

# AUDIO **GENRE** CLASSIFICATION



— **HOLA!**

We are  
Skepsis

# **INTRODUCTION**

Overview of the Project and summary of the Libraries used.



# OBJECTIVE

In this Project we will see how to handle sound files in python, compute sound and audio features from them, and run machine learning algorithms on them and Classify the Audio Files according to their Genres.



## LIBRARIES USED IN THE MODEL

Manipulate multidimensional matrices or arrays as well as mathematical functions operating on these arrays

NUM  
PY

Data manipulation and analysis. In particular, it provides data structures and operations for manipulating numerical arrays and time series

Plotting and visualizing data in graphical form. It can be combined with the NumPy and SciPy python libraries for scientific computation

MATPL  
OTLIB

Aiming to unify and federate a set of Python libraries for scientific use. SciPy uses the arrays and matrices of the NumPy module

Allows you to save one or more variables in a file and retrieve their values later. Variables can be of any type.

PICKLE

python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems.

LIBR  
OSA

Used for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

SKLE  
ARN

Sued to Train and Infer upon Deep Neural Networks.

TENSOR  
FLOW

Used to Implement Deep Learning Model as fast as possible for RnD.

KERAS

lets you play audio directly in your notebook.

IPYDIS  
PLAY

interacting with the operating system. OS, comes under Python's standard utility modules

OS

Used to manipulate different parts of the Python Runtime Environment. It lets us access system-specific parameters and functions.

