which we change (increase or decrease) the variables in each iteration.

Convergence: When there is no considerable improvement in the loss function and we are stuck in proximity of the minima, the scenario is known as convergence.

Importance of the right learning rate



Source: https://dotnettutorials.net/lesson/gradient-descent-in-artificial-neural-network/

References:

1. https://www.york.ac.uk/depts/maths/histstat/legendre.pdf
2. https://web.williams.edu/Mathematics/sjmiller/public_html/BrownClasses/54/handouts/MethodLeastSquares.pdf
3. https://dotnettutorials.net/lesson/gradient-descent-in-artificial-neural-network/
4. https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931
5. https://learningstatisticswithr.com/book/lsr_files/figure-html/regression1a-1.png
6. https://medium.com/nerd-for-tech/linear-regression-the-history-the-theory-and-the-maths-e8333924b8a2
7. https://suboptimal.wiki/explanation/mse/
8. https://www.youtube.com/watch?v=IH9kqpMORLM

Thank you

References

Chaudhary, K. (2021, June 4). *Understanding audio data, Fourier transform, FFT, spectrogram and speech recognition*. Medium. https://towardsdatascience.com/understanding-audio-data-fourier-transform-fft-spectrogram-and-speech-recognition-a4072d228520

Gaudrain, E., Li, S., Ban, V. S., & Patterson, R. D. (2009). The role of glottal pulse rate and vocal tract length in the perception of speaker identity. https://hal.science/hal-02144510/document

Ge, R. (2016, March). *Escaping from Saddle Points*. Off the convex path. http://www.offconvex.org/2016/03/22/saddlepoints/

Gutierrez-Osuna, R. (n.d.-a). L7: Linear Prediction of speech - Texas A&M University. https://people.engr.tamu.edu/rgutier/lectures/sp/l7.pdf

Gutierrez-Osuna, R. (n.d.-b). L9: Cepstral Analysis - Texas A&M University. https://people.engr.tamu.edu/rgutier/lectures/sp/l9.pdf

IBM. (n.d.). *What is gradient descent?*. IBM. https://www.ibm.com/topics/gradient-descent

Ick, C., & McFee, B. (2021, February 6). *Sound event detection in urban audio with single and multi-rate PCEN*. arXiv.org. https://arxiv.org/abs/2102.03468

iver56, iver56. (n.d.). *Iver56/audiomentations: A python library for audio data augmentation. inspired by albumentations. useful for machine learning.* GitHub. https://github.com/iver56/audiomentations

iZotope, Inc. (2019, April 11). *Understanding spectrograms*. iZotope. https://www.izotope.com/en/learn/understanding-spectrograms.html

Kan Samples. (2021, February 15). *Loudness and amplitude explained: The decibel, human perception and units of measurement*. KAN Samples. https://blog.kansamples.com/2021/02/15/loudness-and-amplitude-explained-the-decibel-human-perception-and-units-of-measurement/

Kaur, P., Wang, Q., & Shi, W. (2022, August 23). *Fall detection from audios with audio transformers*. arXiv.org. https://arxiv.org/abs/2208.10659

Librosa.load¶. librosa.load - librosa 0.10.0 documentation. (n.d.). https://librosa.org/doc/latest/generated/librosa.load.html

Lostanlen, V., Salamon, J., Cartwright, M., McFee, B., Farnsworth, A., Kelling, S., & Bello, J. P. (2018, November). Per-channel energy normalization: Why and how - IEEE xplore. https://ieeexplore.ieee.org/ielaam/97/8539853/8514023-aam.pdf

Maheshwari, H. (2021, September 10). *Terms you need to know to start speech processing with Deep Learning*. Medium. https://towardsdatascience.com/all-you-need-to-know-to-start-speech-processing-with-deep-learning-102c916edf62#:~:text=Hop%20length%20is%20the%20length,portion%20of%20the%20window%20length.

Miller, S. J. (n.d.). The method of least squares - williams college. https://web.williams.edu/Mathematics/sjmiller/public_html/BrownClasses/54/handouts/MethodLeastSquares.pdf

MIT 6.390. (n.d.). Chapter gradient descent - introml.mit.edu. https://introml.mit.edu/_static/fall22/LectureNotes/chapter_Gradient_Descent.pdf

Mlearnere. (2021, April 14). *Learning from audio: The Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral coefficients*. Medium. https://towardsdatascience.com/learning-from-audio-the-mel-scale-mel-spectrograms-and-mel-frequency-cepstral-coefficients-f5752b6324a8

Oppenheim, A. V., & Schafer, R. W. (2004, October). From frequency to quefrency: A history of the cepstrum. https://www.researchgate.net/publication/3321562_From_Frequency_to_Quefrency_A_History_of_the_Cepstrum

Turin, A. (2020, February 4). *Gradient descent from scratch*. Medium. https://towardsdatascience.com/gradient-descent-from-scratch-e8b75fa986cc

YouTube. (2020, October 5). *Mel-frequency cepstral coefficients explained easily*. YouTube. https://www.youtube.com/watch?v=4_SH2nfbQZ8

日々、学ぶ. (2022, November 12). 【*pcen*】対数*log-mel*に代わる特徴量*pcen*【*librosa*】. 日々、学ぶ. https://take-tech-engineer.com/librosa-pcen/