Detection of COVID-19 by Cough

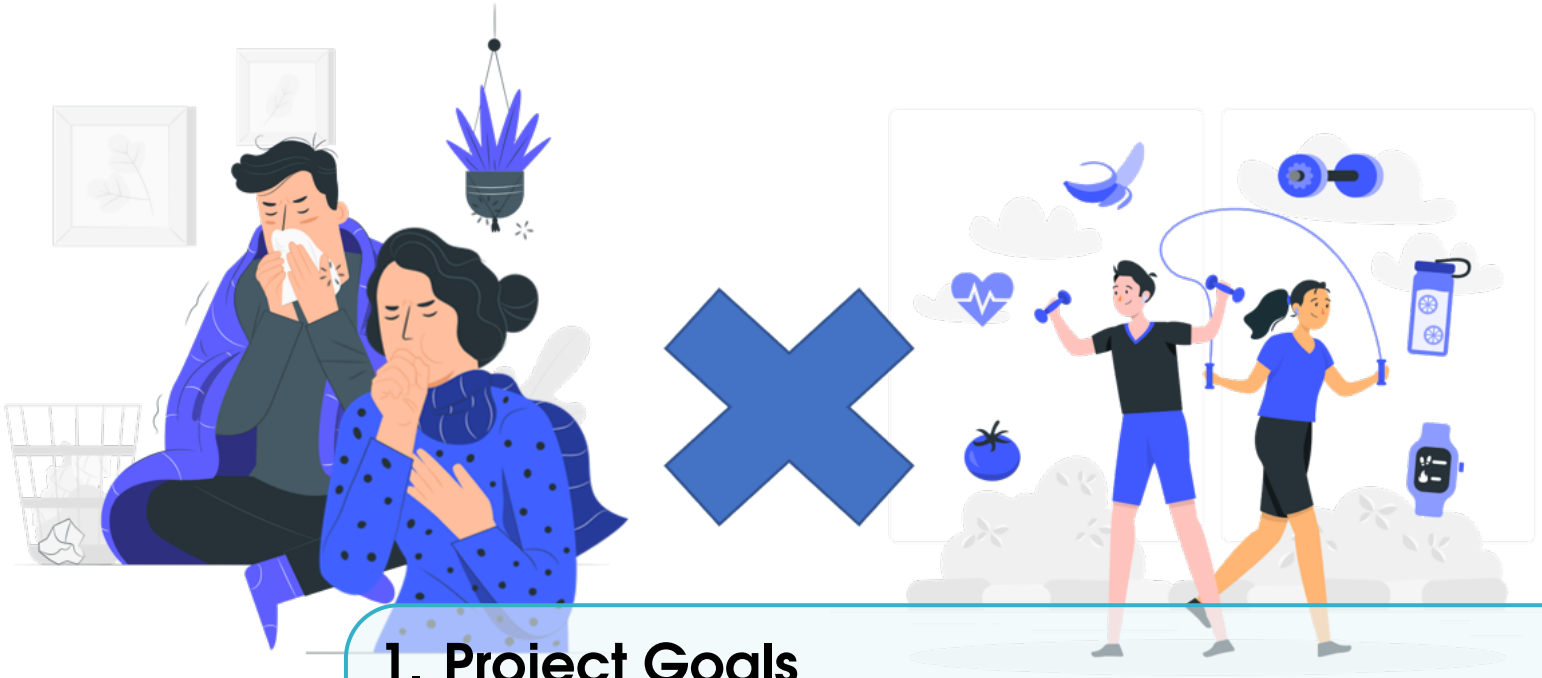
An approach with TinyML

Luccas Delgado Targa (ECA) - 2019011503
Pedro Henrique Lemes de Souza (ECA) - 2019015208
Mateus Dumas Lima (ELT) - 2017017965
Stéfany Coura Coimbra (ECO) - 2019008562

IESTI01 - TinyML

Unifei - Itajubá, August/2021

Contents



1. Project Goals

We are living in a very different situation compared to some time before. That's because since last year we are facing a difficult and complex enemy that is Covid-19. New studies and researches are being making and new ways are being tested to combat that diesese. With that in mind, we decided to study a way to detect Covid-19 by cough. We based in very sources, datasets and projets and we hope to see the final result, with a funcional model even not being a scientific one.

Our biggest motivation is the fact that even after a long time, some communities either still do not have full access to COVID-19 tests or need more tests to combat the pandemic. It is possible to notice this problem when we pay attention to the several headlines about this subject around the world.

It is important to say that we expect to have a trained model for COVID-19, but the analysis can be used for detection in other respiratory illnesses that also cause the same problems analyzed. Hence, by changing only the dataset and keep the same methodology, it may be possible to identify new kinds of viruses with this model, while new tests have not yet been produced at scale.

2. Project Description

2.1 About Covid-19

The Covid-19, according to the Brazilian Ministry of Health ("Ministério da Saúde", in portuguese), is a respiratory infection caused by the coronavirus SARS-Cov-2. It's a serious illness that has a high transmissibility. Because of that, we are living in a situation that we have to take extremely care of us and our family.

The SARS-CoV-2 has registers to be discovered at China, on December 2019 and could spread and infect people as we are seeing in this pandemic.

2.2 Why use Cough Detection?

More and more studies are being made to help the health system and each one to identify earlier the Covid-19 and to fabric vaccines that could covers more virus mutations. New ways to reach that, like analyse images of lungs are being considered and machine learning is a form to do it and it's already in use for many students and universities. An interesting way is by cough detection. The Qualcomm Institute published a video that shows a work of a professor from UC San Diego, his son and a researcher from University of Bordeaux. According with their study,

> Coughs and other vocal sounds contain pulmonary health information that can be used for diagnostic purposes, and recent studies in chaotic dynamics have shown that nonlinear phenomena exist in vocal signals.

The following steps of the cough detection deployed by them could be founded at: `https://link.springer.com/content/pdf/10.1007/s42979-020-00422-6.pdf`. Their dataset was used in a partneship between Corona Voice Detect project, Voca.ai and Carnegie Mellon University.

2.3 The general ideia of the model

The general idea of our model is detect anomalys in the cough of the users, the input data should be captured by one microphone contained in the pcb of our Arduino Nano 33 BLE Sense, then this data will be process using different types of filters and techniques of dsp (digital signal processing) then this modifyed data should enter the model to be process and give us a result, if the user has covid or not. In this study, we intended to build a model that detects if the person has Covid-19 or is a Healthy one.

It's very important to say here that this is only an academic project and not substitute a medical diagnosis.

2.4 What kind of neural networks we used and how We did that?

As planned before to the project, We used a CNN (Convolutional Neural Networks) after we pre-process our data (transforming the time series - audio signals - in signals at frequency domain, obteining spectograms - images), because when we use that kind of filter, we are be able to deal with images. Between convolutional layers we used Max Pooling, which is a special mechanism to extract the maximum value of each square with an especific size. That way, we could sumarize the images and try to reach a better model. After that, we had to use a Flattering Layer, to put into a vector to finally give the predict classification of data, using the Softmax. The Softmax was used because we have two different labels: "Healthy" and "Covid-19" and it's an activation function that transforms the vector of numbers into a vector of probabilities, more convenient for us. An example of

how the model works, with a general panorama can be seen by the image below.

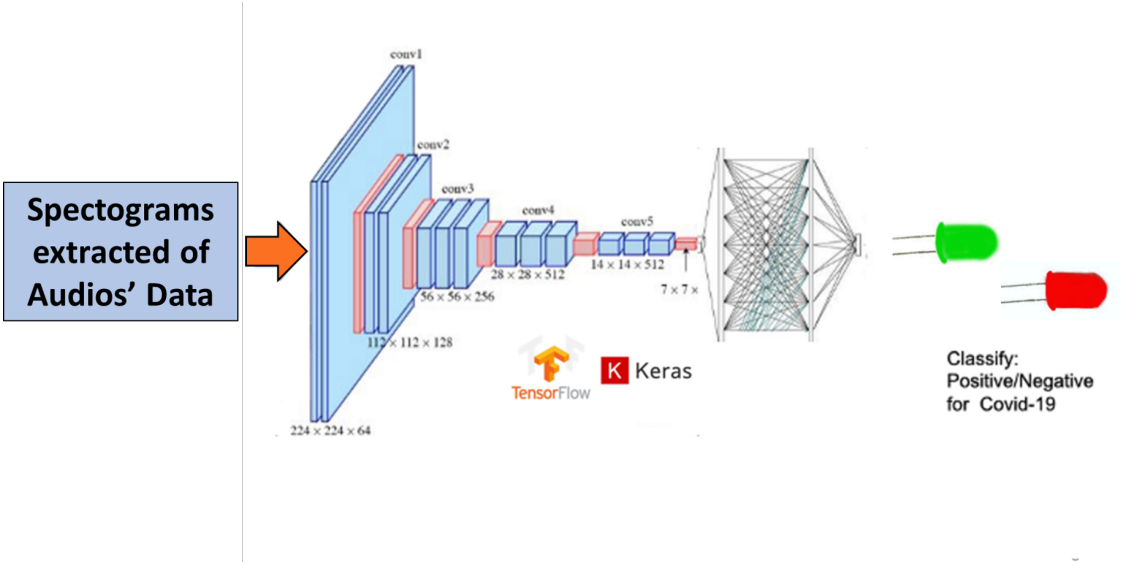
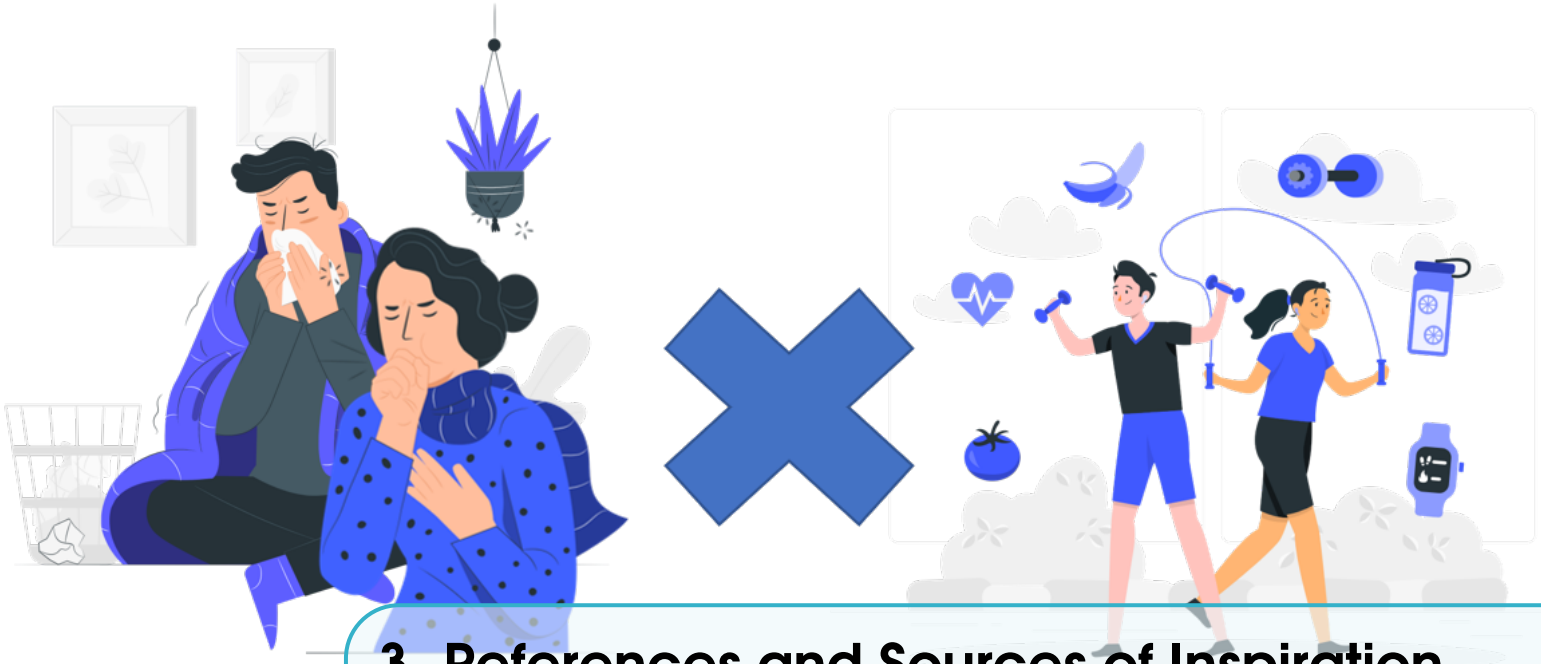


Figure 2.1: Example of the layers used in the project
Fonte: Build image based on presentations made by Prof. Marcelo Rovai

2.5 What do we need to consider to develop this kind of project?

The first thing to be said, that despite this project works with a detection of covid, this should not replace a real exam, we are just doing this for academical purpose, to learn more about neural networks and TinyML. Besides that, other concerns that we should have is about our dataset, because we need to look for a dataset balanced with a considerable quantity of data that comprehends as much as possible conditionals that could trained the model properly.

Therefore, for our data set we will use "The COUGHVID crowdsourcing dataset", published by Nature on June 23, 2021. At: `https://www.nature.com/articles/s41597-021-00937-4`



3. References and Sources of Inspiration

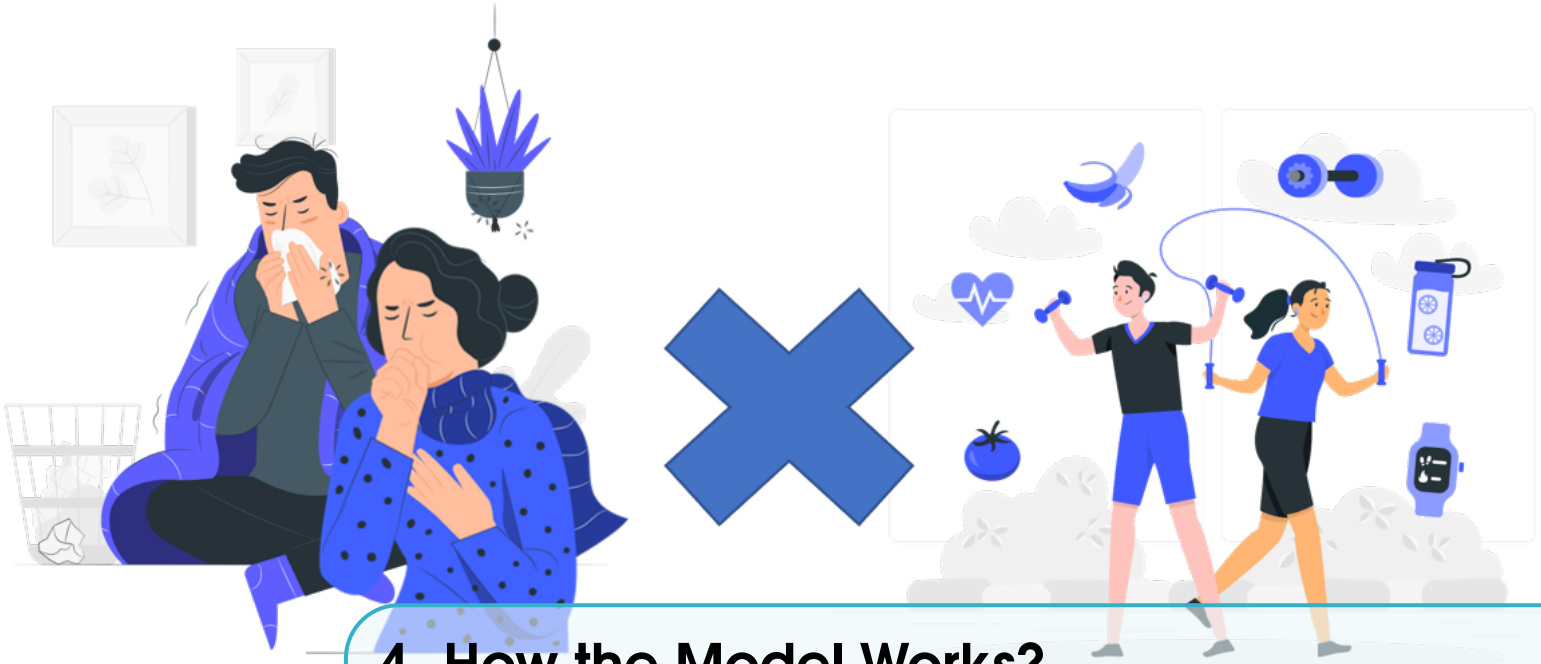
We made an extensive research to see how It works and If was another similar model or sources that should be used to guide us. From the beggining of annoucement to build a project that use TinyML, the inspiration was: "Why not try to build a model that is connected with our reality now?". That's was the inspiration. Even if the model doens't have the certification to be used in health field, we could study about to see how cough detection works and how to deploy that in a tiny dispositive such as Arduino. That's a value knowleadge nowadays.