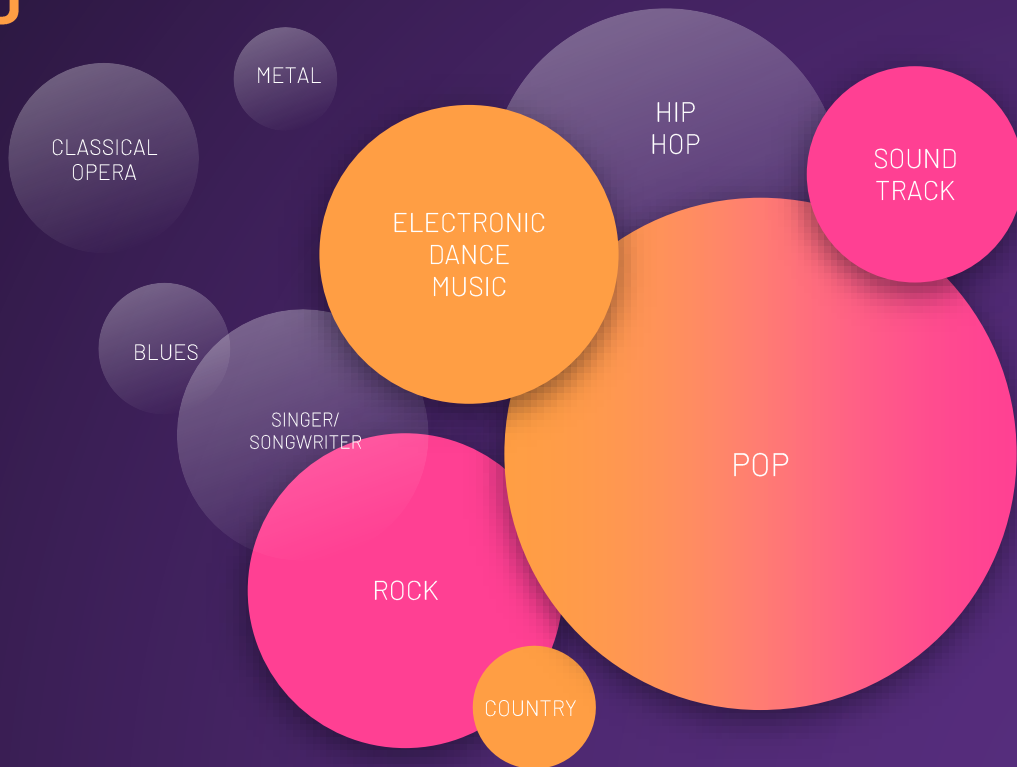
SYS

# MUSIC GENRE CLASSIFICATION

Types, Applications & Difficulties



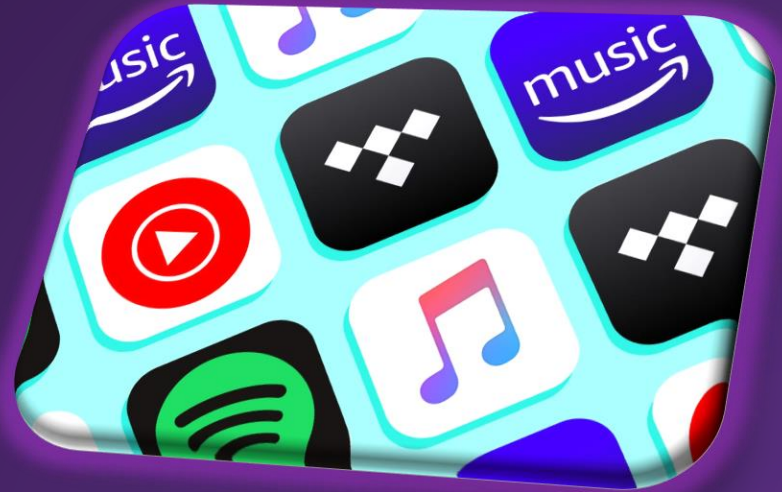
# MUSIC GENRES ALL OVER THE WORLD



## — APPLICATIONS

Automatically:

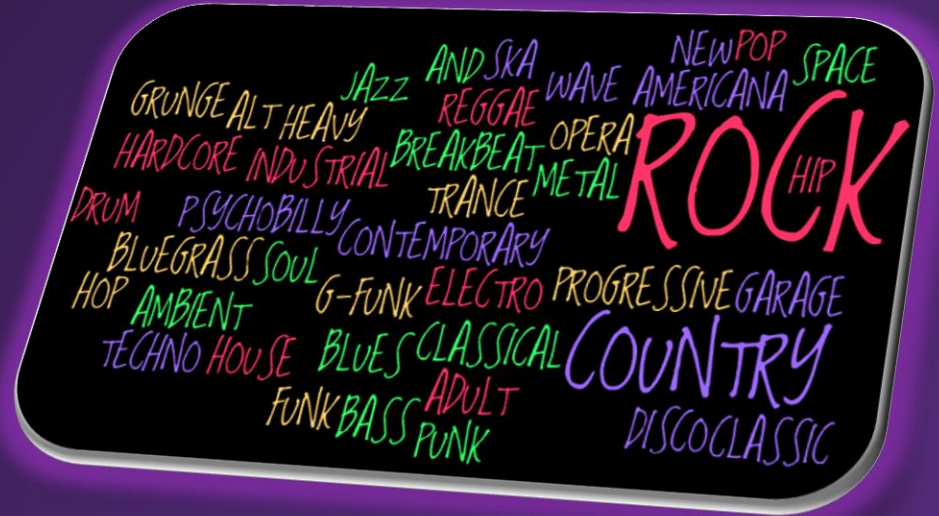
- Suggesting Genre-Specific Artists
- Recommend new music
- Organize music libraries
- Select Music according to choice





## — DIFFICULTIES

- Different songs' length
- Overlapping of genres during songs
- Mixed Genre Audios



# ROADMAP

Loading and Decoding the Audio as a Time Series as a NumPy array with a default sampling rate(SR)

1

Plotting a Spectrogram to visually representing the signal loudness, of a signal over time at various frequencies present in a particular waveform.

3

Plotting Chromagram to analyze music whose pitches can be meaningfully categorized and whose tuning approximates to the equal-tempered scale.

5

Visualizing the Audio File (Decoded into an Arraylike Time Series Data) with Raw Wave plot or the amplitude envelope of audio waveform

2

Analyzing the Zero-Crossing Rate that can occur if successive samples have different algebraic signs. The rate at which zero crossings occur is a simple measure of the frequency content of a signal.

4

Preprocessing the data with SKLearn by encoding the label column with the function `LabelEncoder()` and splitting into test and train to obtain the Test Loss & Accuracy Score

6

# THE GTZAN — MUSIC GENRE DATASET



## — ABOUT THE DATASET

- **genres original** – A collection of 10 genres with 100 audio files each, all having a length of 30 seconds (the famous GTZAN dataset, the MNIST of sounds)
- **images original** – A visual representation for each audio file. One way to classify data is through neural networks. Because NNs (like CNN, what we will be using today) usually take in some sort of image representation, the audio files were converted to Mel Spectrograms to make this possible.



