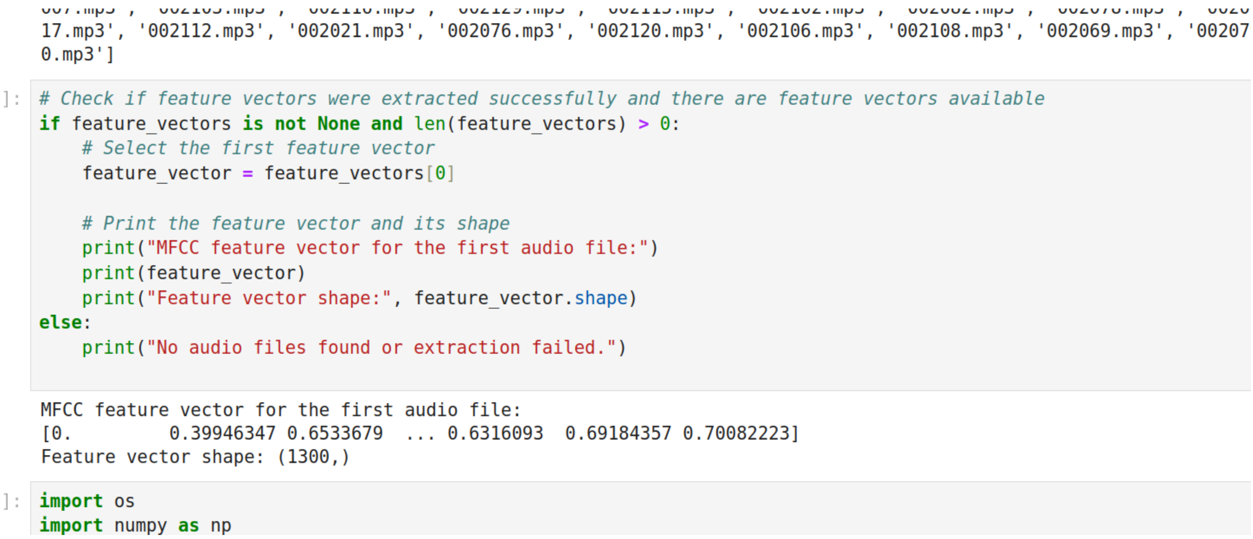
Talha Iftikhar   (22I-1701)
Abdul Sami Khan (22I-2062)
Maaz Ahmed (22I-2057)

REPORT:
The very first thing we did is that we downloaded the data that contained all the audios. That was approx 95 GB and after extraction it became 105 GB. Then we started working on it. The first thing we did was contain feature vectors of every audio.we did it with help of libs os, numpy, pandas , librosa, sklearn,sklearn.decomposition PCA.
Here is the justification.