Because of that, with searches, we founded a project made by a group of people and posted at EdgeImpulse that use TinyML, an Arduino BLE Sense, TensorFlow and EdgeImpulse to build a project that detects Covid-19 by cough. This study use a small database, but could reach some results and see improvements. The access link to this project: `https://bit.ly/3pZQrTe`.

More researches are being made in this area, because It's a potencial way to be a new form of understand the diesese. Another three studies were founded, one by MIT (`https://bit.ly/3gGvl8o`) and the others by Stanford University (`https://stanford.io/3zzHPHm`) and University of Cambridge. Cambridge, for example, developped an app to collect cough audios from people and they have more than 40000 data to be tested there (`https://www.covid-19-sounds.org/pt/`). The content and infor-

mation about the data could be found at `https://bit.ly/3q3QNsk`. We tried to contect them to possible use this dataset, but they didn't answer us back yet. We also founded a public dataset, made by Embedded Systems Laboratory (ESL), at Switzerland, that was build to contribute with studies.

About the way we should treat our data, before they enter in model, a contributor to that dataset build a drive with a collection of data and code to helps pre-proccess all the audios, to extract features of them, what we need to work at a easier way.



4. How the Model Works?

4.1 A simple block diagram

So, before actually starting the project, we made a diagram to help on the process. The final project that we intended to do show from the data collection to deplyment on Arduino 33 BLE Sense, using the TinyML kit (`https://store.arduino.cc/usa/tiny-machine-learning-kit`). It's important to see that in all projects it's crutial the step-by-step below. First, we decided a goal and with that, we already started to think about it if it's possible or not reach it and, at least, even with difficults, there's a way to do it. In that case, we saw that wasn't going to be easy and there was a lot of obstacles in our way. After that, we could start to think about the data and we can't measure how important is this step, because it can be modelling all steps next and make the job makeable, or not. Our next move was to make the Featire Engineering and you could ask: "Once I have the dataset in my hands, why can't I just skip to build the model?" Well, let's think about that: imagine you have a lot of images on different devices and you want to load them into you computer, but each one have a distinct input and your PC only accepts USB. How could you handle with that? Well, you have to use adapters to make that all devices had a way to have USB port so you can load you images, in other words, you need to specify and deal with a pattern and a limitation. Here, we have the same logic, we need a dataset with some patterns and somethings are desible, such as different genders, nationalities

and backgrounds to involve more people as it can, thinking about the user experience. With the data already settled, we could finally think how to treat that by building our model. We design a model architeture to work based on our input, that are audios. A great way to deal with it is by using filters to convert them into images and then, work with convolutional layers. After, we trained and tested the model and it's a two-way street (training and testing). To a complete project, we would have to deploy on Arduino, evaluate and troubleshoot with real people, but we focused on the dataset and model here.

The image below shows a simple way to represent a real project with the theme we wanted to approach:

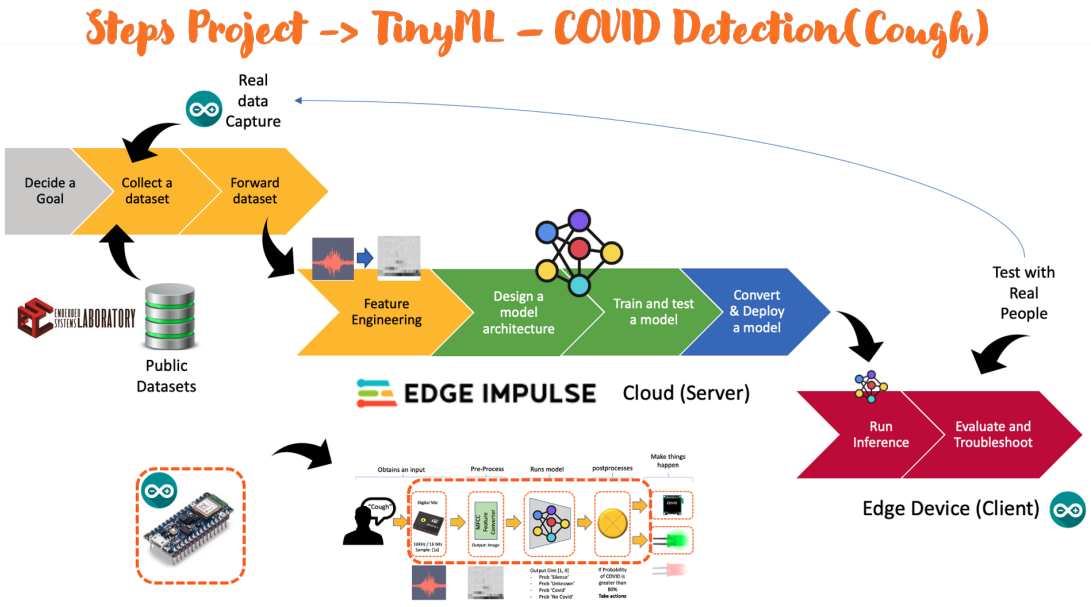


Figure 4.1: General Block Diagram
Fonte: Prof. Marcelo Rovai

4.2 Data collection

We used the database from ESL (Embedded Systems Laboratory), because they have more than 20000 data, with a big variety of people, like gender, nationality and age. You can access the project by the link below that have the links to the "open database access", "open code access" with some ways to treat the data and extract features using the Python as programming language and Jupyter Notebook and their publication on Nature Scientific Data (`https://www.nature.com/articles/s41597-021-00937-4`), explaining the dataset and his characteristics.

According with the article, "all of the recordings were collected between April 1st, 2020 and December 1st, 2020 through a Web application deployed on a private server located at the facilities of the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland". These recordings were preprocessed by lowpass filtering (fcutoff = 6 kHz) and downsampling to 12 kHz. Besides that, some audios was identified by some characteristics from four expert physicians:

- Quality: Good; Ok; Poor; No cough present
- Type of cough: Wet (productive); Dry; Can't tell
- Audible dyspnea
- Audible wheezing
- Audible stridor
- Audible choking
- Audible nasal congestion
- Nothing specific
- Impression: I think this patient has...: An upper respiratory tract infection; A lower respiratory tract infection; Obstructive lung disease (Asthma, COPD, ...); COVID-19; Nothing (healthy cough)
- Impression: the cough is probably...: Pseudocough/Healthy cough (from a healthy person); Mild (from a sick person); Severe (from a sick person); Can't tell

The audio files on dataset have two extensions: .webm or .ogg. For all data, the sampling frequency was 48 kHz.

The link to the project mentioned, with the link for the dataset and paper: `https://coughvid.epfl.ch/about/`.

The link to a GitHub from a contributor that put togheter some resources at Drive and make a code to treat part of the data: `https://github.com/covid19-detection/data_collection`. We used because of the labels and percentage of certain to be a ciugh from a healthy person or with one with Covid-19, just 837 samples labeled as "Healthy" and 441 samples labeled as "Covid-19".

4.3 What did we use to load data and build the model?

In the project, we used the Edge Impulse platform to load data and build the model. To create an account and create a project, you can follow the steps below:

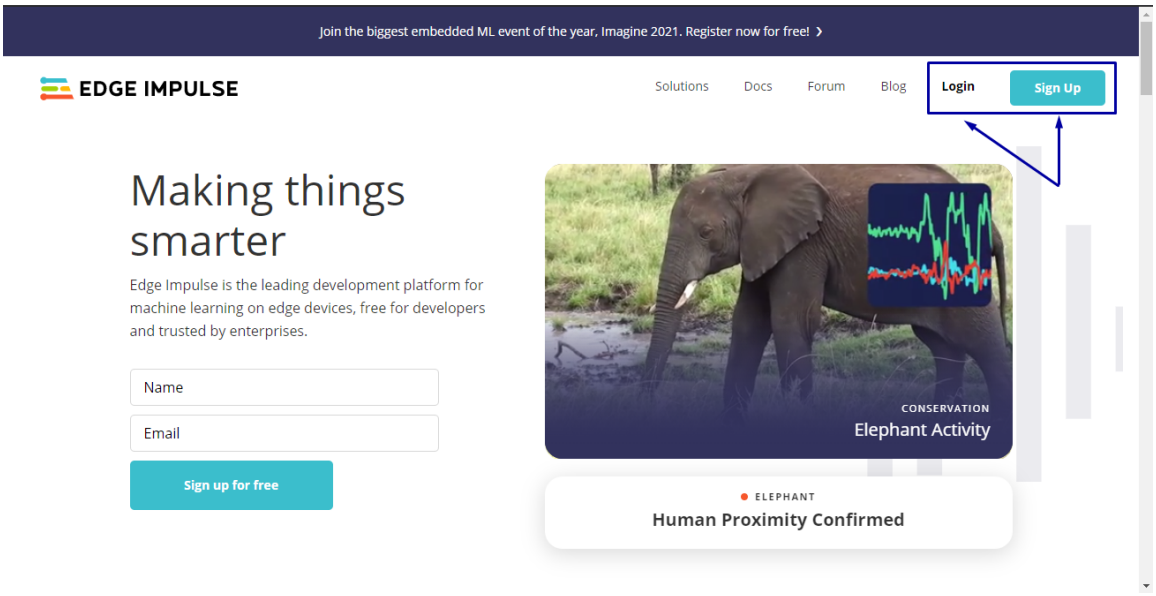


Figure 4.2: Login - Edge Impulse
Fonte: Edge Impulse

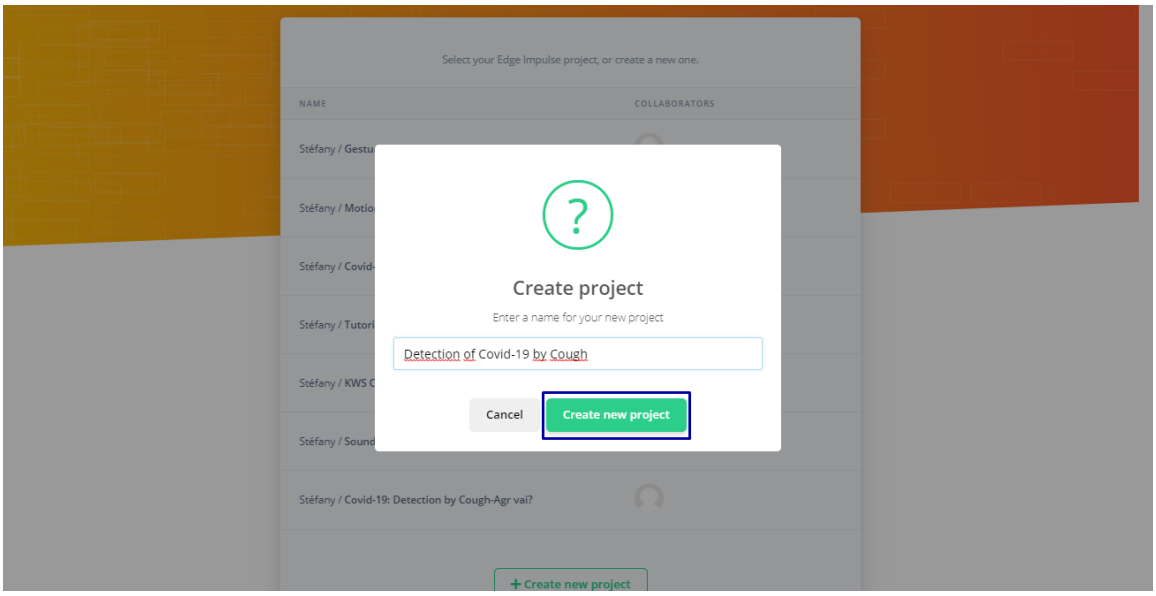


Figure 4.3: Create New Project
Fonte: Edge Impulse

The important tabs to pay attention, build a model and deploy it:

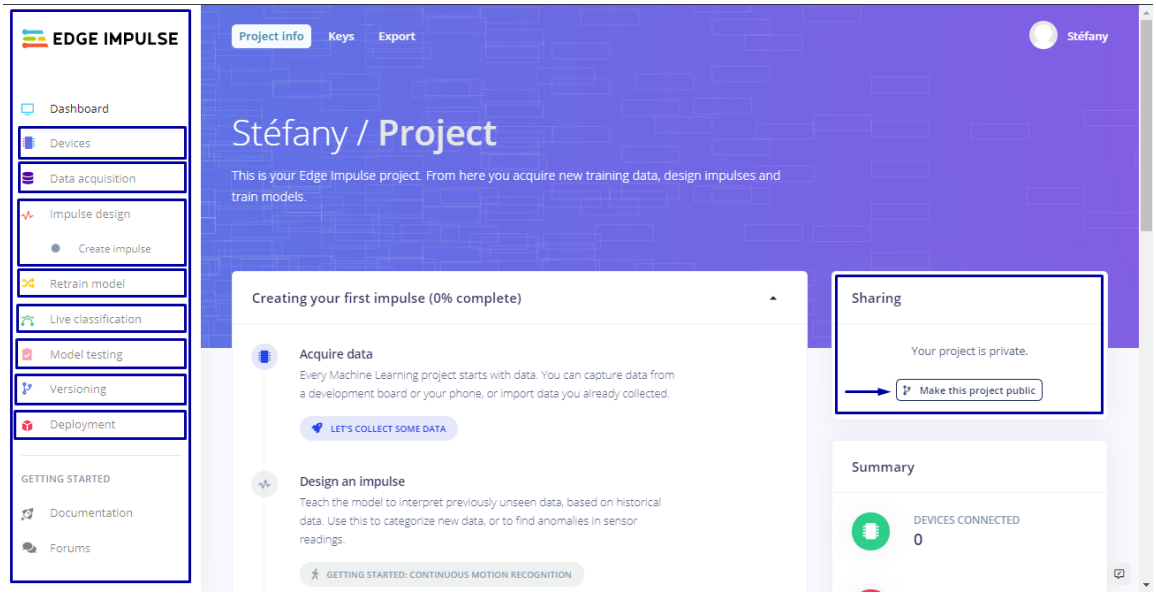


Figure 4.4: Tabs and Sharing
Fonte: Edge Impulse

4.4 Preprocessing

We founded some obstacles to work with the data, because we needed audios with .wav extension to upload on Edge Impulse and with the frequency of 16kHz to use the microphone on Arduino. To do that, we used a project, which contains a GitHub with a notebook for data pre-processing and model training as well, that we based on to make a function that transform the high frequency into the frequency we wanted. In addition, we had to separate the dataset that had audios without the label names and were in a separate .json file for each.

