MUSIC GENRE CLASSIFICATION

Under the guidance of:
Ms. Gayathri R
Mr. Ragesh Verma

Team Members:
Maddali Sowmya
Pooja Chandrashekara

AGENDA

1. Problem Statement
2. Objectives
3. Proposed Architecture
4. Dataset
5. Technologies used
6. Library used for the project
7. Outputs
8. Applications
9. Future Milestones

♪ PROBLEM STATEMENT

Music plays a very important role in people’s lives. Music brings like-minded people together and is the glue that holds communities together. Communities can be recognized by the type of songs that they compose, or even listen to. Different communities and groups listen to different kinds of music. One main feature that separates one kind of music from another is the genre of the music.

♫ **OBJECTIVES**

1. Developing a machine learning model that classifies music into specific genres depending on the features they exhibit.
2. To reach a good accuracy so that the model classifies music into its genre correctly.

𝄞 PROPOSED ARCHITECTURE

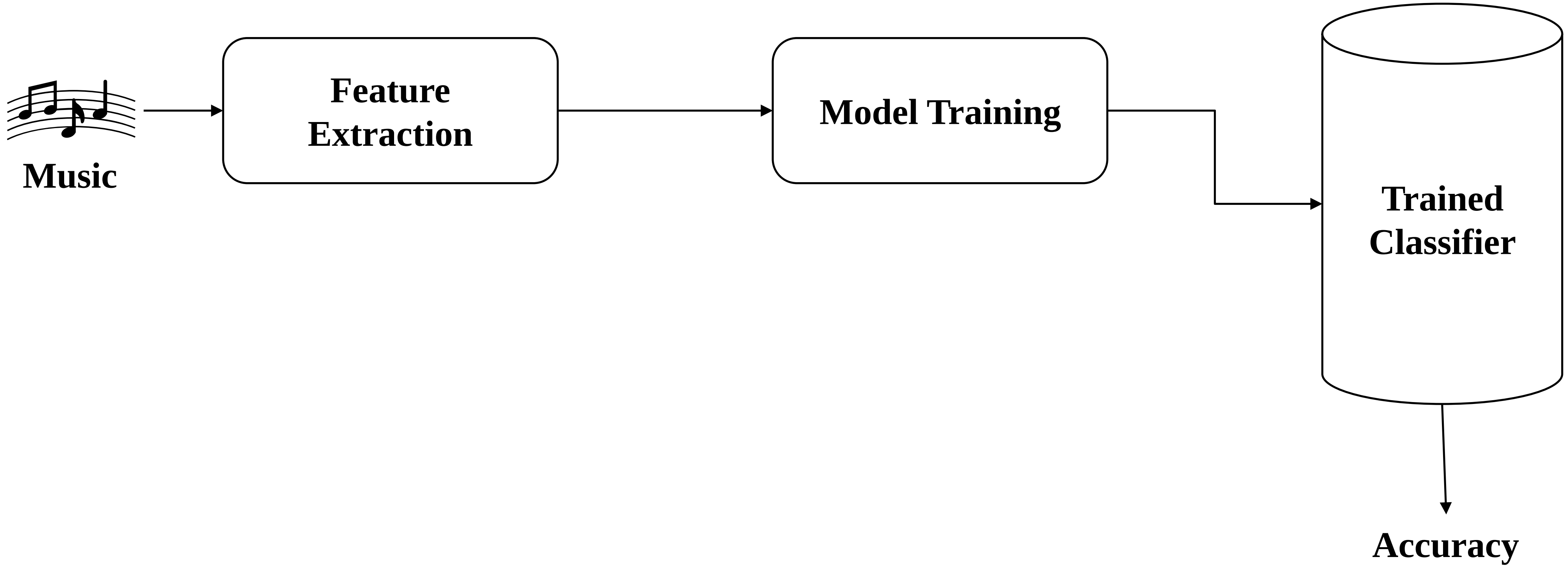

Fig-1: Proposed Architecture for Music Genre Classification

DATASET

- GTZAN dataset consists of 1000 audio files each having 30 seconds duration.
- There are 10 classes (10 music genres) each containing 100 audio tracks.
- Each track is in “**.wav**” format.

