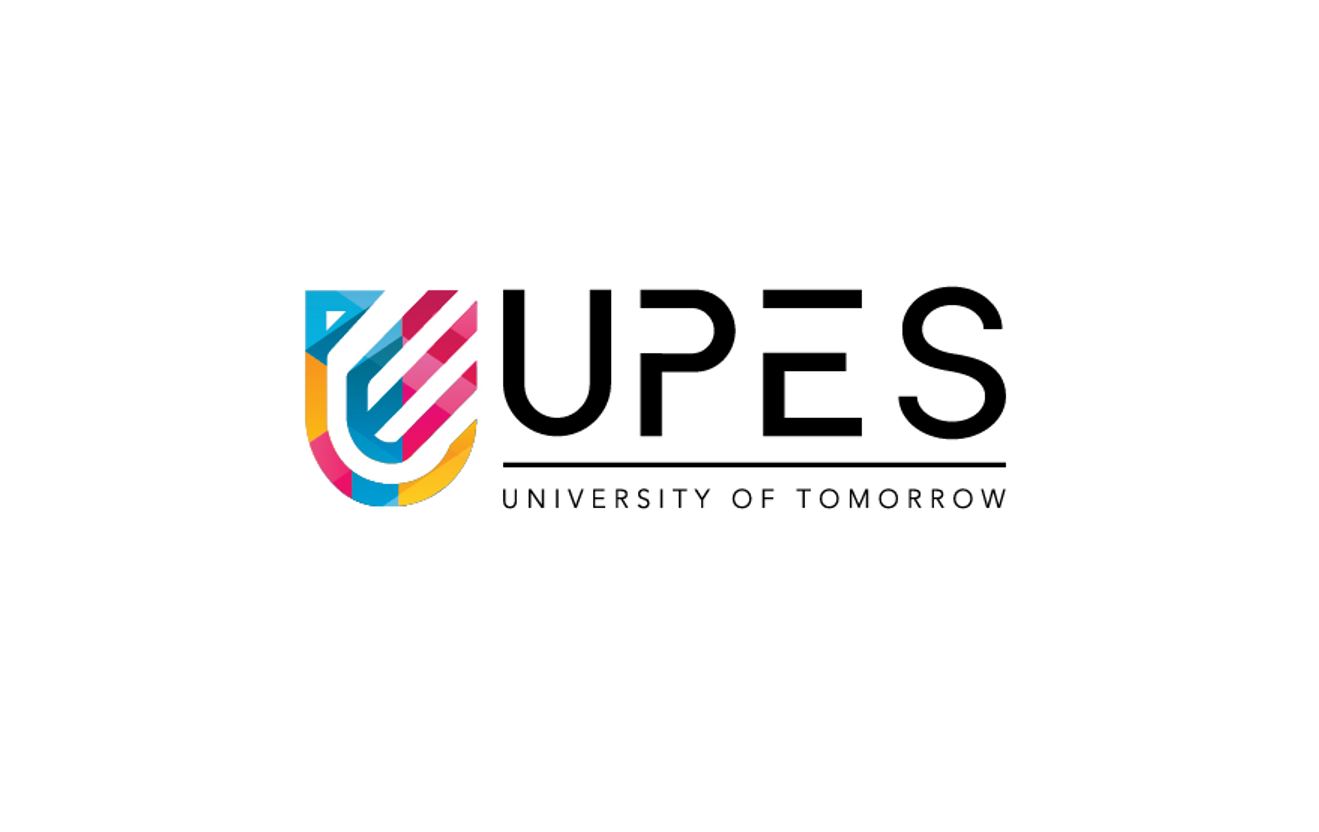
**Audio Frequency Visual Classifier**

|  |  |  |  |
| --- | --- | --- | --- |
| Member Name | Member SAP ID | Member Roll No | Batch No. |
| Rishabh Rawat | 500105035 | R2142220613 | B3 Non Honours |
| Divyansh Bansal | 500110160 | R2142221409 | B3 Non Honours |
| Vaibhav Singh | 500110074 | R2142221363 | B3 Non Honours |
|  |  |  |  |
|  |  |  |  |