The link to access it: `https://stanford-cs329s.github.io/reports/virufy-on-device-detection-for-covid-19/`.

The link for the GitHub of the Virufy Cough: `https://github.com/dtch1997/virufy-tm-cough-app`.

The code with all steps is on Google Colab and could be viewed with Jupyter Notebook too: `https://colab.research.google.com/drive/14Jma2DLTEEopEU7A26aLDInvVkzGpM-C?usp=sharing`.

We explored the data a little bit and we created the separated directory to "healthy" and "covid-19":

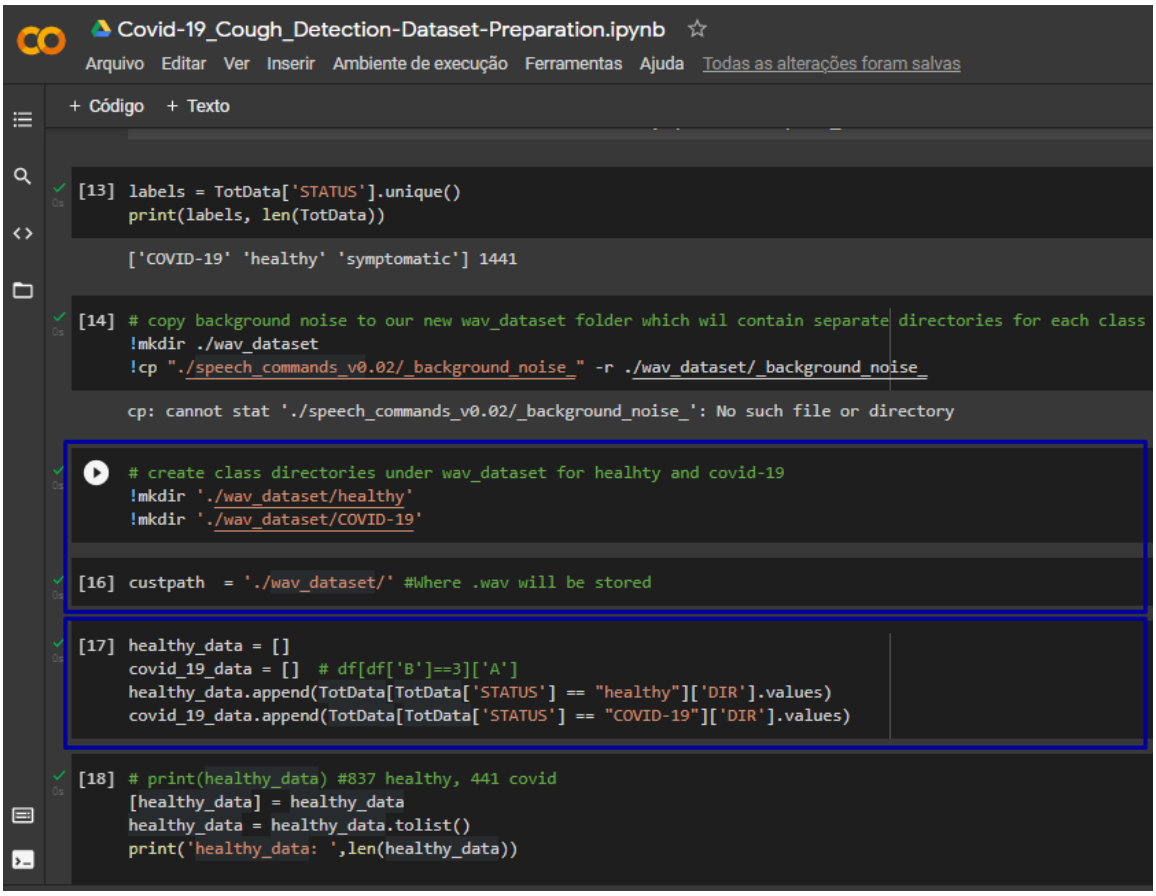


Figure 4.5: Separating Directories
Fonte: Google CoLab

We also created, based on Prof. Marcelo code, a function that transform the files format and the frequency, using the librosa library.

```
def wav_PCM16_extractor(row):
    name = row[0]
    diag = row[-3]
    try:
        audio,sr = librosa.load(row[-2], sr=16000)
        savepath = os.path.join(custpath, diag, name+'.wav')
        soundfile.write(savepath, audio, sr, subtype='PCM_16')
    except:
        print('File cannot open')
        return None,None
    return savepath
```

```
[36] for row in TotData.values:
        savepath  = wav_PCM16_extractor(row)
```

Figure 4.6: Converting Audios from .webm and .ogg to .wav and the setting the frequency of 16kHz.
Fonte: Google CoLab

4.5 Model Design

4.5.1 Uploading Data

After preprocess the data, we could upload it into Edge Impulse, because the file had the right extension and already were in the correct pasts.

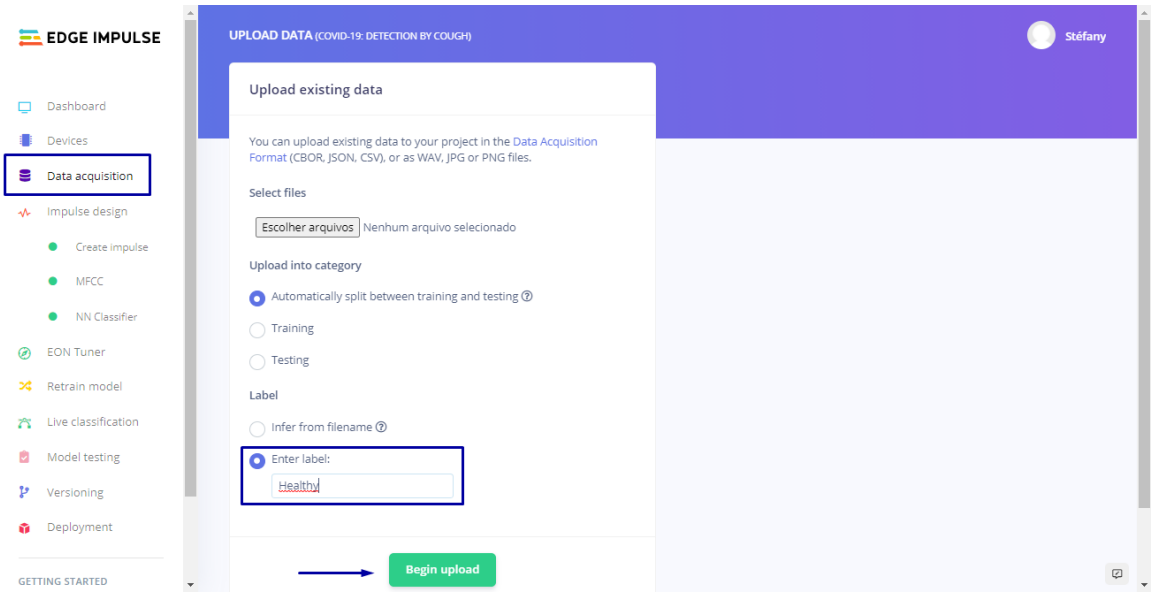


Figure 4.7: Uploading Data
Fonte: Edge Impulse

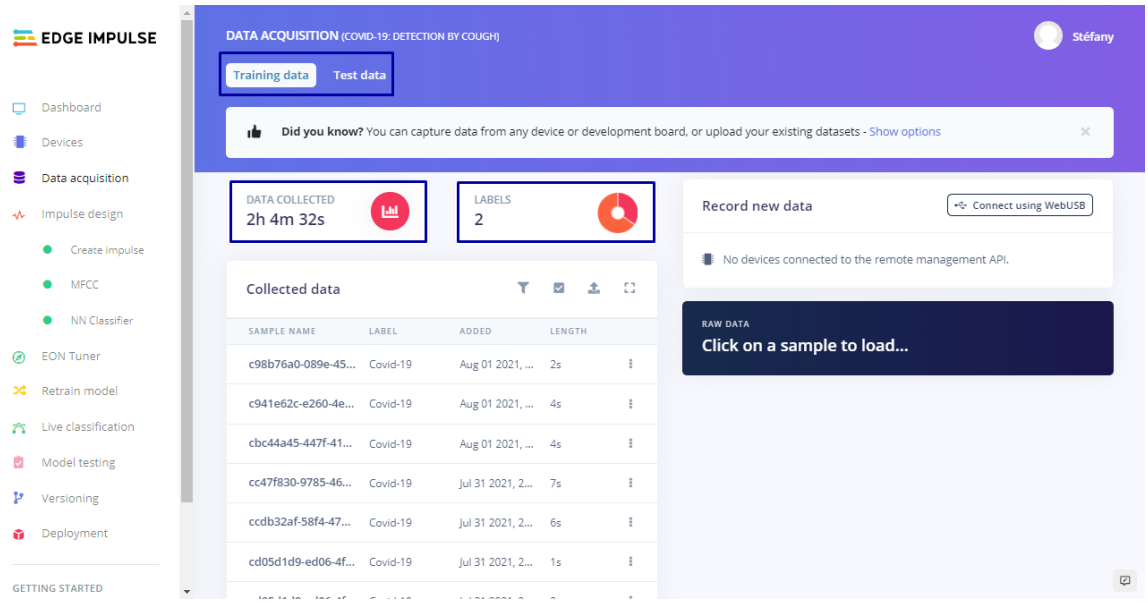


Figure 4.8: Training Data
Fonte: Edge Impulse

Look the two audios in the time domain and see too that the data have blank spaces between cough and some background too, what shows

the diversity of environment we have.

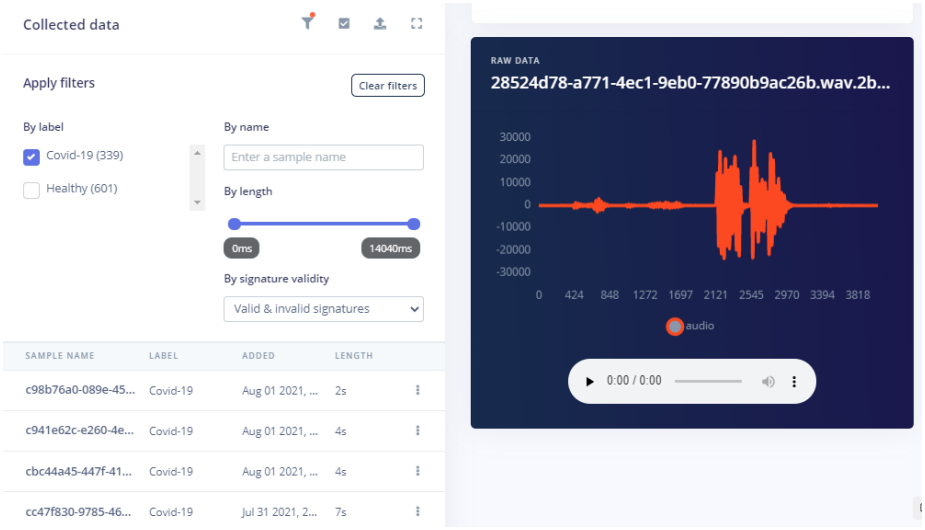


Figure 4.9: Sample: Covid-19
Fonte: Edge Impulse

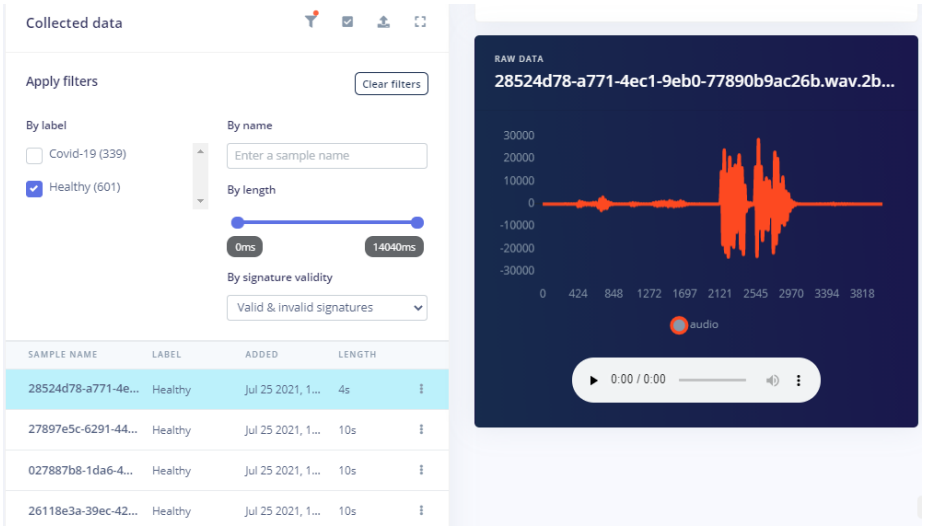


Figure 4.10: Sample: Healthy
Fonte: Edge Impulse

4.5.2 Creating an Impulse

Then, we created an impulse. This part were made a lot of times to get the better model and an interesting fact was that we tried with different windows size to the "Time series data" and the best one was with 4000ms. To work with audios, we needed to choose MFCC, which extracts features

from audio signals using Mel Frequency Cepstral Coefficients. We also did some tests with MFE, because of the cough don't be a speak and the last possibility was to mix MFCC and MFE and another interesting fact: the model had a pretty awesome result, very close of 90%.