DATASET

- It contains audio files of the following 10 genres:

1. Blues
2. Classical
3. Country
4. Disco
5. Hip-hop
6. Jazz
7. Metal
8. Pop
9. Reggae
10. Rock

♪♫ TECHNOLOGIES USED

Machine Learning:

- Machine Learning is a field of study that looks at using computational algorithms to turn empirical data into usable models.
- The machine learning field grew out of traditional statistics and artificial intelligence communities.
- Machine learning algorithms can be used to:
  - Gather understanding of the cyber phenomenon that produced the data under study.
  - Abstract the understanding of underlying phenomena in the form of a model.
  - Predict future values of a phenomena using the above-generated model
  - Detect anomalous behavior exhibited by a phenomenon under observation.

LIBRARY USED FOR THE PROJECT

Librosa:

- It is a Python module to analyze audio signals in general but geared more towards music.
- It includes the nuts and bolts to build a MIR(Music information retrieval) system.
- It helps to visualize the audio signals and also do the feature extractions in it using different signal processing techniques.

OUTPUTS

Load the dataset

In [2]:
```
df = pd.read_csv('Data/features_30_sec.csv')
df.head()
```

Out[2]:

	filename	length	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var	spectral_centroid_mean	spectral_centroid_var	spectral_bandwidth_mean
0	blues.00000.wav	661794	0.350088	0.088757	0.130228	0.002827	1784.165850	129774.064525	2002.449060
1	blues.00001.wav	661794	0.340914	0.094980	0.095948	0.002373	1530.176679	375850.073649	2039.036516
2	blues.00002.wav	661794	0.363637	0.085275	0.175570	0.002746	1552.811865	156467.643368	1747.702312
3	blues.00003.wav	661794	0.404785	0.093999	0.141093	0.006346	1070.106615	184355.942417	1596.412872
4	blues.00004.wav	661794	0.308526	0.087841	0.091529	0.002303	1835.004266	343399.939274	1748.172116