00



Course Name: Demystifying AI

Course Code:

Semester: 4

List of Contents

Introduction

Problem Statement

Methodology

Result and Discussion

Conclusion

Bibliography

Introduction

The project aims to see whether there is a correlation between songs of same genres, and if the frequency features of different genres in a visualized form differ enough from each other to predict which genre a song belongs to. To approach the problem, we make use of 3 different Classification models, Support vector machines, MLP Classifier, and Logistic regression to predict the genre of a song. The use of 3 classification models will further devise which model is most apt for classification.

##### :Literature Review

Different genres of music are often associated with different feelings. An accurate model will correctly identify the features of a song and predict it’s genre. While some popular genres, like “romance” and “party” are different enough to identify, other genres like “chill” and “feel good” may often have such overlapping features that even a human might fail to correctly classify a song, depending on one’s taste in music.

There have been multiple attempts at creating models that can classify a song correctly, with attempts such as loudness, vocals, instruments, etc. However, genre has proved to be a difficult set to classify, since it varies so much with regionality, languages, instruments and even time periods.

Findings on exploring music genres, and looking for similar features in hit songs of same genre by Rebecca Stelter[1] indicates that the is a large number of features that groups together many features of the same genre.

##### Findings

While most genres have specific features enough to classify with high accuracy, some other extraneous genres like “workout” may not have enough features to be classified. Further, the Neural Network model has the highest accuracy at 78%, then the SVM at 75% and Logistic Regression stands last at 72%. However, the Logistic regression takes least time, and SVM the highest.

Problem Statement

Make use of any 3 classification models, and comapre the accuracy and time complexity on data set. In our case this dataset is a collection of plots of frequencies of 354 songs, divided into 5 genres, “workout”, “chill”, “sleep”, “Party”, and “sad”.

Methodology

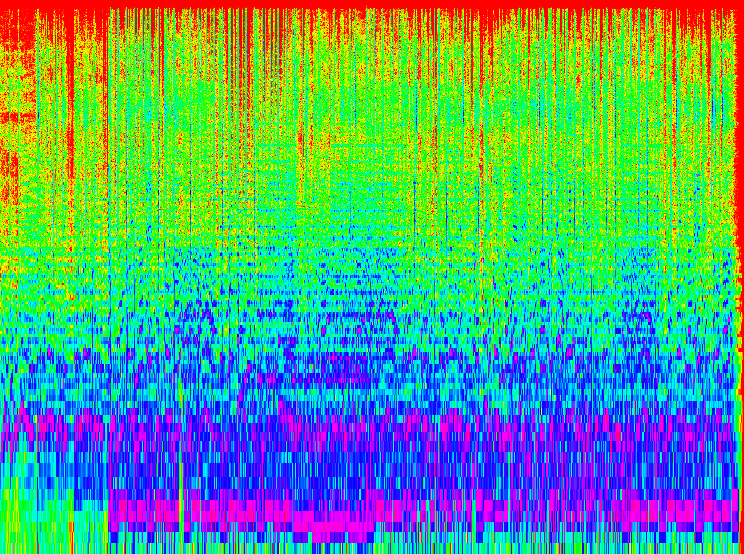
* **Gathering the data-set:** As mentioned before, the data-set has 354 songs, divided across 5 genres. These genres vary widely from each other. Since most copyrighted music is intellectual property protected by copyright law, we will not discuss the availability of any such sources throughout this paper.
* **Decoding:** MP3 is likely most popular audio sharing format, which is widely used and available all over the world. However, MP3 format uses a special encoding of its own, where it converts raw data to encoded data using fast fourier transformation as well as a psyhoacoustic model.