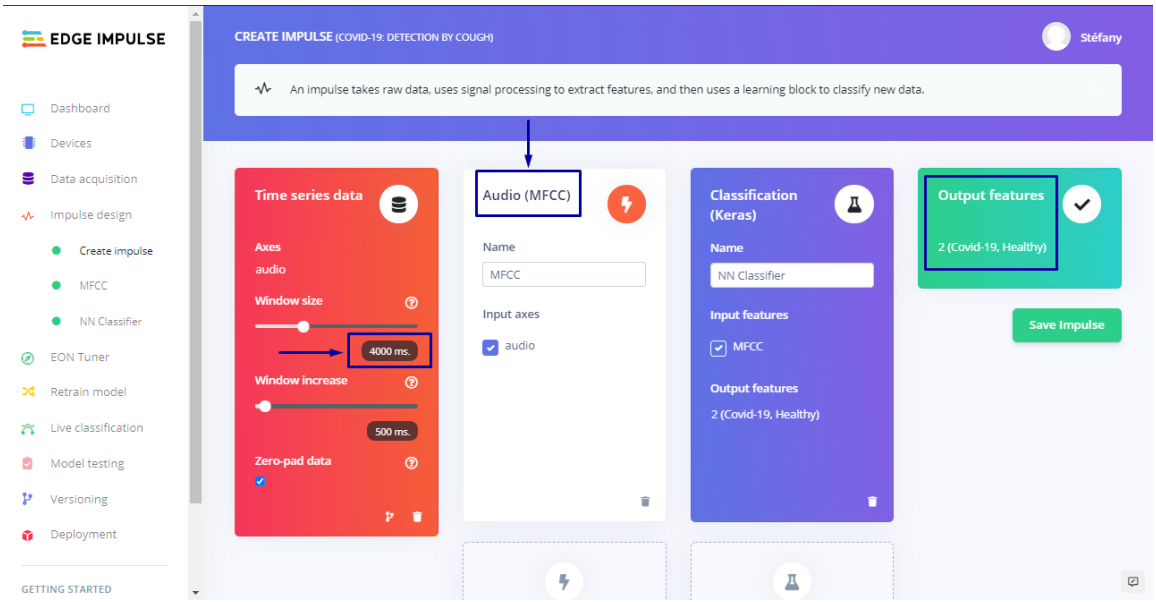


Figure 4.11: Creating an Impulse
Fonte: Edge Impulse

4.5.3 Filter MFCC

The next step in the final version until now was to set somethings on MFCC's tab. We change the number of coefficients from 13 (pattern) to 40, because of the lot of tests we used to identify the better option to us. Here, the features were extracted to enter in the model with the convolutional layers later. This part convert into a scale of MEL frequency, that is an espectrum of power of a sound. When MFE tab, we also used 40 to the coefficients and we setted the high frequency as zero. Then we generated the features.

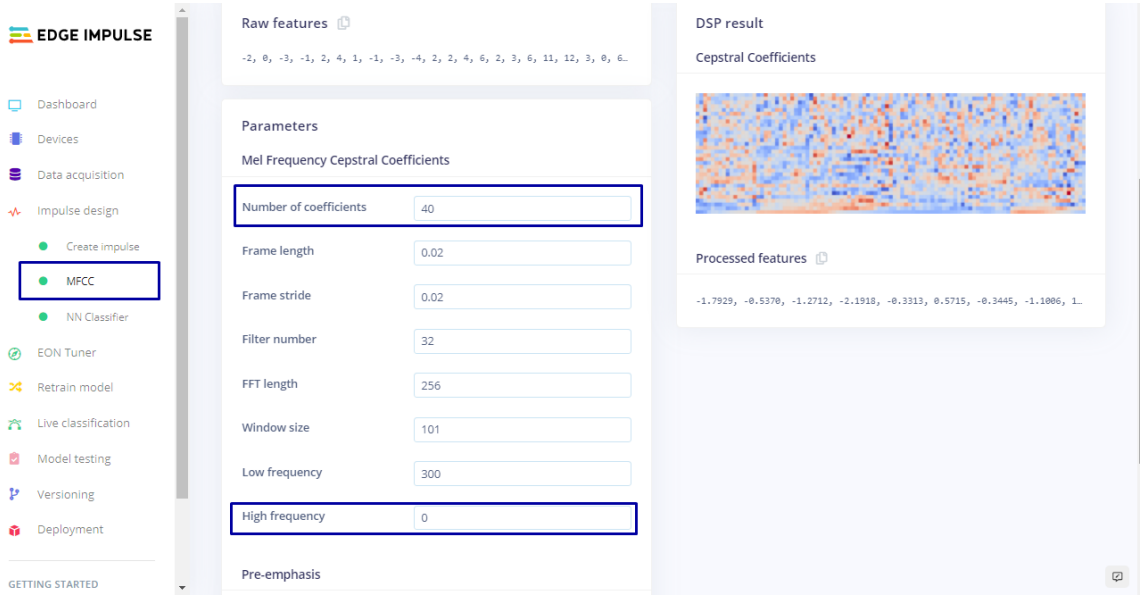


Figure 4.12: Filter MFCC
Fonte: Edge Impulse

We can also see the model closer and how we have to treat the data before they enter into the model:

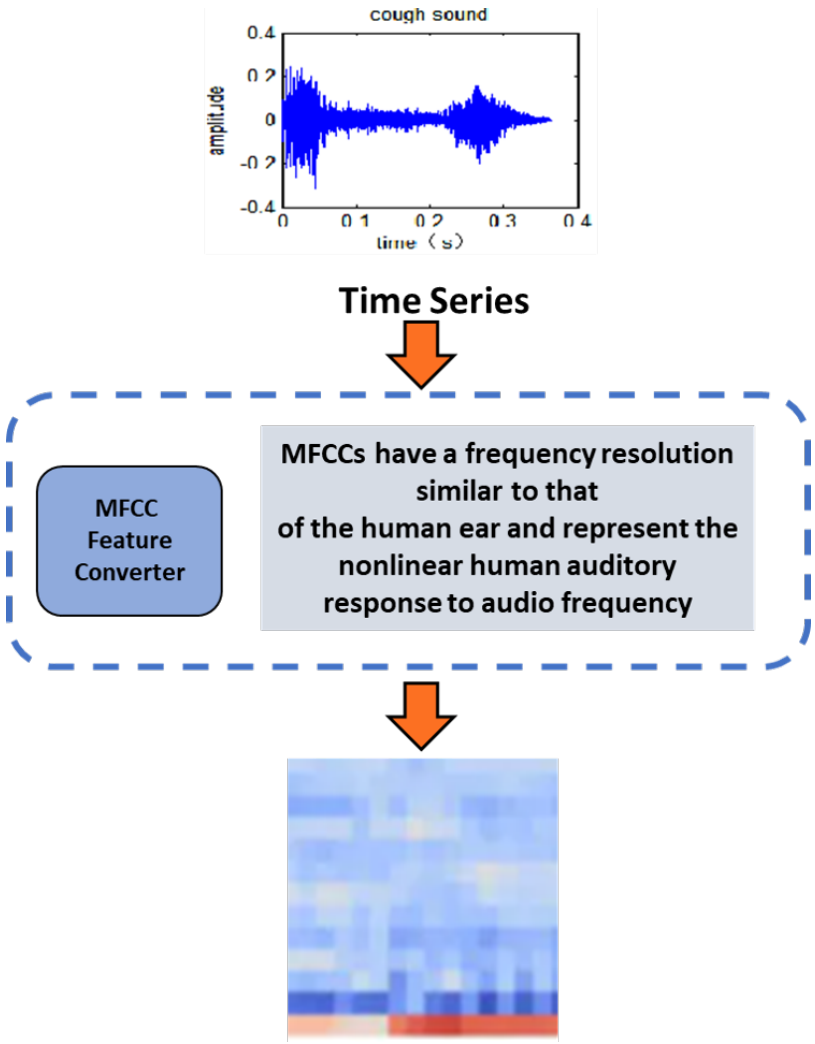


Figure 4.13: Block Diagram Showing the Pre-process of the data
Fonte: Build image constructed based on report: `https://bit.ly/3zwAgkS`

The audio enters on time domain and we need to transform it to frequency domain to use CNN to construct the model.

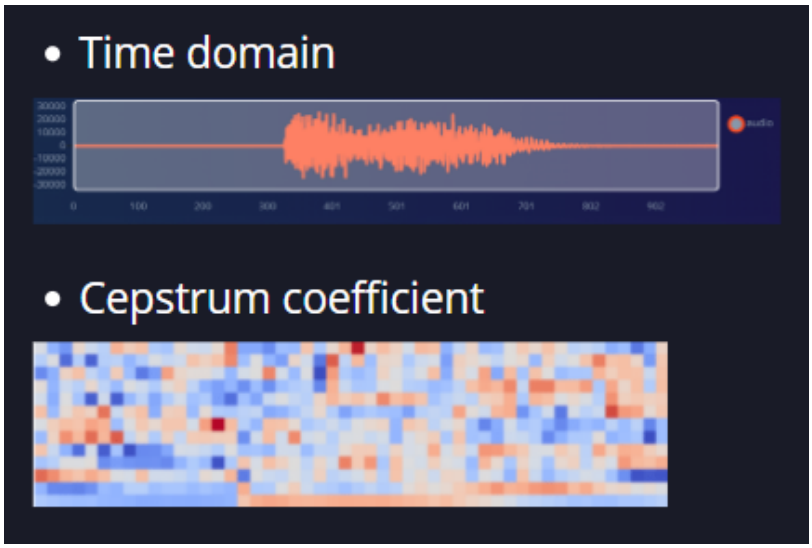


Figure 4.14: A Sample (Input and After Applying MFCC
Edge Impulse

How about the features generated? See:

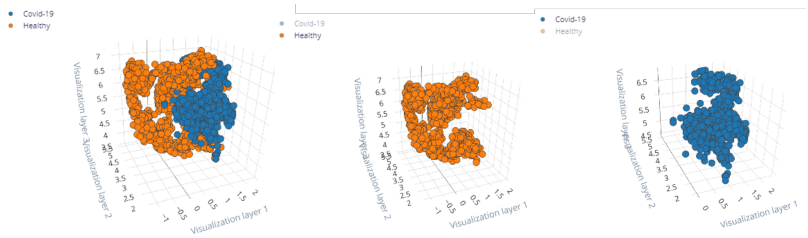


Figure 4.15: Feature Explorer
Edge Impulse

4.5.4 Hyperparameters

After doing all of the steps mentioned before, it was time to define our model. We started setting the number of epochs necessary to make the training step. We choosed 100 and we saw that was a good number, because we did make some modifications always thinking to improve our model. After a good guess in the learning rate as 0.005, we also tried to change it for 0.0005 and 0.00005 and we noticed that was a bad idea, but here is the tip to make a good project: "try, try and try". Each application have some specifications and we need to find the best attend us by testing and testing again. That's how Machine Learning Flow works! We tried to use the SpecAugment too with the intuit of increase our dataset, but didn't behavior

weel, so we drop this idea. Bbut the Edge Impulse has this new functionality if you are interesting about it.

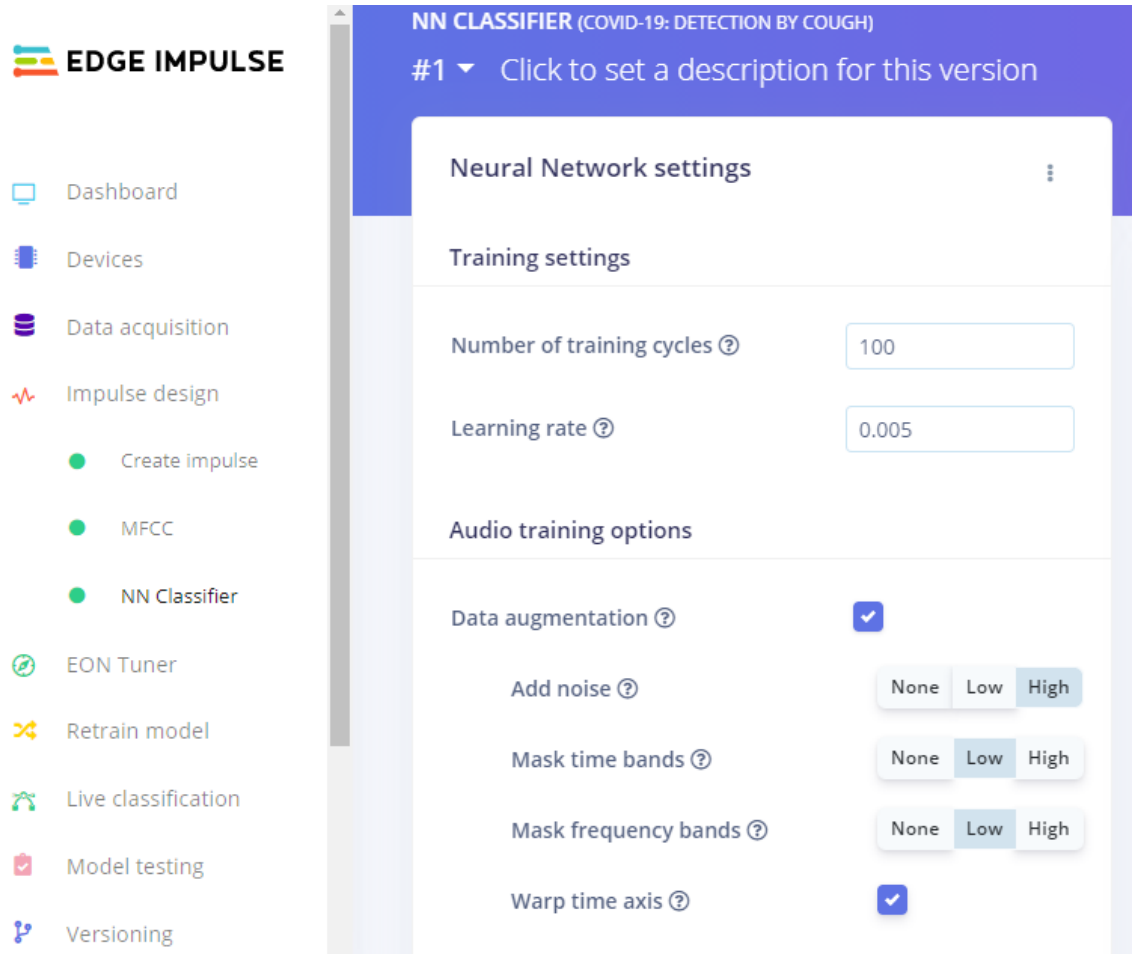


Figure 4.16: Hyperparameters
Fonte: Edge Impulse

4.5.5 Model Architeture

We also setted the quantity of layers, Dropout and we have an output with 2 features because of our classification. We didn't need a robust model to this kind, but we based on the project of Edge Impulse to work with elephants to see the step-by-step. That way, we configured a neural network that has, in order: a reshape to the 40 colums because of the number of coefficientes we used previously, one convolutional layer 1D with 8 neurons(it worked efectivelly as 2D, but consumes less memory) and max pooling, Dropout of 0.2 (20% randomly are cutted of in that moment), another con-

volutional layer 1D with 16 neurons and max pooling, Dropout of 0.2 (20% randomly are cutted of in that moment), another convolutional layer 1D with 32 neurons and max pooling, Dropout of 0.2 (20% randomly are cutted of in that moment), Flatten Layer to finally walk in the output direction of the model. Note that we used the quantity of layers as powers of 2, based on what we know so far about other models and architetures.