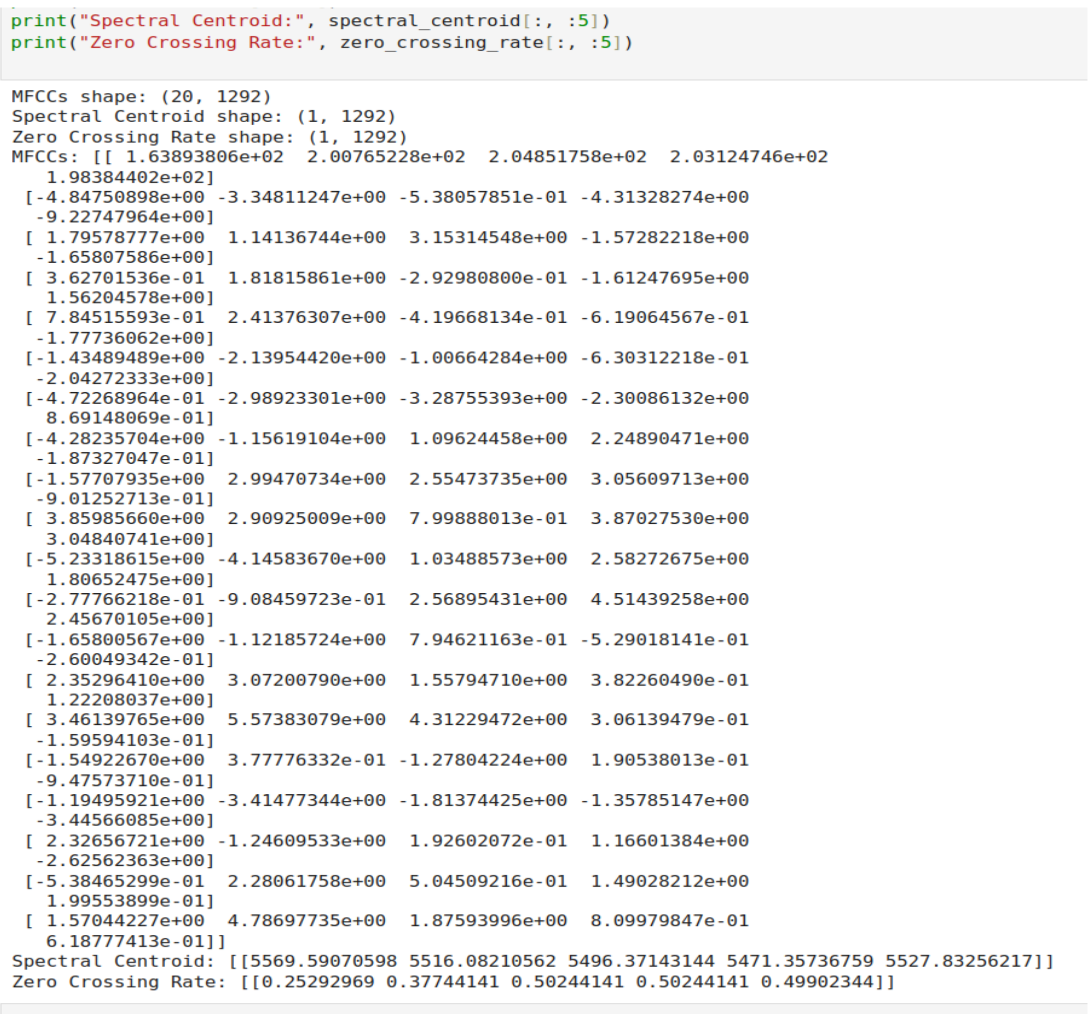
```
17.mp3', '002112.mp3', '002021.mp3', '002076.mp3', '002120.mp3', '002106.mp3', '002108.mp3', '002069.mp3', '00207
0.mp3']
```

```
]: # Check if feature vectors were extracted successfully and there are feature vectors available
   if feature_vectors is not None and len(feature_vectors) > 0:
       # Select the first feature vector
       feature_vector = feature_vectors[0]

       # Print the feature vector and its shape
       print("MFCC feature vector for the first audio file:")
       print(feature_vector)
       print("Feature vector shape:", feature_vector.shape)
   else:
       print("No audio files found or extraction failed.")
```

```
MFCC feature vector for the first audio file:
[0.         0.39946347 0.6533679  ... 0.6316093  0.69184357 0.70082223]
Feature vector shape: (1300,)
```

```
]: import os
   import numpy as np
```