## — CONTENTS OF THE DATASET

- **CSV files** – Containing features of the audio files. One file has for each song (30 seconds long) a mean and variance computed over multiple features that can be extracted from an audio file. The other file has the same structure, but the songs were split before into 3 seconds audio files (this way increasing 10 times the amount of data we feed into our classification models). With data, more is always better.

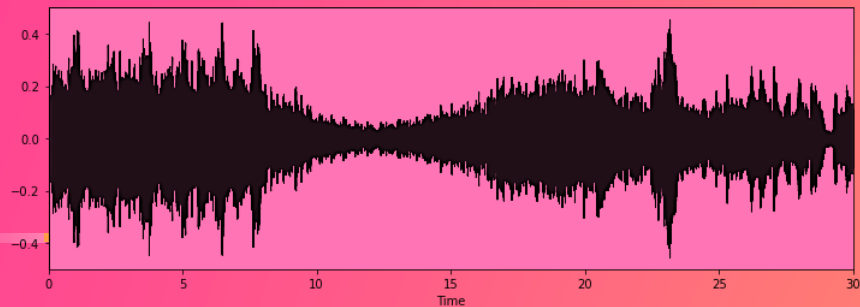




# **RAW WAVE PLOT**

Plot Raw Wave plot is the amplitude envelope of a waveform



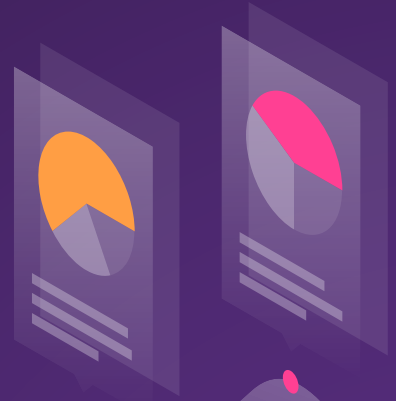


```
plt.figure(figsize=(12, 4))  
librosa.display.waveplot(data, color = "#502A75")  
plt.show()
```

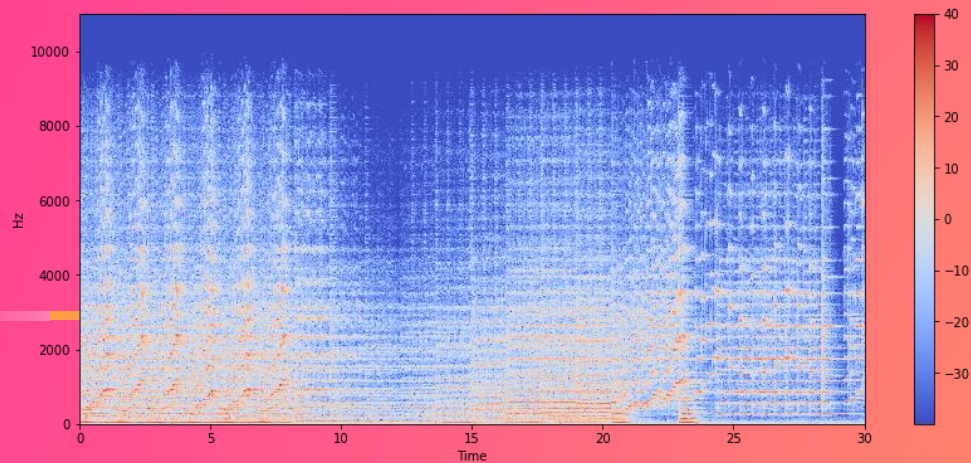
**Visualizing the Audio**  
Plotting raw wave file.

# **PLOT SPECTROGRAM**

The Visual Way of representing the Signal Loudness of a Signal over time at Various Frequencies present in a particular waveform







```
X = librosa.stft(data)
Xdb = librosa.amplitude_to_db(abs(X))
plt.figure(figsize=(14, 6))
librosa.display.specshow(Xdb, sr=sr,
x_axis='time', y_axis='hz')
plt.colorbar()
```