Since we cannot directly use the encoded data, we decode it to a wav file with pcm encoding. Unlike MP3, “.wav” file format is just a container, that can store any data, most popularly with PCM encoding. We achieve this using a handy library called “pymp3”. The decoder file is in the “preprocessing” directory as “decoder.py”. It has two functions, both named “mp3\_to\_wav”, where one takes a file name as argument, while the other takes a io.BytesIO object with wav data as argument.

* **Visualization:**  Once we have the files decoded to “.wav” container with PCM format, we can then visualize it. The visualization file is stored in  the “preprocessing” directory, in the “visualiser.py” file. It contains a singular function “audio\_to\_image” that takes a file name or io.BytesIO object with wav data.

For plotting we use librosa and matplotlib. Librosa is a Python library for analyzing, visualizing, and working with audio data, particularly designed for music and signal processing tasks.

1. First we load each file into librosa, and set mono channel to True so we can have a one dimensional array of values.
2. Then we perform the stft transformation on data, to get the value of all frequencies at time ‘t” of the signal.
3. Then we use abs function form the numpy library to convert the complex values to receive their magnitude.
4. Next, we use the amplitude\_to\_db function of librosa to scale data to loudess, since each song likely has different loudness associated with it. We possibly lose some features due to this, but since the music is produced by such a large variety of people, doing this ensures there is no disparity due to it.
5. The we use librosa to convert this to a specshow graph, and matplotlib to either save it or return.
6. Output Example of a given wav file:



* **Traning Models**:

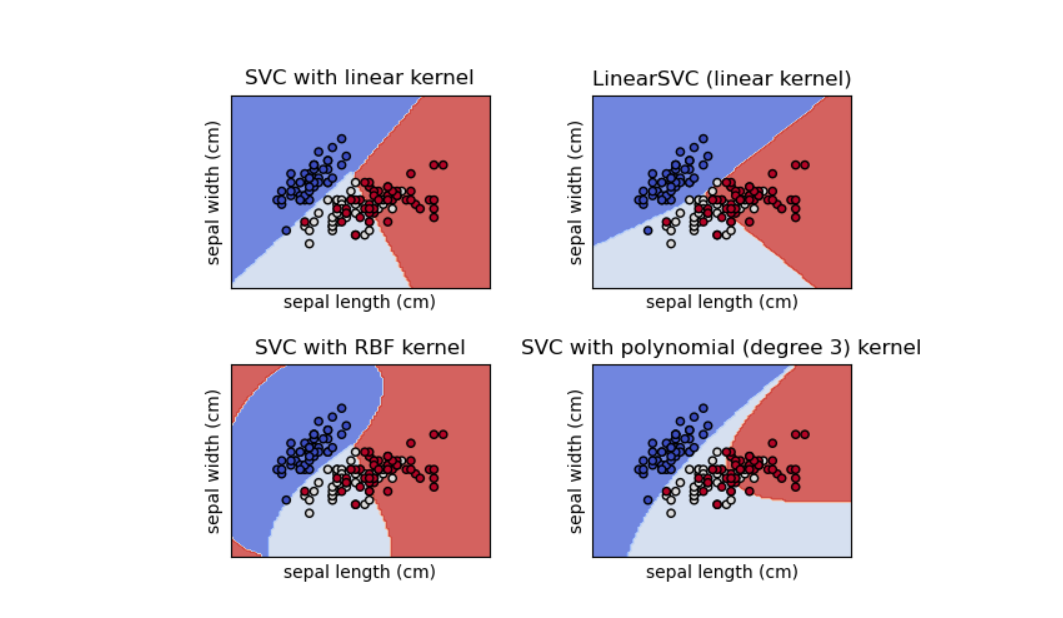
Loading data is similar across the three classification model. We iterate through each of the genre directory and load each image into memory. We flatten each image since we require data in a 1 D vector, but directly reading image will give us a 3d since we have color images. We also load the respective label for each image.

* 1. **Support Vector Machines:**

A supervised machine learning algorithm used for classification and regression analysis, aiming to find the optimal hyperplane that separates data points into different classes.