5 rows × 60 columns

Fig-2: Sample of the dataset used for the model

OUTPUTS

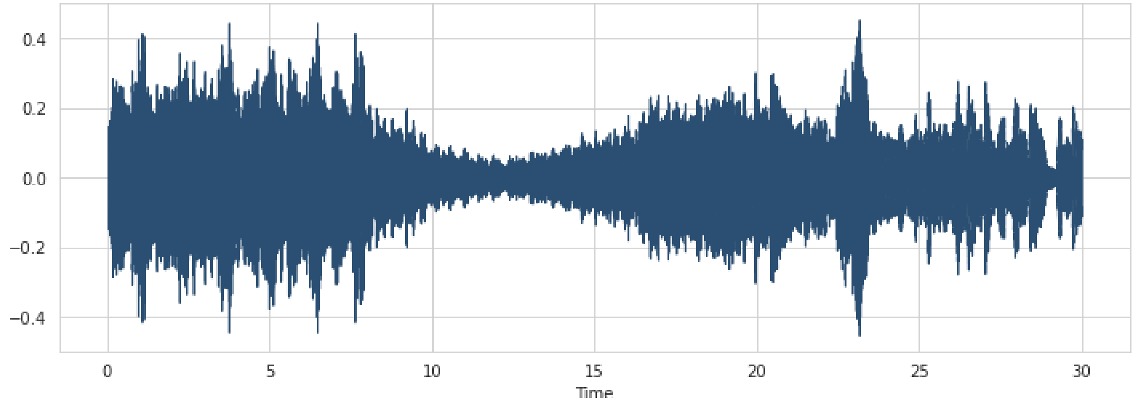

Fig-3: Plot Raw Wave File

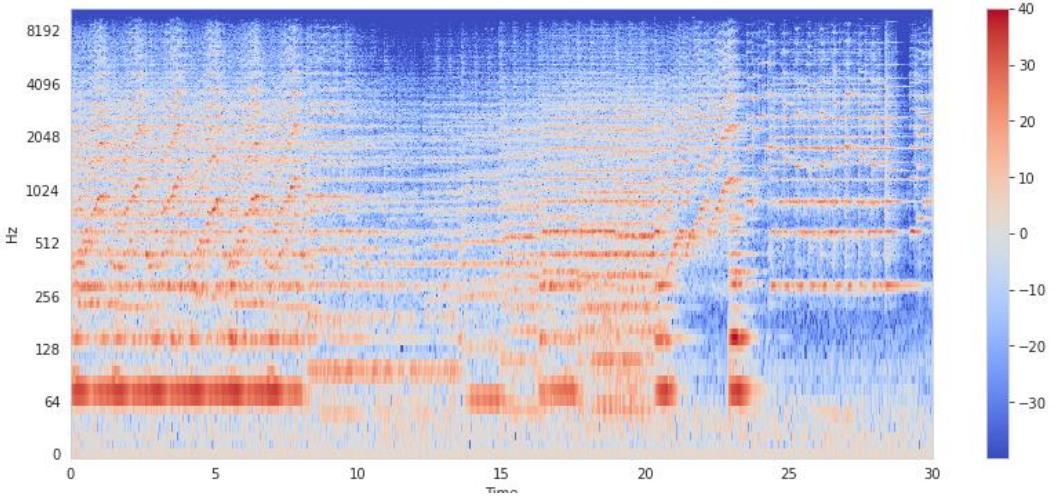

Fig-4: Plot Spectrogram

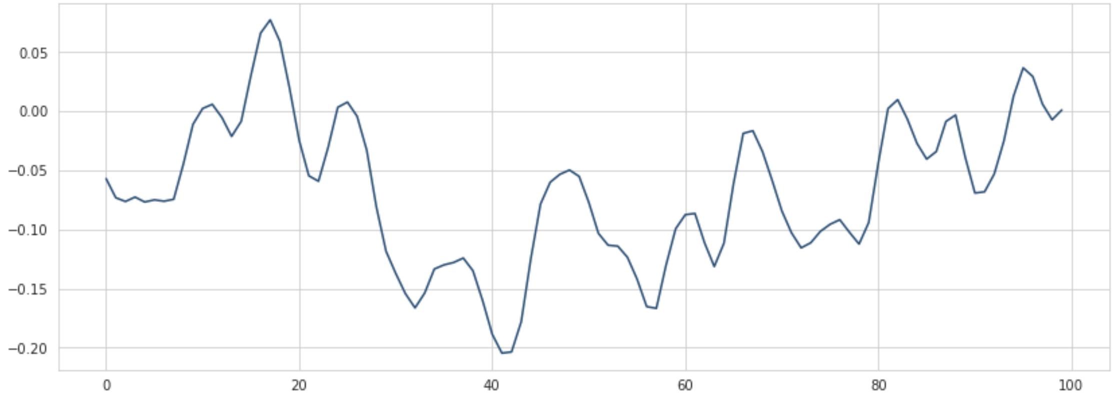

Fig-5: Zero Crossing Rate Waveform

OUTPUTS

Initial Model Fitting & Recursive Feature Elimination

```
In [16]: estimator = XGBClassifier(eval_metric='merror')
         rfecv = RFECV(estimator, step=1, cv=5,scoring='accuracy',verbose=1)
         rfecv.fit(X_train, y_train)
```

```
Fitting estimator with 65 features.
Fitting estimator with 64 features.
Fitting estimator with 63 features.
Fitting estimator with 62 features.
Fitting estimator with 61 features.
Fitting estimator with 60 features.
Fitting estimator with 59 features.
Fitting estimator with 58 features.
Fitting estimator with 57 features.
Fitting estimator with 56 features.
Fitting estimator with 55 features.
Fitting estimator with 54 features.
Fitting estimator with 53 features.
Fitting estimator with 52 features.
Fitting estimator with 51 features.
Fitting estimator with 50 features.
Fitting estimator with 49 features.
Fitting estimator with 48 features.
Fitting estimator with 47 features.
```