## Plot Spectrogram

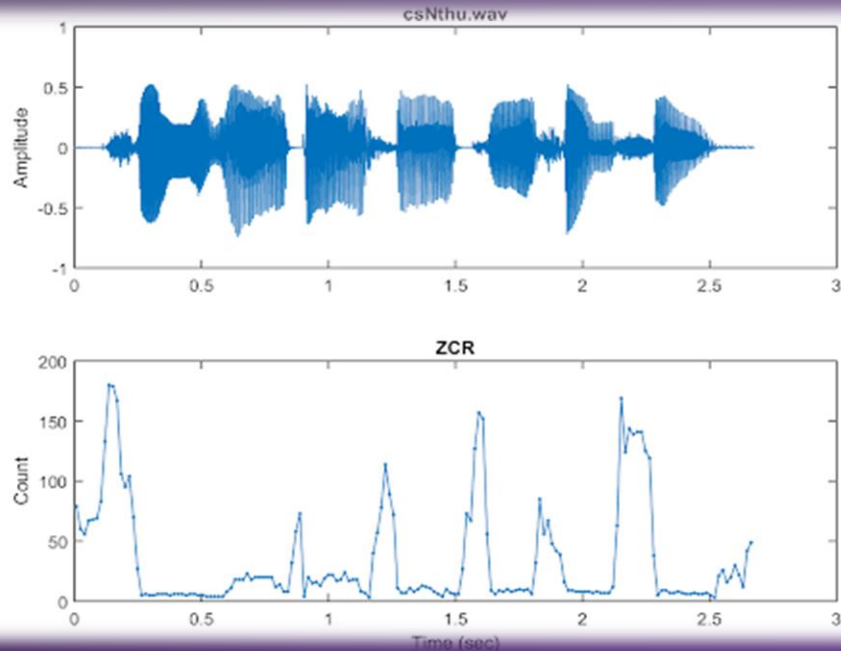
(Frequency vs Time)

# **ZERO CROSSING RATE**

Said to occur if successive  
samples have different  
Algebraic Signs



A Visualization of Zero Crossing Rate which is the rate at which a signal changes from positive to zero to negative or from negative to zero to positive.

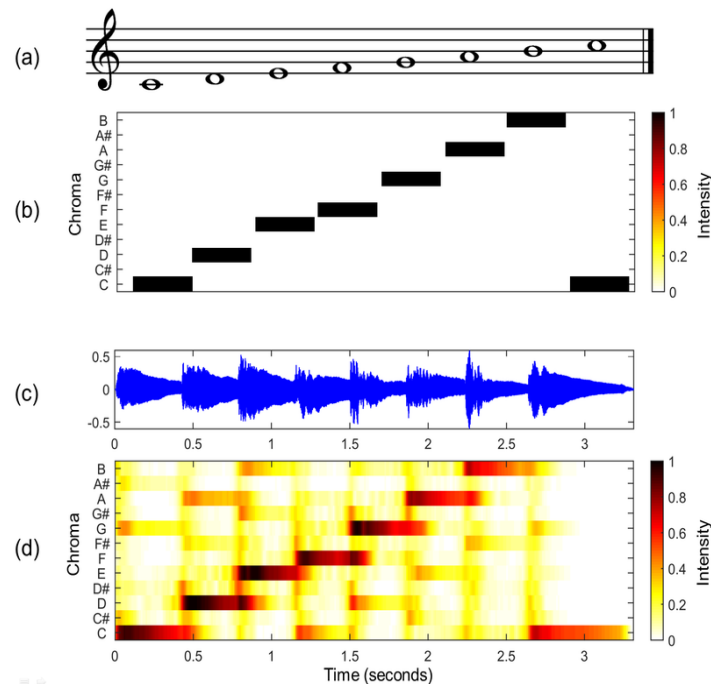


# **CROMA FEATURE**

A powerful tool for analyzing music whose pitches can be meaningfully categorized and whose tuning approximates to the equal-tempered scale.



Croma Feature  
capture harmonic  
and melodic  
characteristics of  
music, while being  
robust to changes  
in timbre and  
instrumentation.



## — PROCESS FLOW

### FOUR LAYERED NEURAL NETWORK

We worked on the Dataset to Prepare a Four Layered Neural Network with the Four Consecutive Layers having 256, 128, 64, 10 Nodes respectively.