The SVM model and related files are stored in the “SVM directory”. For training the model, we use the SVC model, and not the Linear SVC model which is specifically made for linear classification task, or the NuSVC which provides more control over training error and number of support vectors which would increase complexity.

We also use the “rbf” kernel, which gave the best performance amongst the kernels. Below is a sample visualization of each kernel:



Next we simply split the dataset, train the model and save it as “SVM.pkl”.

For the dataset of 5 genres we test on, we now create a confusion matrix (discussed later).

2. **Logistic Regression Model**

Logistic regression is a binary classification model and is built for two-class classification. We use the “lbfgs” solver, for no particular reason.

Since the logistic regression was made for binary classification, our implementation through scikit uses a workaround through “multinomial”. In this case, the model predicts the probability of each possible class for a given input. Instead of just outputting a single probability for a binary outcome, it produces a vector of probabilities.

In the training phase, the model aims to minimize a specific loss function known as multinomial loss, also commonly known as cross-entropy loss. This loss function measures the disparity between the predicted probabilities and the actual class labels. It penalizes the model based on how far the predicted probabilities are from the true probabilities associated with each class.

The model creation process similar to the SVM model creation, we split the dataset, train model, check accuracy, create confusion matrix and save the dataset.

3. **MLP Classifier**

MLP Classifier is a form of Neural network and a basic version of a CNN model. It uses back propagation to train the model.

For the Solver, we use stochastic gradient descent. It uses less memory than “L- BFGS” which is a must since we use a scaler for MLP. Using the “Adam” solver was discarded since it has low accuracy on small datasets.

We use an alpha of value 0.001, but changing it does not seem to have an effect on the accuracy score.

We also need to use a Scaler, in this case a standard scaler since MLP classifier is very sensitive to feature scaling, which can lead to large changes in accuracy.

Rest is much the same, we split the dataset, create model, predict, create confusion matrix and save the model in a “pkl” file.

* **Prediction**

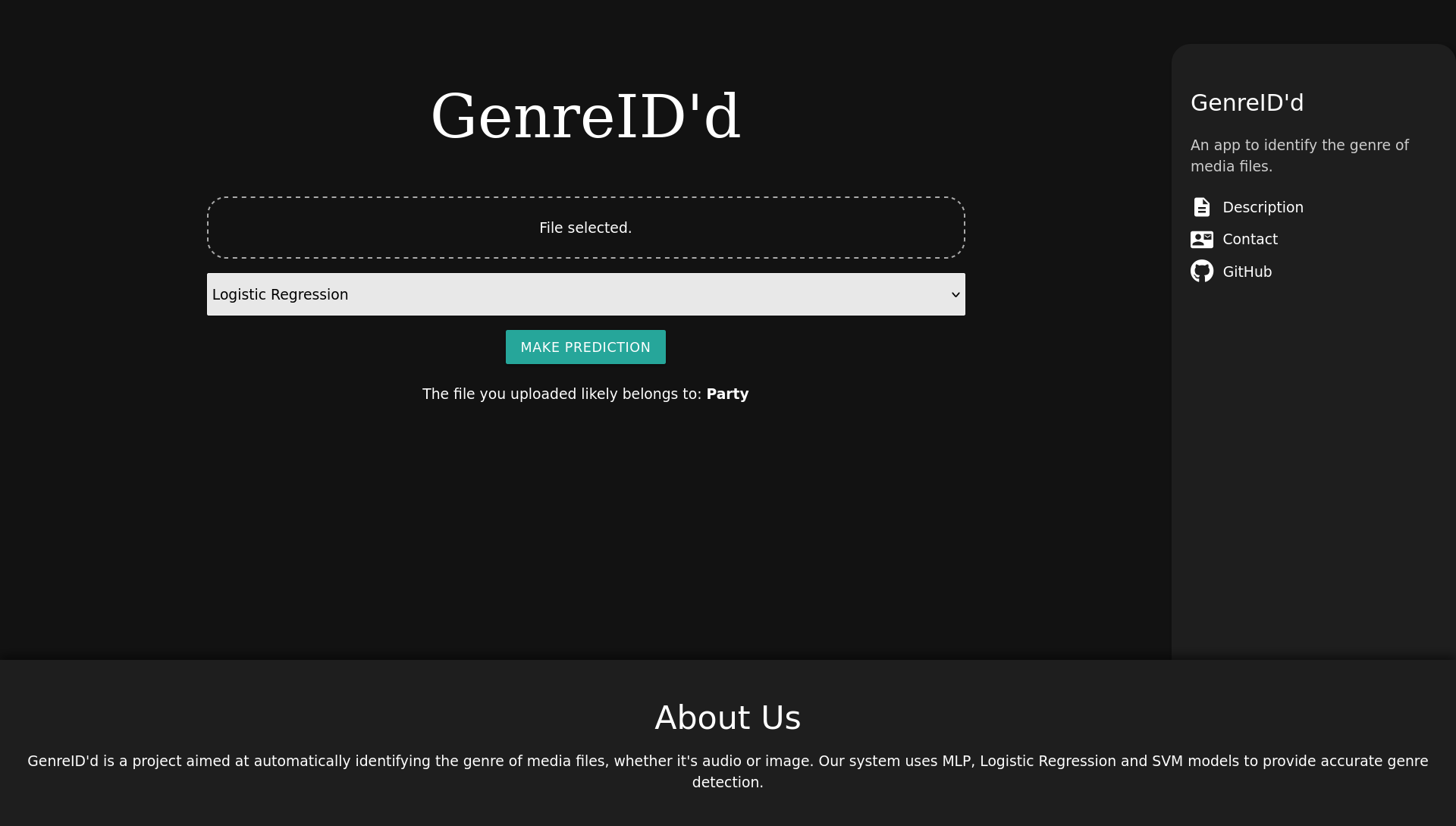
Now we can simply take a audio file, convert it to wav file with PCM encoding. Then we use visualiser.py to convert this audio file to image, which we can pass onto the models for processing.