Fig-6: XGBoost Initial Classifier

Model Training

```
In [19]: model = XGBClassifier(n_estimators=1000)
         model.fit(X_train,y_train,eval_metric='merror')

Out[19]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
                       colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
                       gamma=0, gpu_id=-1, importance_type=None,
                       interaction_constraints='', learning_rate=0.300000012,
                       max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
                       monotone_constraints='()', n_estimators=1000, n_jobs=1,
                       num_parallel_tree=1, objective='multi:softprob', predictor='auto',
                       random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weight=None,
                       subsample=1, tree_method='exact', validate_parameters=1,
                       verbosity=None)
```

Fig-7: XGBoost Model Training

OUTPUTS

```
In [24]: model1 = XGBClassifier(n_estimators=1629, reg_lambda=7)
         model1.fit(X_train,y_train,eval_metric='merror')
         y_pred_test1 = model1.predict(X_test)
         print(f"accuracy: {accuracy_score(y_test,y_pred_test1)}")
         print(f'New tuned model:\n {classification_report(y_test, y_pred_test1, labels=target_names)}')
```

```
accuracy: 0.81
New tuned model:
               precision    recall  f1-score   support

       blues       0.75      0.90      0.82        10
   classical       1.00      0.80      0.89        10
     country       0.83      0.50      0.62        10
       disco       0.82      0.90      0.86        10
      hiphop       1.00      0.90      0.95        10
        jazz       0.77      1.00      0.87        10
       metal       0.83      1.00      0.91        10
         pop       0.80      0.80      0.80        10
      reggae       0.89      0.80      0.84        10
        rock       0.50      0.50      0.50        10

    accuracy                           0.81       100
   macro avg       0.82      0.81      0.81       100
weighted avg       0.82      0.81      0.81       100
```

Fig-8: XGBoost Classifier after Hyperparameter Tuning and the Accuracy obtained

OUTPUTS

```
In [15]: classifier = model.fit(X_train,
                                y_train,
                                epochs=100,
                                batch_size=128)
```

```
Epoch 1/100
53/53 [==============================] - 1s 5ms/step - loss: 1.4796 - accuracy: 0.4835
Epoch 2/100
53/53 [==============================] - 0s 9ms/step - loss: 0.9004 - accuracy: 0.6980
Epoch 3/100
53/53 [==============================] - 0s 8ms/step - loss: 0.6973 - accuracy: 0.7753
Epoch 4/100
53/53 [==============================] - 0s 9ms/step - loss: 0.5860 - accuracy: 0.8083
Epoch 5/100
53/53 [==============================] - 0s 9ms/step - loss: 0.4847 - accuracy: 0.8440
Epoch 6/100
53/53 [==============================] - 0s 7ms/step - loss: 0.4159 - accuracy: 0.8642
Epoch 7/100
53/53 [==============================] - 0s 6ms/step - loss: 0.3505 - accuracy: 0.8887
Epoch 8/100
53/53 [==============================] - 0s 6ms/step - loss: 0.3058 - accuracy: 0.9050
Epoch 9/100
53/53 [==============================] - 0s 6ms/step - loss: 0.2516 - accuracy: 0.9254
Epoch 10/100
```

Fig-9: CNN Classifier

```
In [17]: print("The test loss is :",test_loss, "\nThe test accuracy is :",test_acc)

The test loss is : 0.548119306564331
The test accuracy is : 0.8935396075248718
```

Fig-10: CNN Model output

APPLICATIONS

1. **Mall:**

- Music is played continuously in the malls, and selection of right music to be played is hectic as well as time consuming work.
- So here, our system helps to choose the song according to any occasion or event.

2. **Restaurant:**

- In a restaurant, choosing the right music is an important task when it comes to various occasions as per customer’s demand; our system will help to choose a particular genre song for the same.

3. **Airport:**

- Music is played in the airports for the entertainment of people as they wait for hours due to various reasons, so our system will help to choose the song as per the requirements.

FUTURE MILESTONES

Step 1
Feature Extraction

Step 2
Training and Testing Dataset

Step 3
Classification of music into different genres

Step 4
Measuring Accuracy

Step 5
Measuring Error

Step 6
Prediction of new music

🎼 THANK YOU 🎼