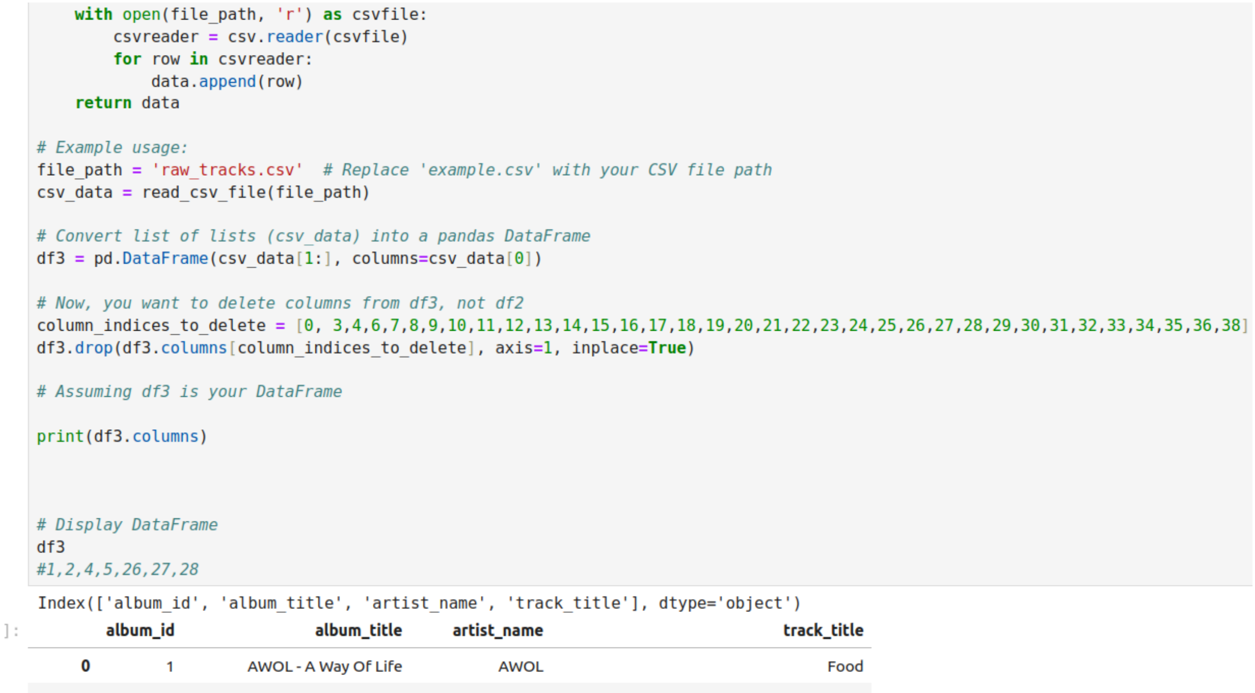
Moreover we also checked the feature vectors in np array and that is



```python
print("Spectral Centroid:", spectral_centroid[:, :5])
print("Zero Crossing Rate:", zero_crossing_rate[:, :5])
```

```
MFCCs shape: (20, 1292)
Spectral Centroid shape: (1, 1292)
Zero Crossing Rate shape: (1, 1292)
MFCCs: [[ 1.63893806e+02  2.00765228e+02  2.04851758e+02  2.03124746e+02
   1.98384402e+02]
 [-4.84750898e+00 -3.34811247e+00 -5.38057851e-01 -4.31328274e+00
  -9.22747964e+00]
 [ 1.79578777e+00  1.14136744e+00  3.15314548e+00 -1.57282218e+00
  -1.65807586e+00]
 [ 3.62701536e-01  1.81815861e+00 -2.92980800e-01 -1.61247695e+00
   1.56204578e+00]
 [ 7.84515593e-01  2.41376307e+00 -4.19668134e-01 -6.19064567e-01
  -1.77736062e+00]
 [-1.43489489e+00 -2.13954420e+00 -1.00664284e+00 -6.30312218e-01
  -2.04272333e+00]
 [-4.72268964e-01 -2.98923301e+00 -3.28755393e+00 -2.30086132e+00
   8.69148069e-01]
 [-4.28235704e+00 -1.15619104e+00  1.09624458e+00  2.24890471e+00
  -1.87327047e-01]
 [-1.57707935e+00  2.99470734e+00  2.55473735e+00  3.05609713e+00
  -9.01252713e-01]
 [ 3.85985660e+00  2.90925009e+00  7.99888013e-01  3.87027530e+00
   3.04840741e+00]
 [-5.23318615e+00 -4.14583670e+00  1.03488573e+00  2.58272675e+00
   1.80652475e+00]
 [-2.77766218e-01 -9.08459723e-01  2.56895431e+00  4.51439258e+00
   2.45670105e+00]
 [-1.65800567e+00 -1.12185724e+00  7.94621163e-01 -5.29018141e-01
  -2.60049342e-01]
 [ 2.35296410e+00  3.07200790e+00  1.55794710e+00  3.82260490e-01
   1.22208037e+00]
 [ 3.46139765e+00  5.57383079e+00  4.31229472e+00  3.06139479e-01
  -1.59594103e-01]
 [-1.54922670e+00  3.77776332e-01 -1.27804224e+00  1.90538013e-01
  -9.47573710e-01]
 [-1.19495921e+00 -3.41477344e+00 -1.81374425e+00 -1.35785147e+00
  -3.44566085e+00]
 [ 2.32656721e+00 -1.24609533e+00  1.92602072e-01  1.16601384e+00
  -2.62562363e+00]
 [-5.38465299e-01  2.28061758e+00  5.04509216e-01  1.49028212e+00
   1.99553899e-01]
 [ 1.57044227e+00  4.78697735e+00  1.87593996e+00  8.09979847e-01
   6.18777413e-01]]
Spectral Centroid: [[5569.59070598 5516.08210562 5496.37143144 5471.35736759 5527.83256217]]
Zero Crossing Rate: [[0.25292969 0.37744141 0.50244141 0.50244141 0.49902344]]
```