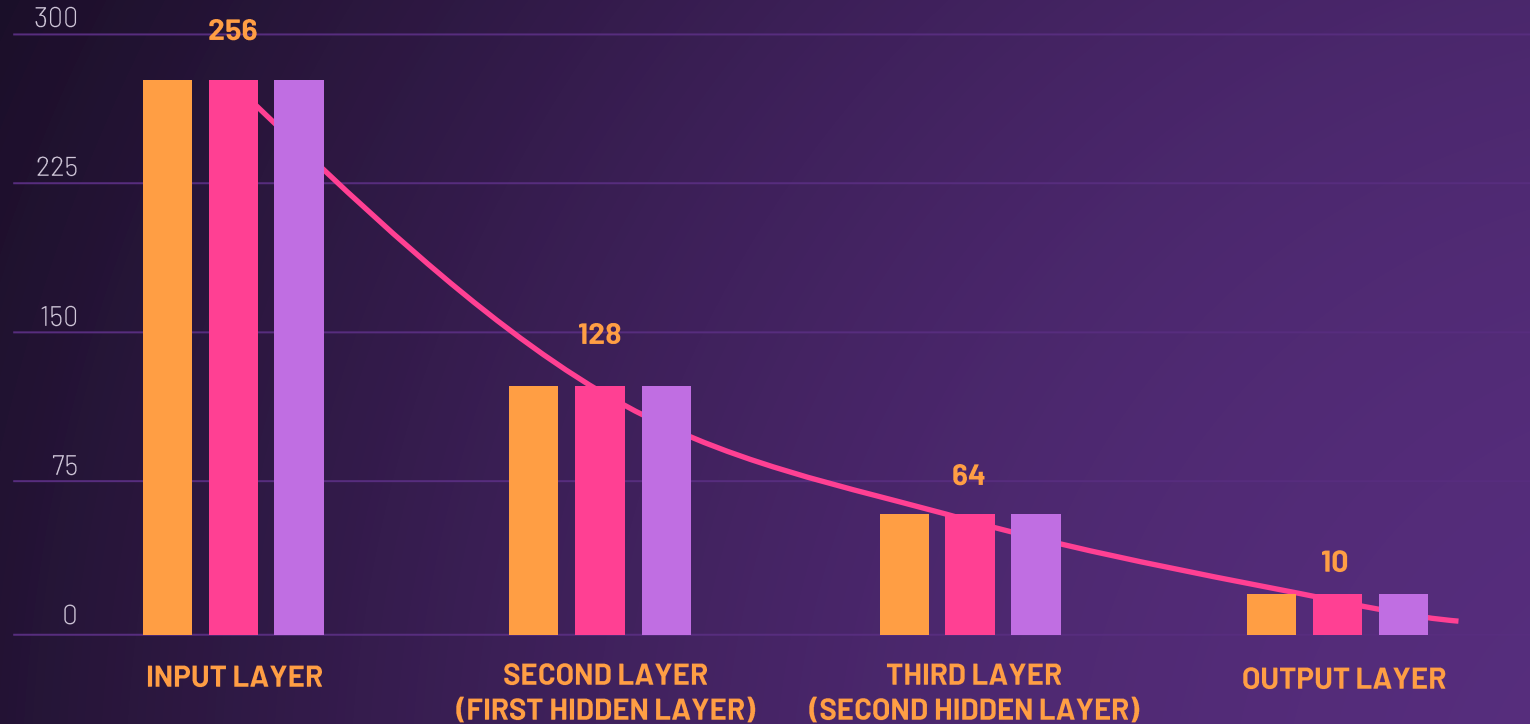
### USING OF ACTIVATION FUNCTION

We used Rectified Linear Unit as the Activation Function in the first three layers and SOFTMAX in the Output a.k.a. the Final Layer and Adam as the Optimizer of the Neural Network