* **Frontend (extra)**

The frontend is a simple webpage hosted via flask, that has two endpoints, one for hosting the webpage, and another for receiving the mp3, wav or png file uploaded by user. The user can upload one file at a time, and select the model to use, and will get the output in the same page. For aesthetics, Materialize, material icons, and font awesome icons have been used.

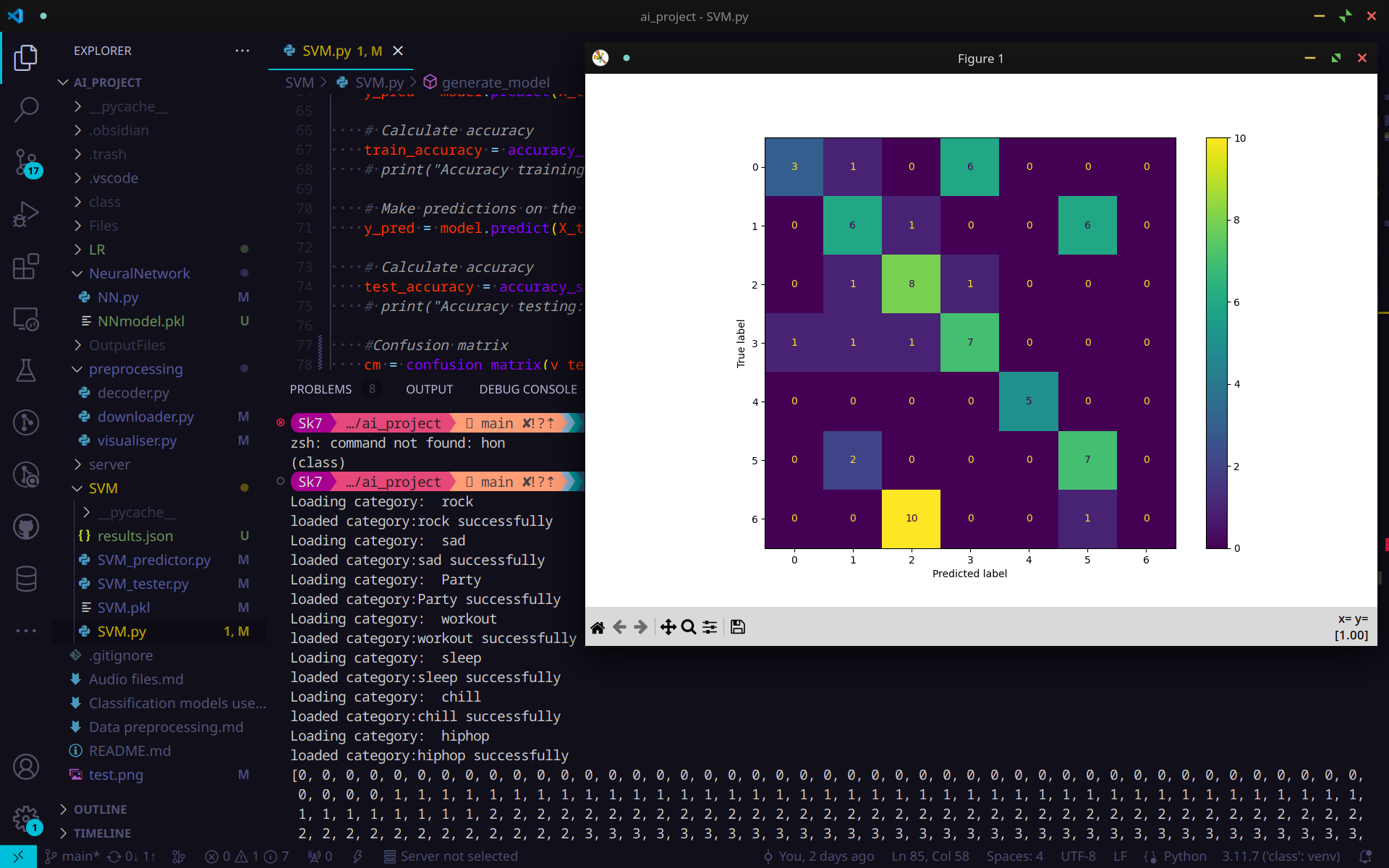
The webpage is hosted at: 127.0.0.1:5000 .

Sample preview:

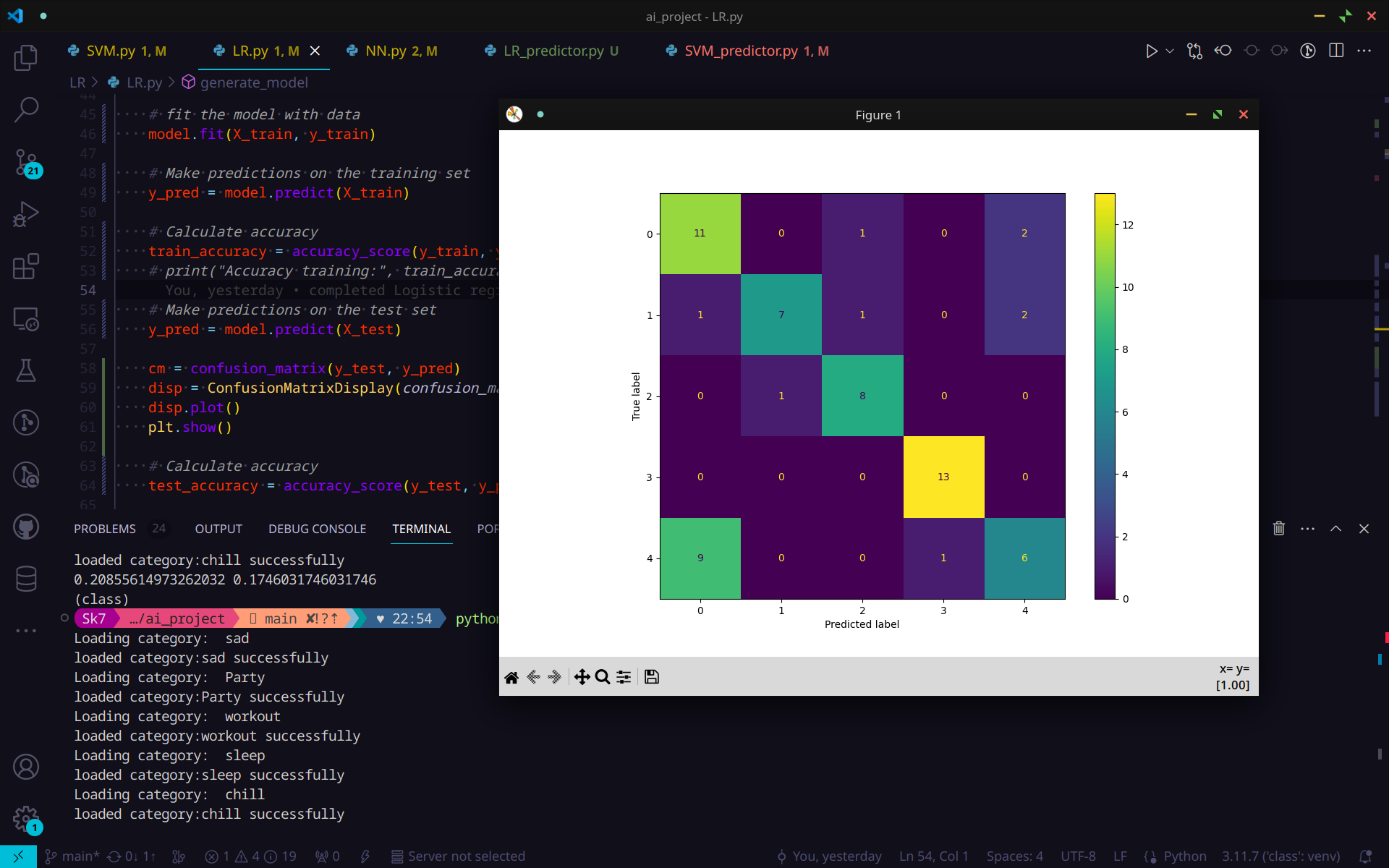


Result and Discussion

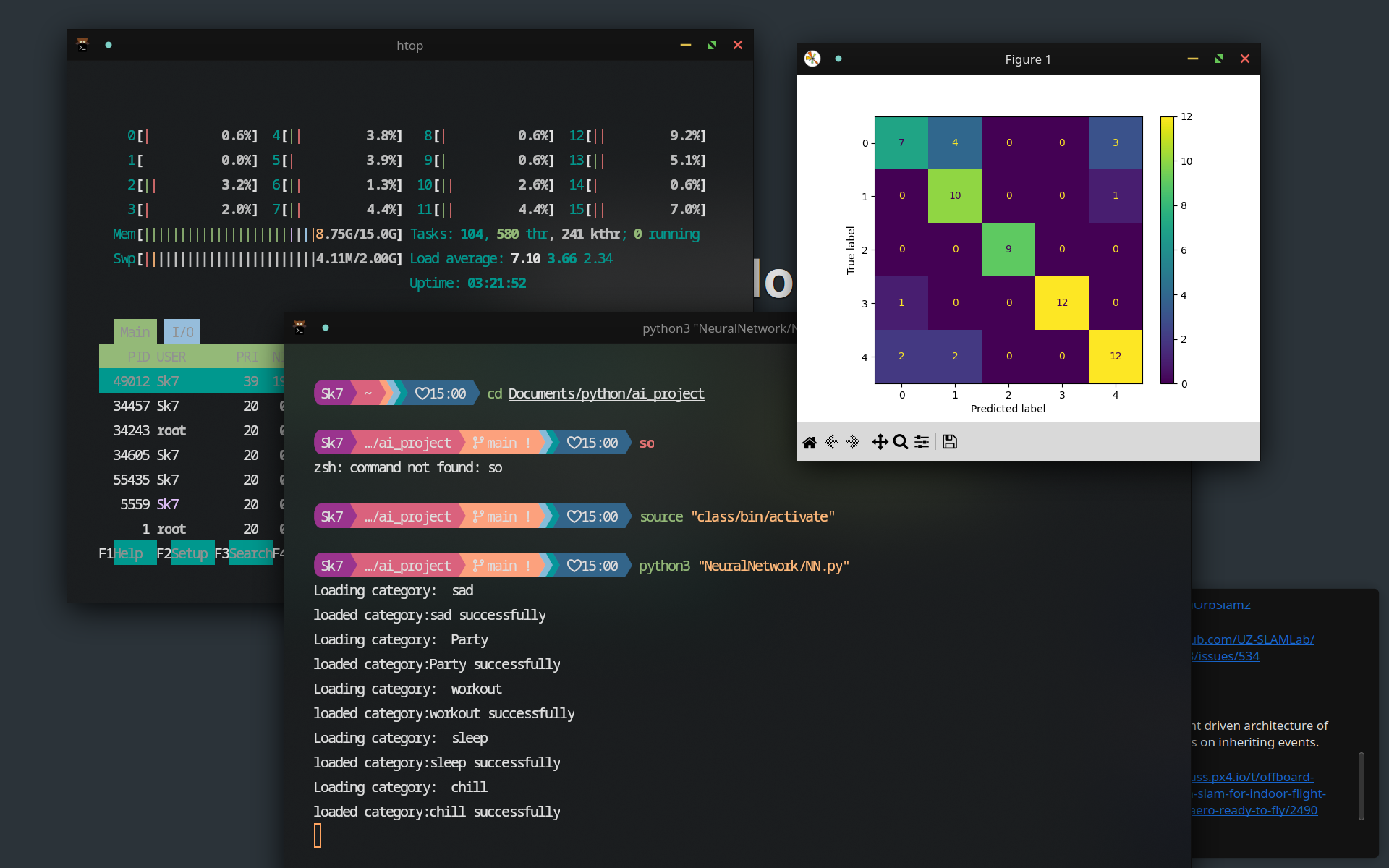
* From our tests, we can say that the 5 genres we selected have a good accuracy of 75% in predicting the genre of a song. Further this indicates that the frequencies of songs have an association with the genres they belong too.
* Of the three classification models, the MLP is the most accurate, with highest sensitivity of 78%, then the SVM model with 75% and Logistic Regression with 72%.
* However the MLP model is severely sensitive to feature scaling, and un-scaled data leads to accuracy drop from 78% to 17%;
* Due to scaling, MLP model even with “sgd” solver had highest memory usage during traning and created the largest model at 486MBs after compression. SVM had the second largest model at 148MBs. However Logistic regression was the best with a model size of 27MBs.
* Further the Logisctic regression Model was the fastest, and SVM the slowest, with the former being 2 to 3 times faster than the latter.
* Further, just for sake of it, Since the models can take any data, we can use any image, for example a car or dog to find out what genre that data would belong to.
* Below are the confusion matrix of SVM, Logistic regression and MLP. The middle diagonal should be lighter for better accuracy and the rest of the graph should be darker: SVM model (2 extra genres for comparison)



LR model:



MLP Classifier:



Conclusion:

Hence we can conclude that songs of same genres share common features between each others.

For higher accuracies, MLP and SVM are better, but the tradeoff of space and time that Logistic Regression provides is too good to be ignored.

The Scikit and Librosa libraries provide advanced tools to manage signal data, and creating macine learning algorithms.

Bibligraphy:

[1]: Smith, J. (2020). The Impact of Climate Change on Biodiversity [PDF document]. Retrieved from <https://scholarworks.bgsu.edu/cgi/viewcontent.cgi?article=1887&context=honorsprojects>

[2]: scikit-learn. (n.d.). Neural network models (supervised). In scikit-learn: Machine Learning in Python (Version 0.24.0). Retrieved from <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>