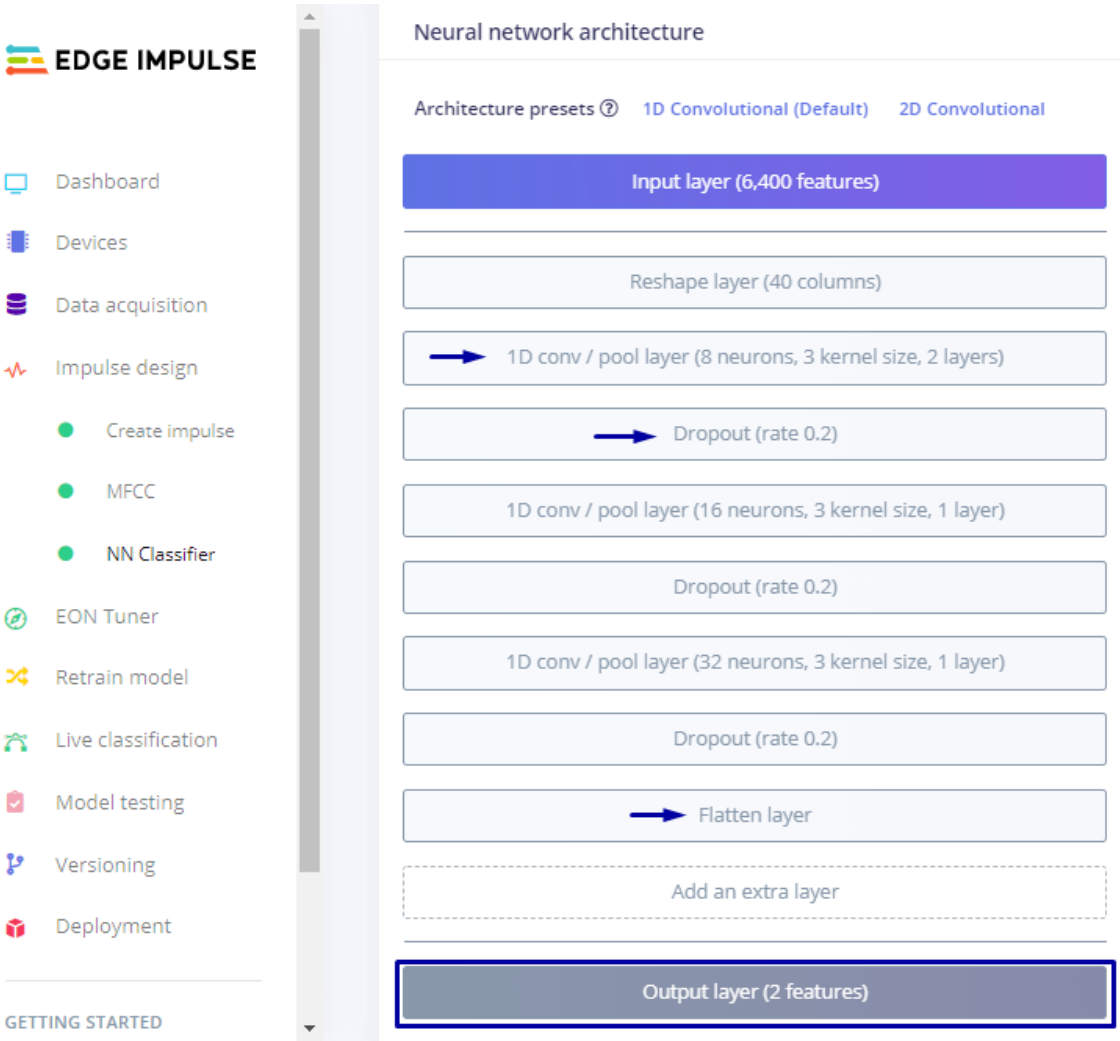


Figure 4.17: Model Architeture
Fonte: Edge Impulse

With all these configurations, we got 92.5%. It was the maximum percentage of the model counting all times and changes.

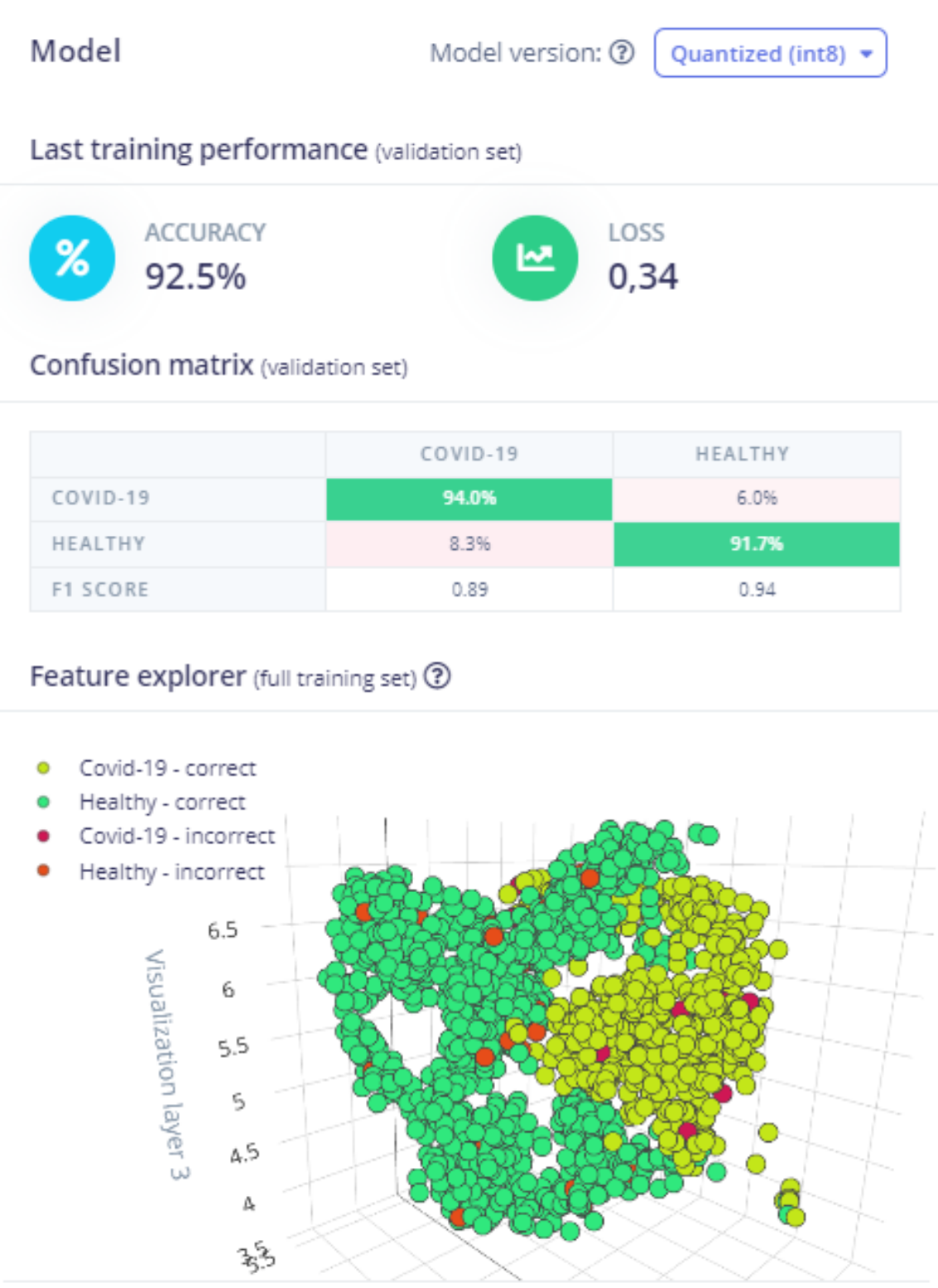


Figure 4.18: Validation Accuracy
Fonte: Edge Impulse

Look! There some wrong predictions, but the model until now it's not bad and the more important thing: Only 6.0%, less that for "Healthy", is got wrong to Covid-19. That's extremely crutial, because if our model gets a wrong prection is better tell the person she have covid-19 and she isn't that tell the person she don't have the disease, but actually she has. So, not bad, right? Well, let's go further. Before that, let us show you the behavior until now of the accuracy by the notebook:

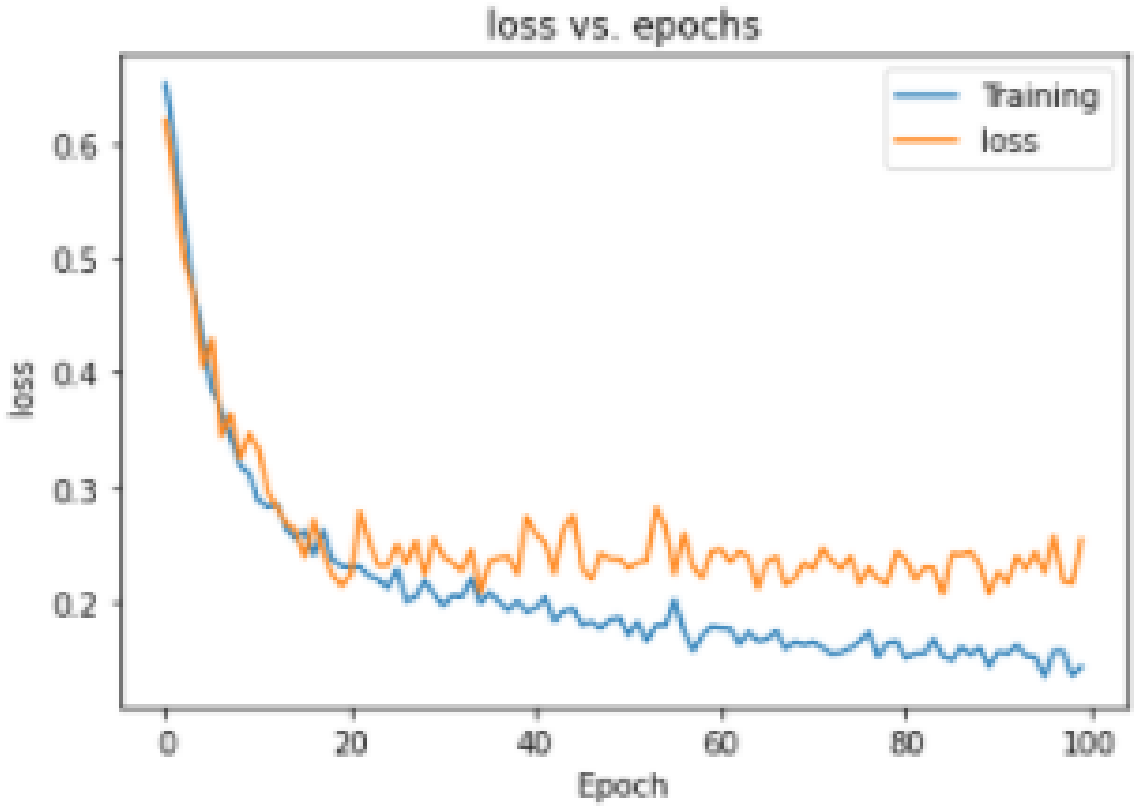


Figure 4.19: The Graph of Loss
Fonte: Google CoLab

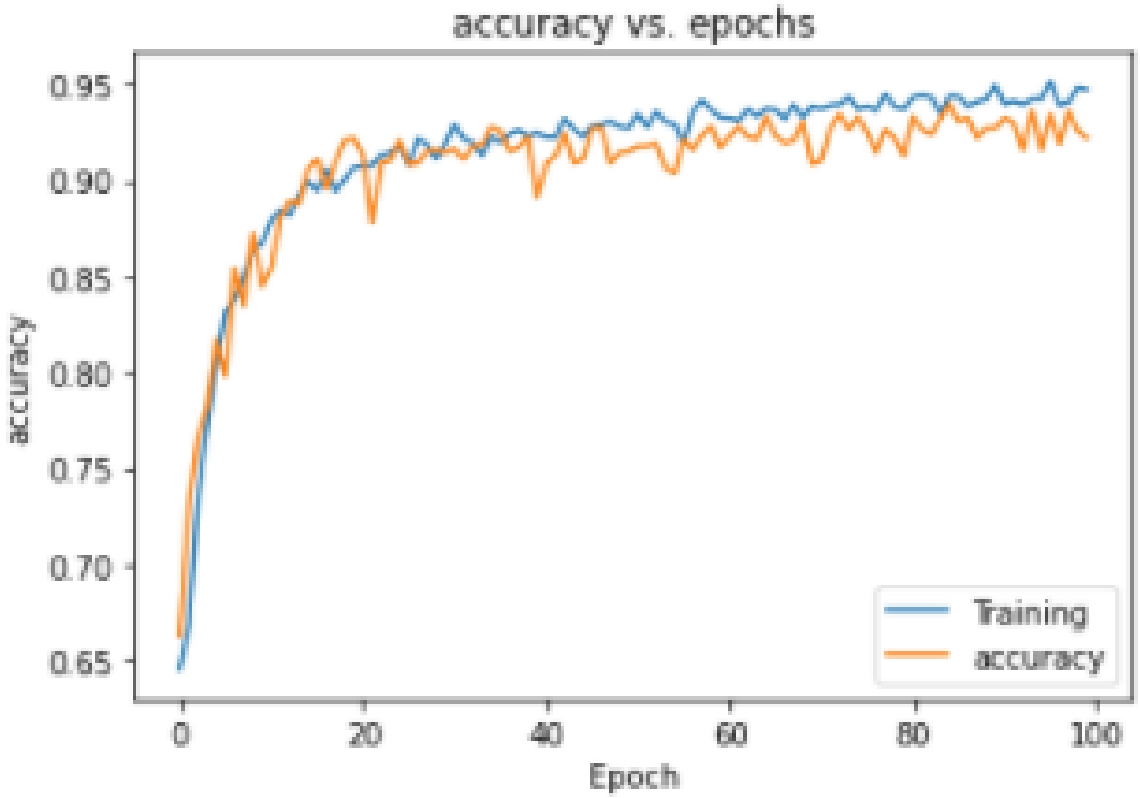


Figure 4.20: The Graph of Accuracy
Fonte: Google CoLab

If you want to see step-step of what I already told you and make some modification by your own, you can access the notebook generated by Edge Impulse and with some extra code to show the graphs and convert and save TF model to a tflite model:

`https://colab.research.google.com/drive/1GKqkcxS_qArbYYmhr7SpwJN9YXK4zyjJ?usp=sharing`

4.5.6 Live Classification and Model Testing

On the "Live Classification" tab, we can load an audio and see how the model responds. There's an example below:

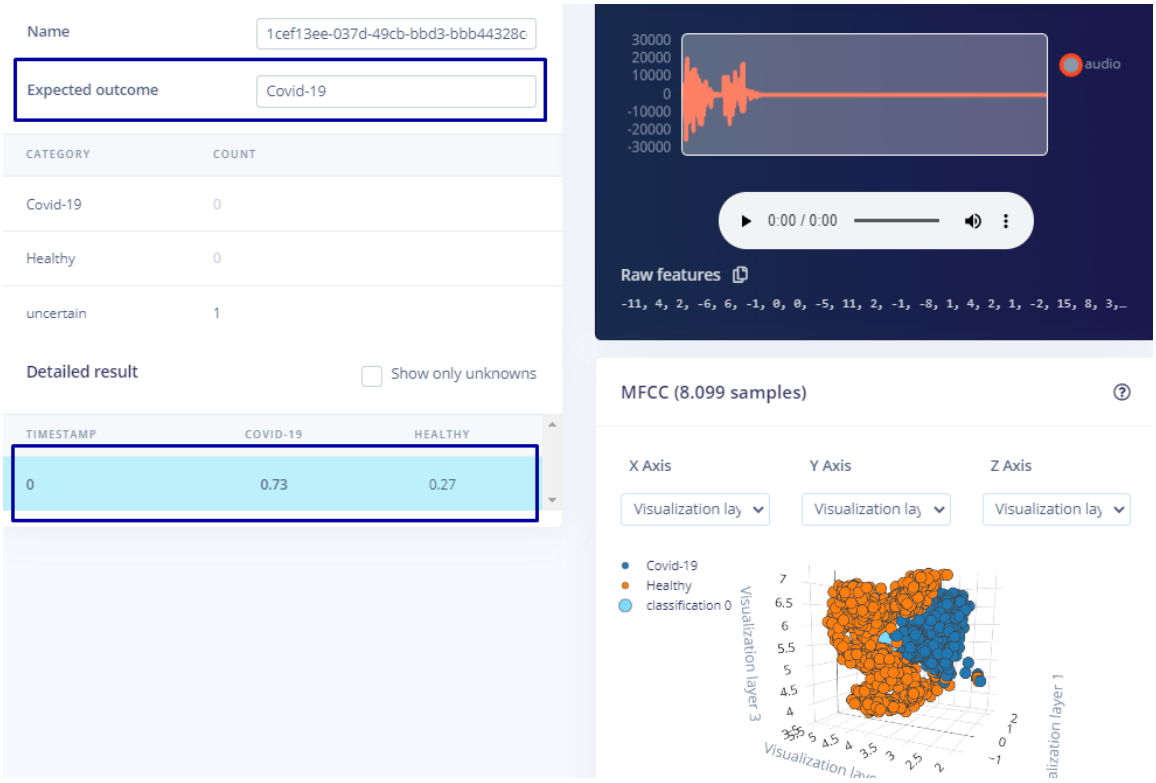


Figure 4.21: Example of Live Classification
Fonte: Edge Impulse

You think the model is great until now, but let me show you what happens in this step. The accuracy went down! But what happened? Well, the model don't respond correctly to new data and that's probably because of the differences of training data and testing data. We tried to work with that and to do a lot of modifications and the last one was upload again the data, with the same model and the validation accuracy and model testing accuracy could be seen. The first one increased:

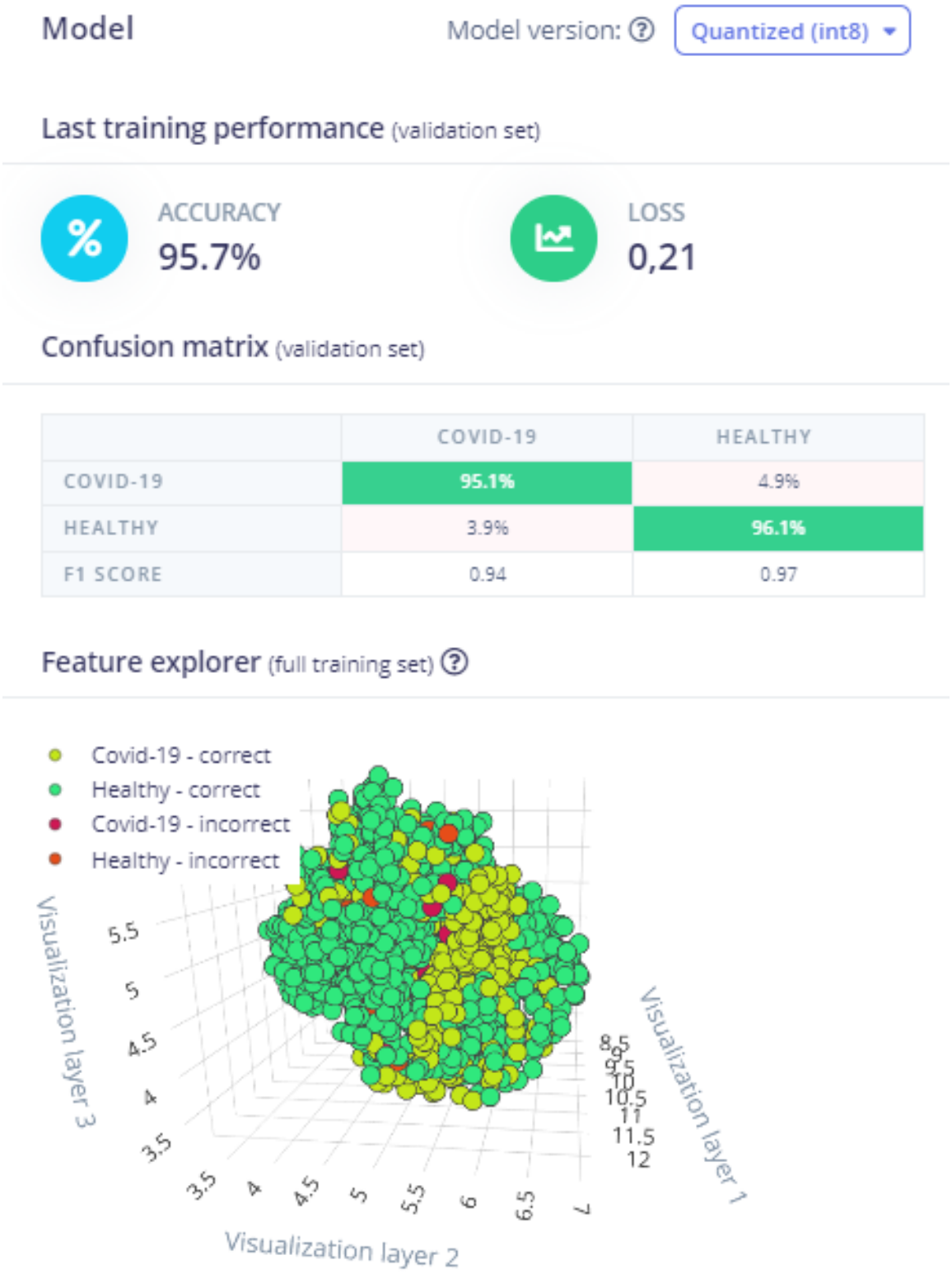


Figure 4.22: Validation Accuracy with Data Being Uploaded Again
Fonte: Edge Impulse

How about the "Model Testing"? The accuracy isn't close to be good and we used a lot of ways to avoid that by incresing and decreasing the "Windows Size", the number of coefficients, split the samples into a lot of another samples, carrying only the cough part (that's actually make the model be a little bit better, but not enough) and even the recent function of Edge Impulse, the EON Tuner. Believe when we saying that we tried a lot of things, but with more time and paying more attention in our data, the continuing of the project could assume new frontiers.

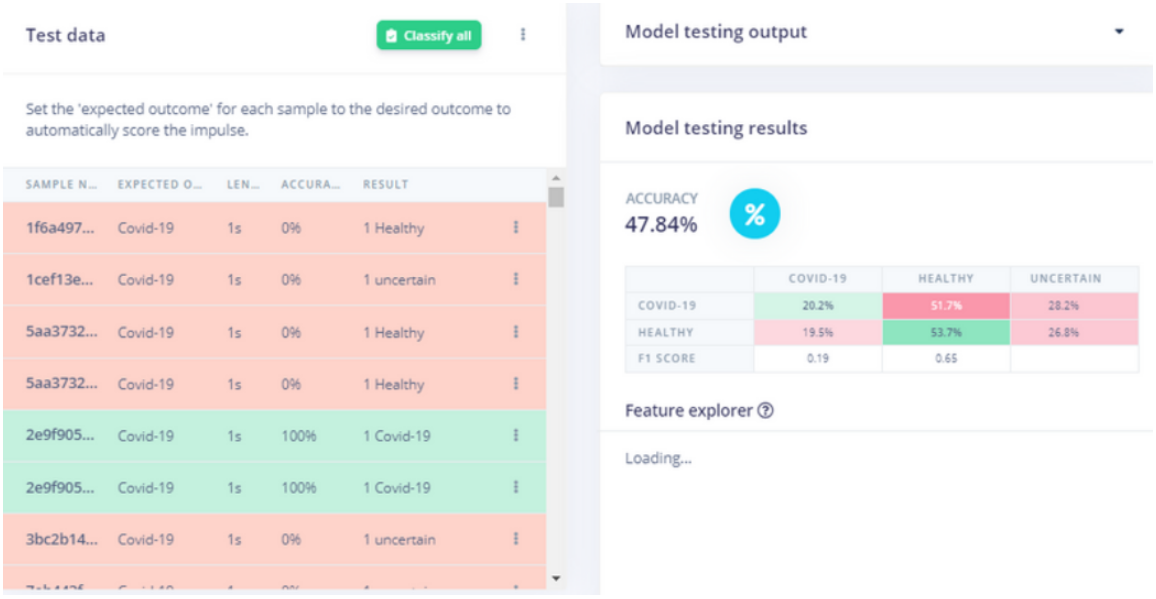


Figure 4.23: Model Testing
Fonte: Edge Impulse

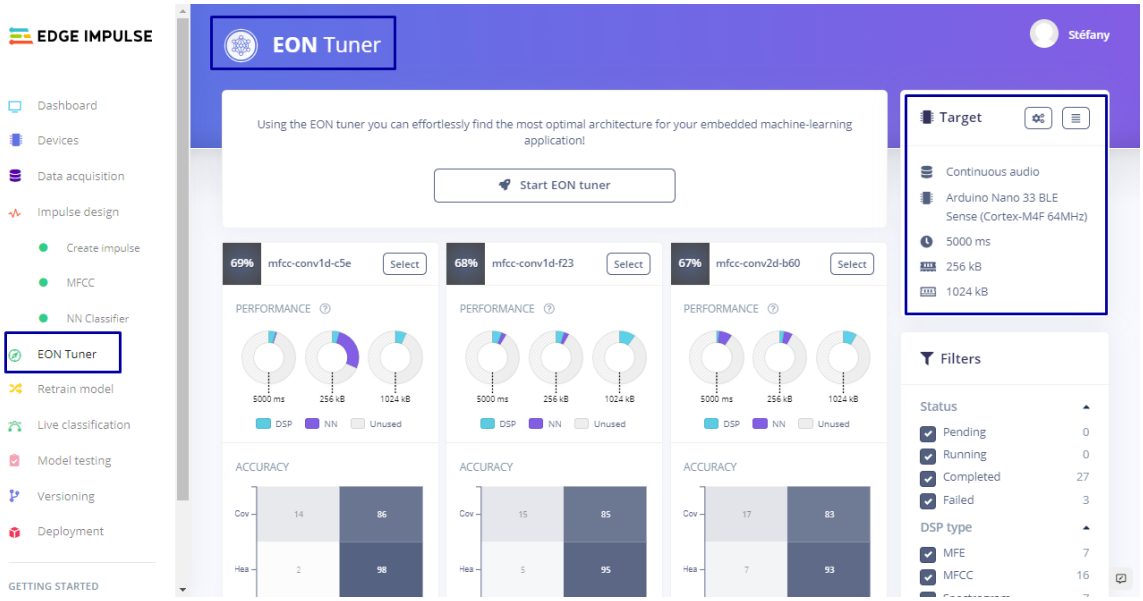


Figure 4.24: EON Tuner
Fonte: Edge Impulse

The percentage of EON Tuner didn't get some great results as we expected!

The diagram model we wanted to achieve is showing below and even with the results we presented to you here, we created a solid base on the step-by-step project and there is new ways to continue after that.

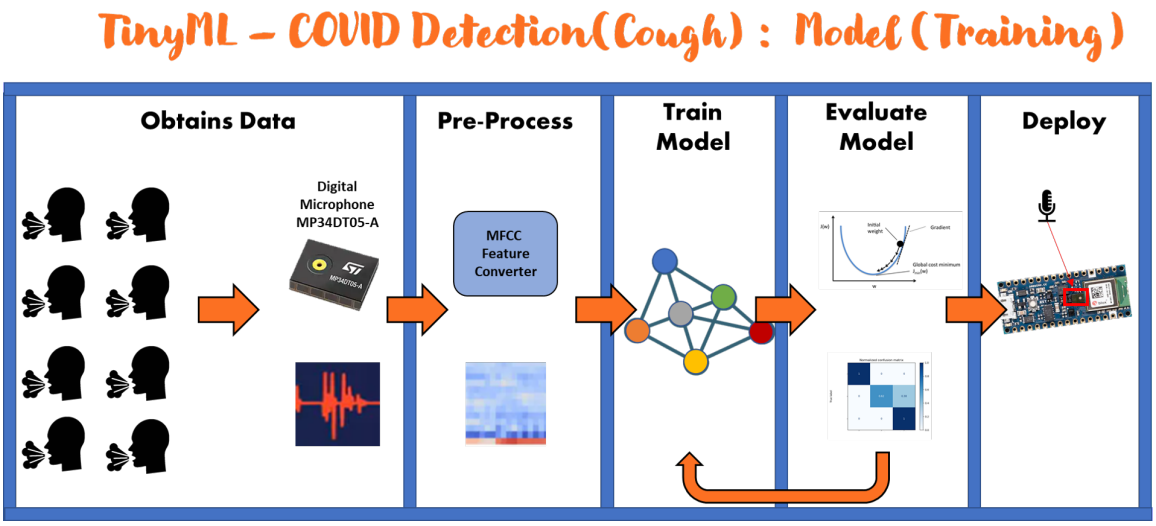


Figure 4.25: Block Diagram Showing the Model Training
Fonte: Build image based on presentations made by Prof. Marcelo Rovai

4.6 Model Inference

We planned to deploy the model into Arduino, but we couldn't, because as we shown to you, the model didn't get a good performance. Still, we tried to use the smartphone to see the behavior of the model in action and we realized that the model confused with a cough of same person and this is a problem. Besides that, the accuracy in each sample is far from realiable and for that reasons, is not a real deployment. Below there is the diagram expected to be accomplished to this project. The audio should be captured by a microphone, transformed into spectogram (an image), then It will enter on the model created and assume: Positive to Covid-19 (More than 80%) and Negative to Covid-19 (Less than 80%). The inference would use LEDs: Red (Positive Result to Covid-19), Green (Negative Result to Covid-19). See this diagram below:

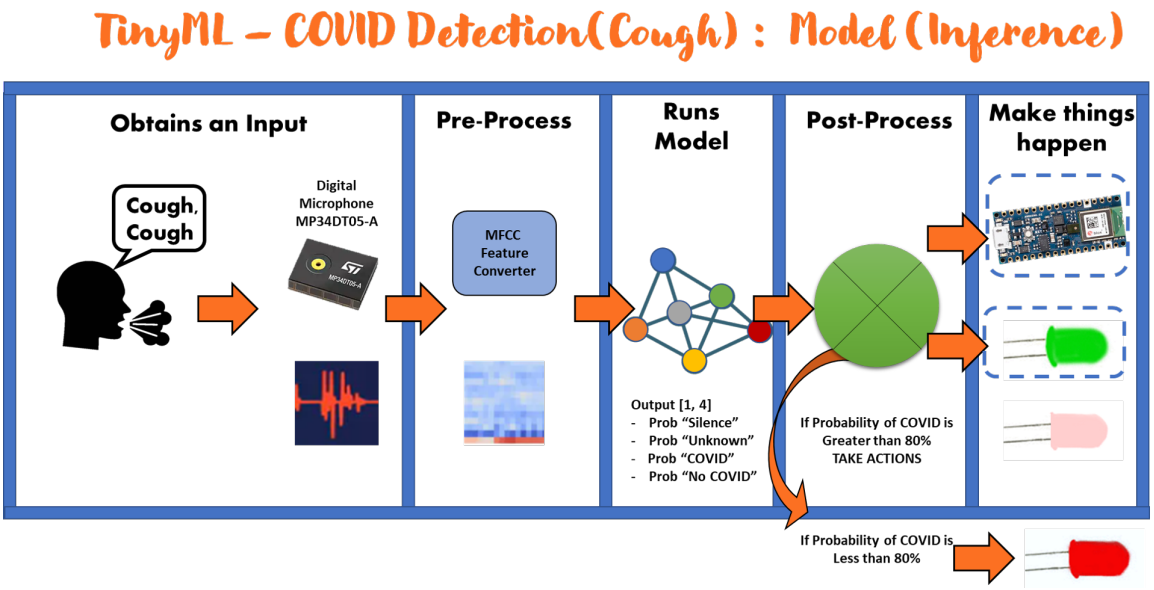


Figure 4.26: Block Diagram Showing the Inference
Fonte: Build image based on presentations made by Prof. Marcelo Rovai

What we actually did was to download the Arduino Library from Edge Impulse and add the Led's. We also looked to the final memory used, that is less that Arduino has. The latenncy isn't bad to the Optimized option.