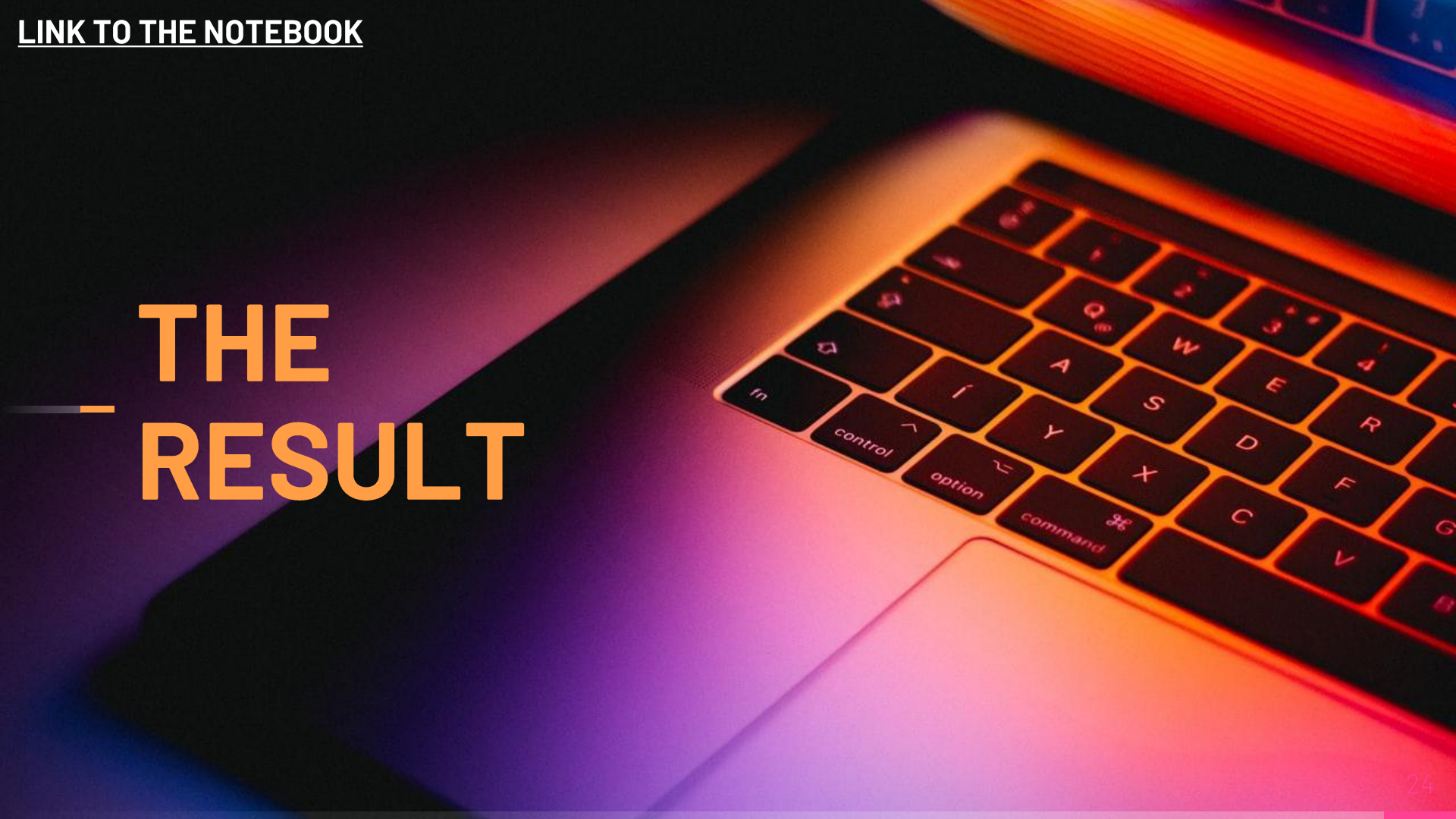
### FEEDING THE DATA IN THE NEURAL NETWORK

We did the Train Test Split in a 67:33 Ratio and reshaped the Spectrogram's Data into a One Dimensional Data and fed in a densely packed Neural Network.

# NUMBER OF NODES PER LAYER







[LINK TO THE NOTEBOOK](#)

# THE RESULT



**150**

Epoch Value

**128**

Batch Size

**HYPERPARAMETERS**

**0.9038 (90.38%)**

Test Accuracy

**0.9931**

Test Loss

**RESULT**

## MEMBERS



**Sagnik Mitra**

PROAIE



**Sneharup Mukherjee**

PROBLE



**Spandan Pal**

PROMLE

## BIBLIOGRAPHY

### ❖ Music Genre Classification with Deep Learning

Author: Bryan Landsdown, University of Birmingham

### ❖ Music Genre Classification Systems – A Computational Approach

Author: Peter Ahrendt

### ❖ MML Classification of Music Genres

Author: Adrian C. Bickerstaffe, Enes Makalic

### ❖ Mugec: Automatic music genre classification

Author: Rohit Singh, Coventry University



**THANK YOU**