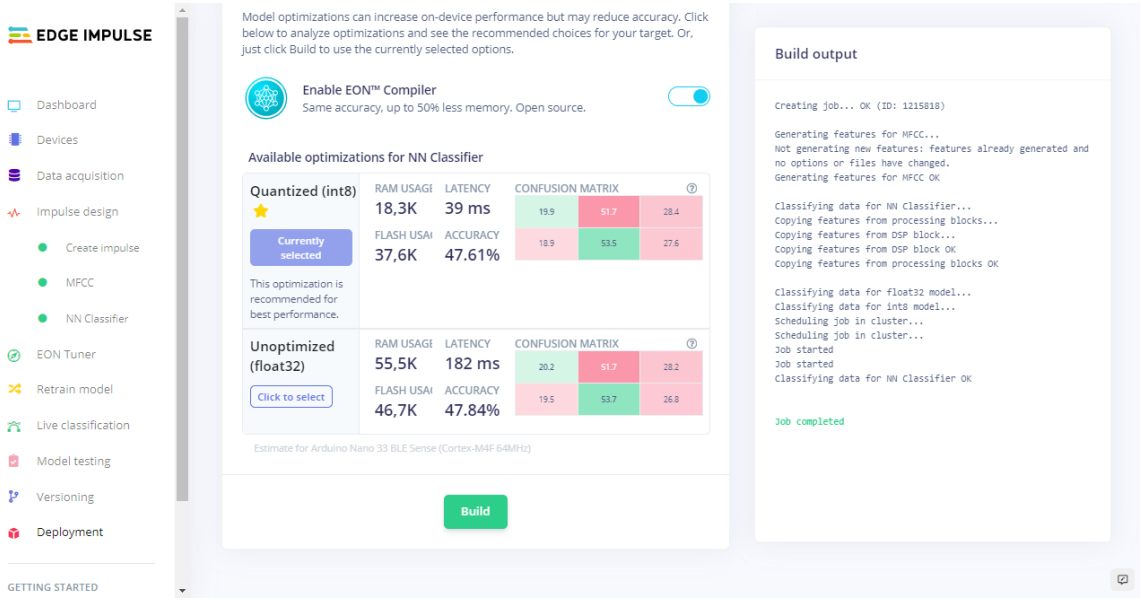


Figure 4.27: Deployment Tab
Fonte: Edge Impulse

Here is the Arduino IDE with the code:

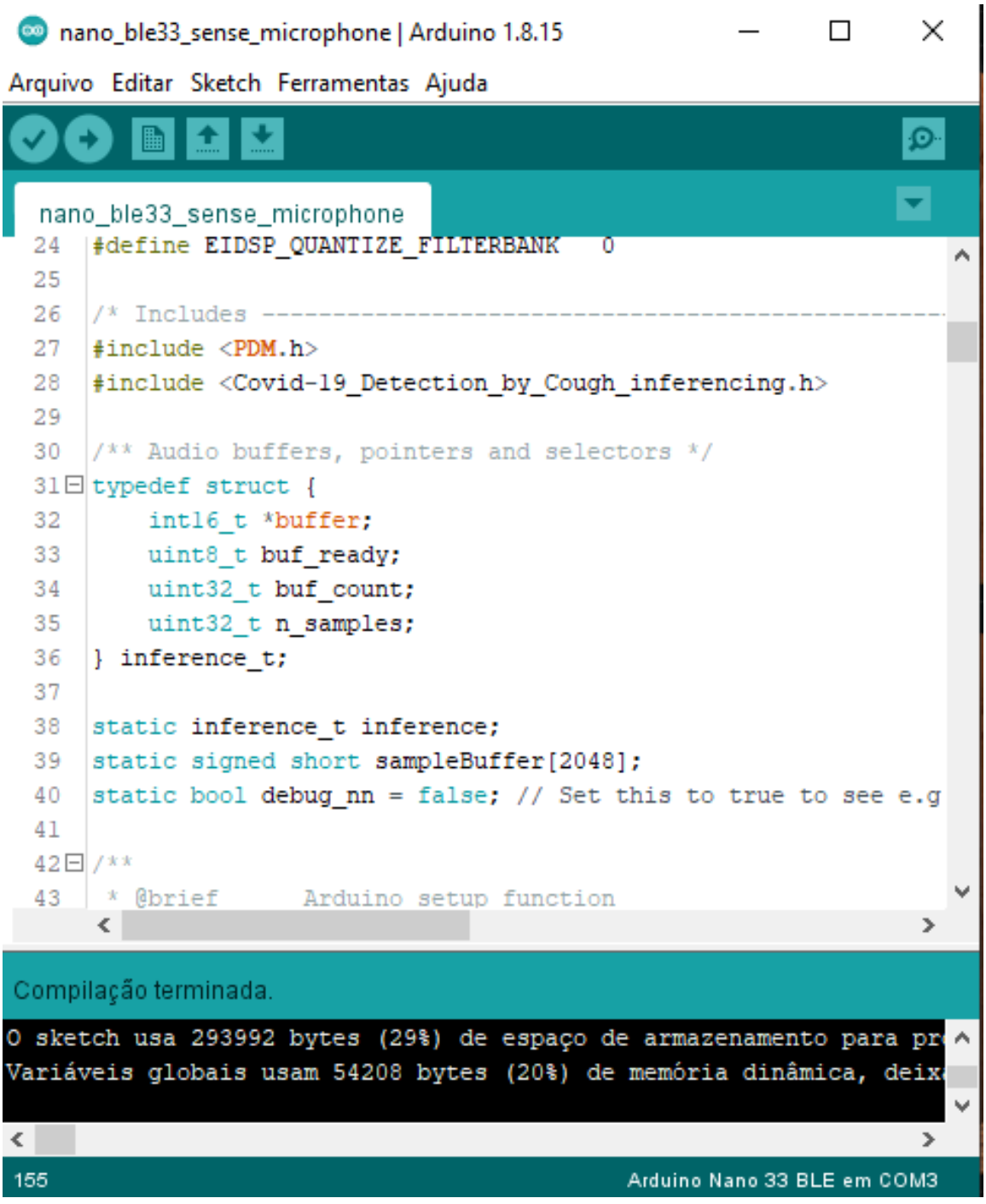


Figure 4.28: A Taste of the Deployment
Fonte: Arduino IDE

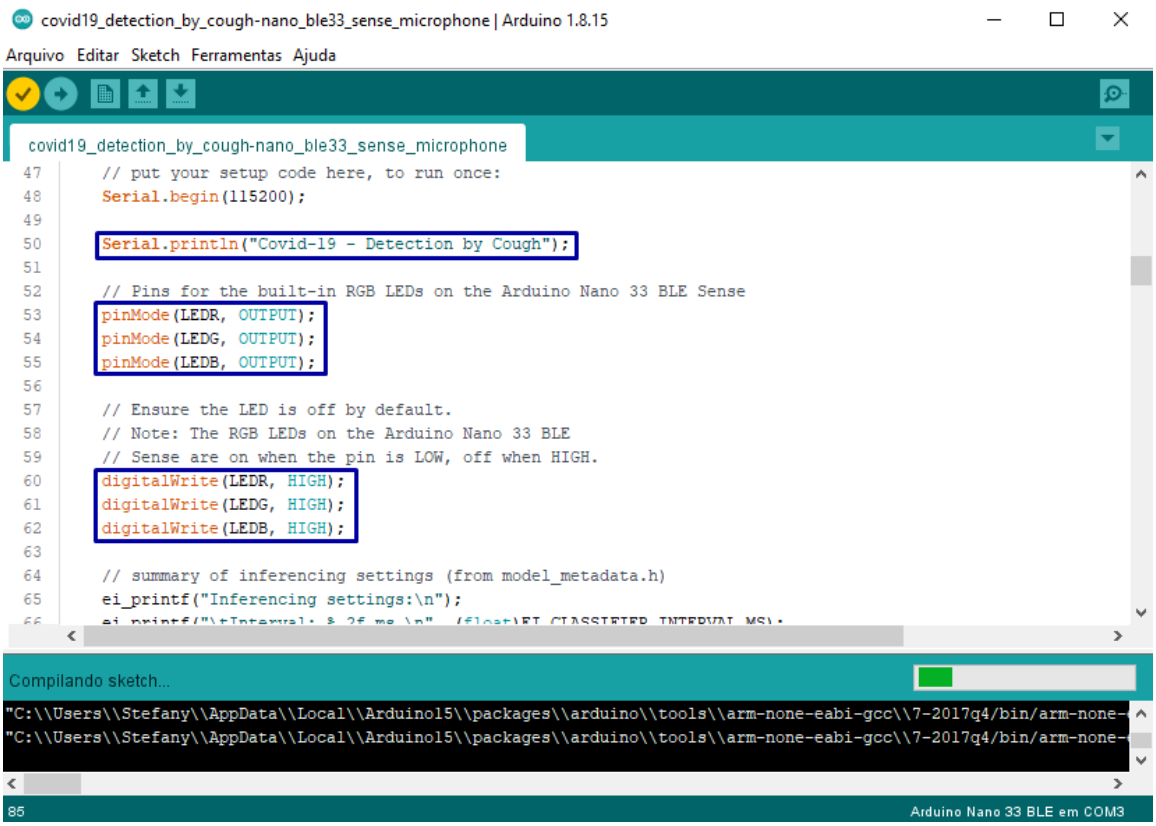
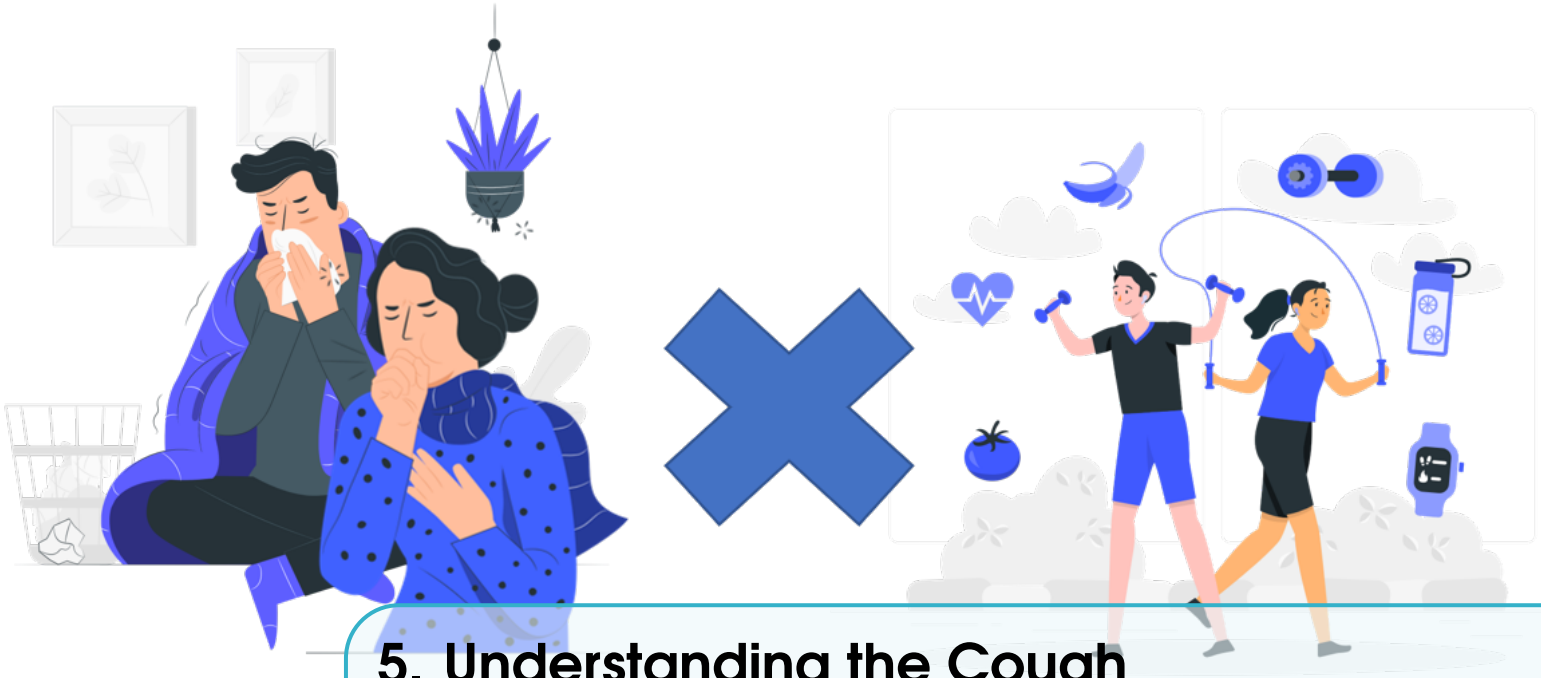


Figure 4.29: Part of Print and LED's
Fonte: Arduino IDE



5. Understanding the Cough

5.1 Types of cough

If we are going to use cough to recognize covid-19 the first thing we need to do it's to understand the types of cough and the difference between them, according to Cohen-McFarlane et al. (2020), the cough should be separate between two classes the wet cough and the dry cough, the first one normally it's produced by stranger bodies in ours organisms and comes with secretion, the second one can be produced by numerous causes and doesn't present any secretion, with this in mind we can now think where does the cough produced by covid 19 fits? Well kind of in both cases, because in the early stage of the disease we have a strong predominance of the dry cough and with the development of the disease the characteristics of a wet cough becomes more predominant.

5.2 Phases of cough

Another important point is that according with Cohen-McFarlane et al. (2020) we have three important phases of the cough (1) initial cough sound, (2) intermediate phase, and (3) second cough sound.

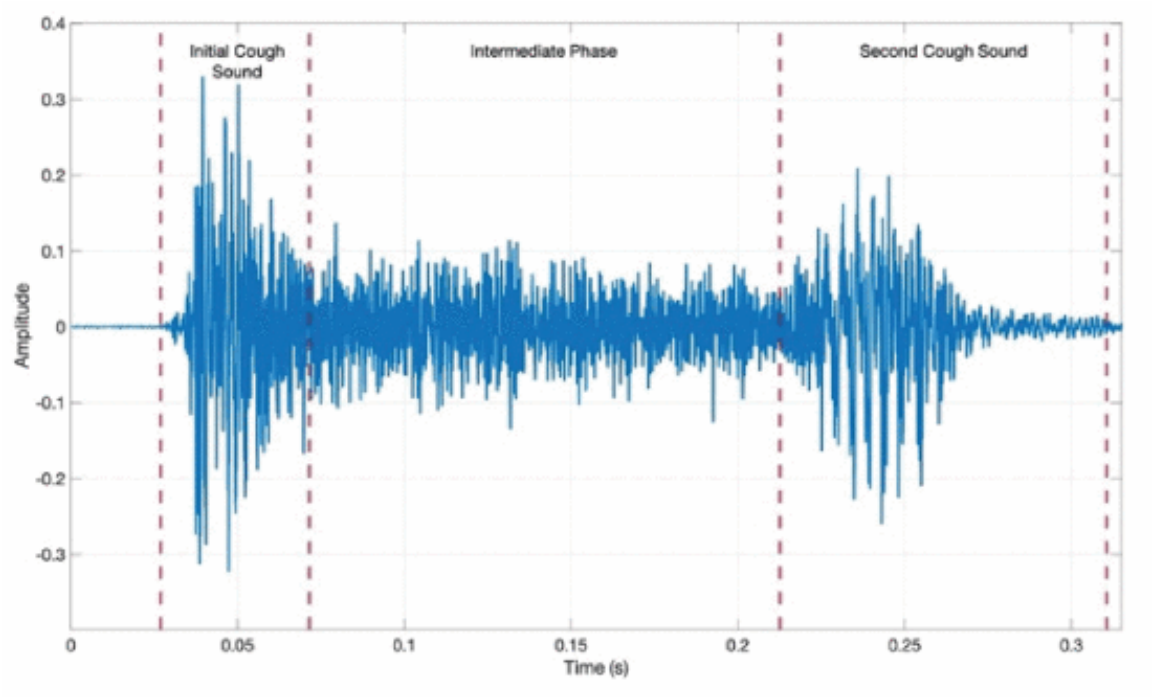
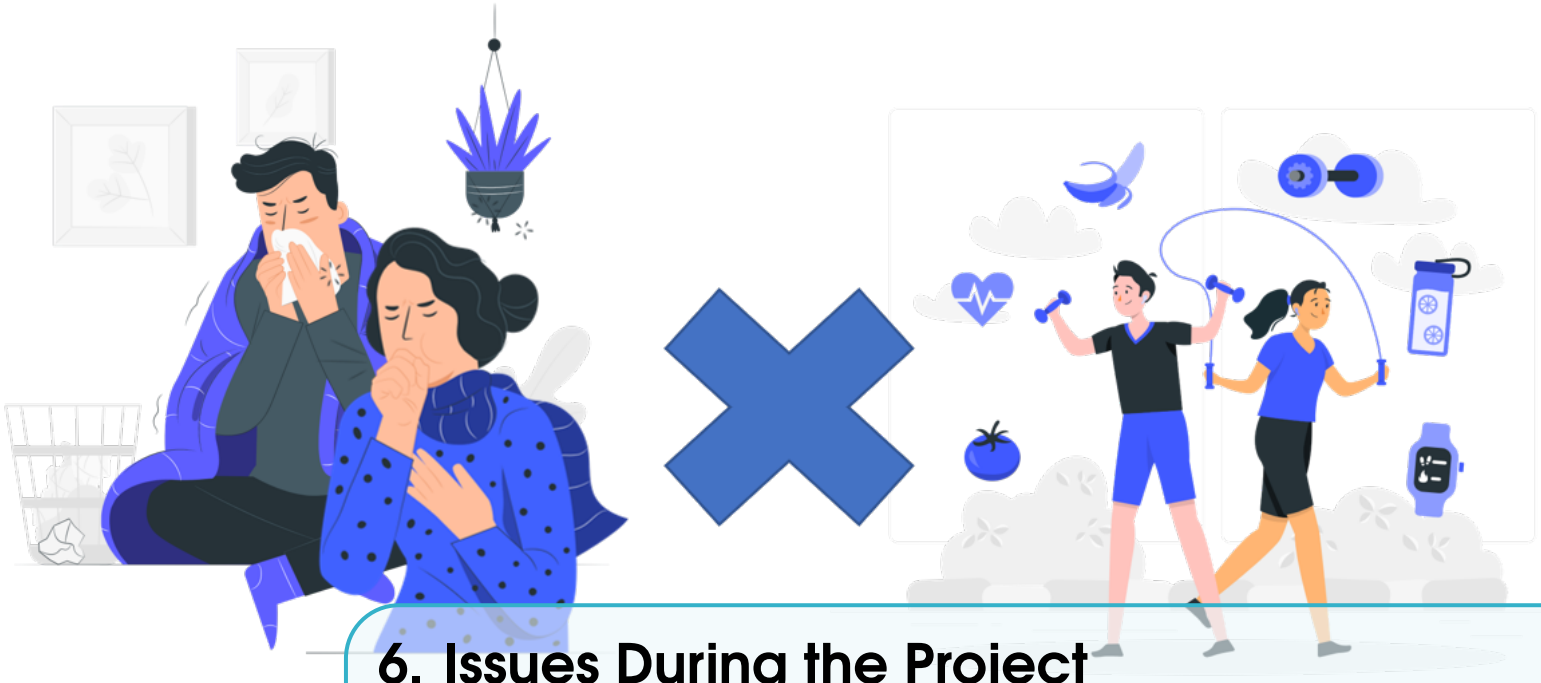


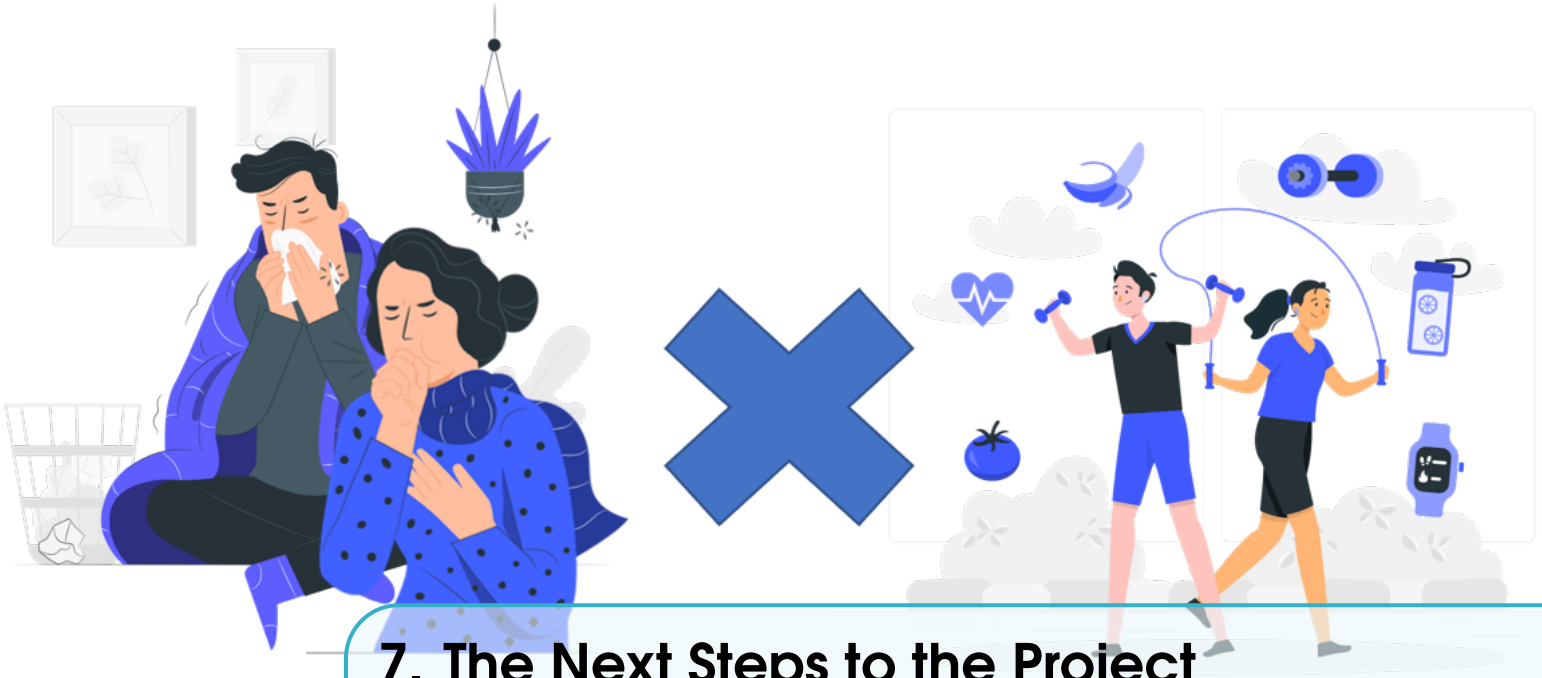
Figure 5.1: Phases of cough
Fonte: Cohen-McFarlane et al. (2020)



6. Issues During the Project

One of the problems related to the dataset described by nature itself is the reliability of the dataset, in this situation of pandemic becomes really dificult to overload all health professionals who are already working in extreme situations, to ask them to record cough of covid patients, because of this the dataset has to really on the contribution of volunteers to capture the data, and here we have a lot of problems associate since the reliability of the data, to the devices used to record the data, where in each one of this devices we have a different sample rate and different charateristics, other problem related to te dataset is the labels that are not separate as the Cohen-McFarlane et al. (2020) indicates, using for labels: wet cought, dry cought, inital covid state and end covid state, here in this dataset we only have two labels: covid cough and cough healthy.

We also had ploblems due the different frequency of the data. We lost a lot of time trying to deal with that and searching solutions and this could be avoidable by picking a dataset with all real data, but already shows the real fact of collecting data. It's not easy to follow a pattern of colletions, principally because we wanted different people on different situations.



7. The Next Steps to the Project

With the end of our discussion, there are still some hypothesis that need to be answere and could be tested to future steps of the project.

First, we can put one more label called "Unknown" to be activated on certain probability values showing the model doens't know if it's a healthy person or not.

Another hypothesis to be tested is if we change the classification to: "Dry Cough, Wet Cough, Cough with COVID" and analyze which behavior is obtained we could improve our final model.

Finally, we can try to reclassify all the training data to try differentiate the cough stages with COVID and without COVID. So, looking for the model to know better all stages of the disease.

Until now and to the next steps we need to consider the metrics: the application of the model (in this case is just for academic studies and not to use in the real medical field; avoid potencial bias comes to the leak of age groups or gender groups, for example; the tradeoff of latency and accuracy, user experience, memory size and all to make the model balanced on thosemetrics.

8. Contribuiting to this Development

We use python code to pre-process the input data and add it to Edge Impulse Studio. So, on this next link, we make available the Jupyter Notebook which contains everything we use for this purpose: https://colab.research.google.com/drive/14Jma2DLTEEopEU7A26aLDInvVkzGpM-C?usp=sharing.

We'll keep this project public and activated for any branches. The link for this project is at: https://studio.edgeimpulse.com/studio/40318.

All the Notebook code generated by Edge Impulse, along with the modifications to show some of the graphics present here, is available at:https://colab.research.google.com/drive/1GKqkcxS_qArbYYmhr7SpwJN9YXK4zyjJ?usp=sharing

We also gave an overview of everything we thought and validated throughout development. Our presentation is recorded and available at: https://www.youtube.com/watch?v=7Lca8lfdKAI.

We created a drive to upload the notebooks, arduino library and the dataset already separated like we mentioned. The link is: https://drive.google.com/drive/folders/1oH3fNWjkqxmpXEN7eLCOaNODulRhulhx?usp=sharing.

Finally, if you need to contact us and wants to contribute, feel free to send a message to the email: covidcoughdetection@gmail.com.

9. Bibliography

MOUAWAD, Pauline; DUBNOV, · Tammuz; DUBNOV, Shlomo. Robust Detection of COVID-19 in Cough Sounds: using recurrence dynamics and variable markov model. **Sn Computer Science**, Switzerland, v. 2021, p. 1-13, 12 jan. 2021. Available at: https://link.springer.com/content/pdf/10.1007/s42979-020-00422-6.pdf. Accessed: 15 jun. 2021.

MINISTÉRIO DA SAÚDE (Brasil). **O que é a Covid-19?** 2021. Available at: https://www.gov.br/saude/pt-br/coronavirus/o-que-e-o-coronavirus. Accessed: 15 jun. 2021.

NGUYEN, Anh Tu. **Covid19-detection**. Available at: https://github.com/covid19-detection/data_collection. Accessed: 15 jun. 2021.

UNIVERSITY OF CAMBRIDGE (England). **COVID-19 Sounds App**: 30,000 audio recordings (and counting!). 30,000 audio recordings (and counting!). 2020. Available at: https://www.covid-19-sounds.org/pt/blog/numbers_update.html. Accessed: 16 jun. 2021.

LARA, Orlandic; TOMAS, Teijeiro; DAVID, Atienza. **The COUGHVID crowdsourcing dataset**: A corpus for the study of large-scale cough analysis algorithms. A corpus for the study of large-scale cough analysis algorithms. 2020. Available at: https://cs.paperswithcode.com/paper/

the-coughvid-crowdsourcing-dataset-a-corpus. Accessed: 16 jun. 2021.

EMBEDDED SYSTEMS LABORATORY (ESL). **A cough-based COVID-19 fast screening project**. 2020. Available at: https://c4science.ch/diffusion/10770/. Accessed: 16 jun. 2021.

M. Cohen-McFarlane, R. Goubran and F. Knoefel, "Novel Coronavirus Cough Database: NoCoCoDa," in IEEE Access, vol. 8, pp. 154087-154094, 2020, doi: 10.1109/ACCESS.2020.3018028.

J. J. Valdés, P. Xi, M. Cohen-McFarlane, B. Wallace, R. Goubran and F. Knoefel, "Analysis of cough sound measurements including COVID-19 positive cases: A machine learning characterization," 2021 IEEE International Symposium on Medical Measurements and Applications (MeMeA), 2021, pp. 1-6, doi: 10.1109/MeMeA52024.2021.9478714.

Solomon K., Vivian C., Daniel T., Amil K. Stanford University. "Virufy on-Device Detection for COVID-19". 2021. Disponível em: https://stanford-cs329s.github.io/reports/virufy-on-device-detection-for-covid-19/. Acesso em: 22 jul. 2021.