After we clarify that feature vectors are correct then we jump to keep tracking the 300MB data that is fma meta_data.zip we checked all the columns of every csv files in that meta data and then we drop all the irrelevant columns



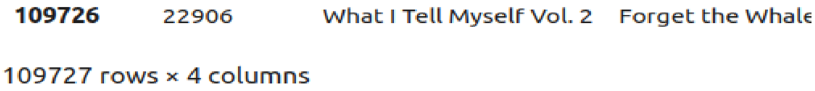
```python
    with open(file_path, 'r') as csvfile:
        csvreader = csv.reader(csvfile)
        for row in csvreader:
            data.append(row)
    return data

# Example usage:
file_path = 'raw_tracks.csv'  # Replace 'example.csv' with your CSV file path
csv_data = read_csv_file(file_path)

# Convert list of lists (csv_data) into a pandas DataFrame
df3 = pd.DataFrame(csv_data[1:], columns=csv_data[0])

# Now, you want to delete columns from df3, not df2
column_indices_to_delete = [0, 3,4,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,38]
df3.drop(df3.columns[column_indices_to_delete], axis=1, inplace=True)

# Assuming df3 is your DataFrame

print(df3.columns)



# Display DataFrame
df3
#1,2,4,5,26,27,28
```

```
Index(['album_id', 'album_title', 'artist_name', 'track_title'], dtype='object')
```

]:

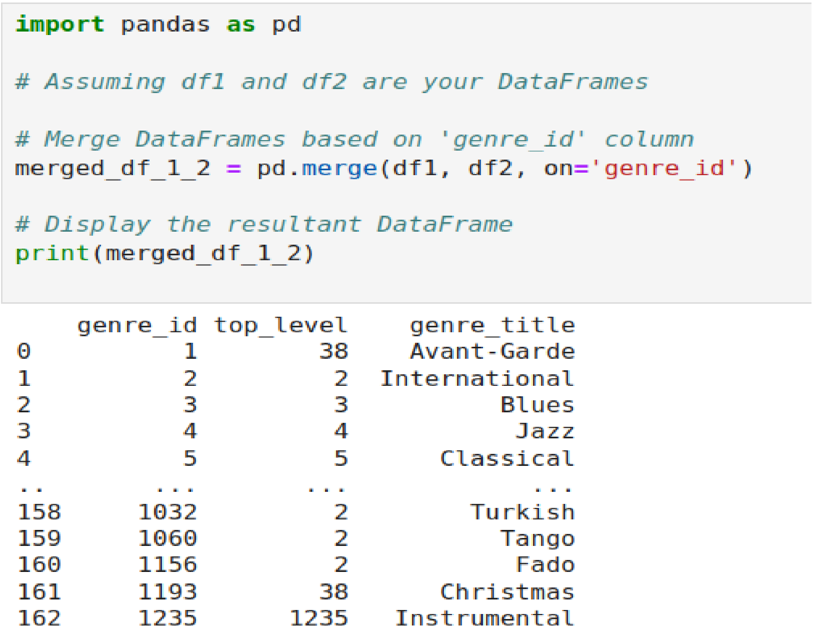
	album_id	album_title	artist_name	track_title
0	1	AWOL - A Way Of Life	AWOL	Food

Here is the justification for which we drop all the irrelevant columns from every csv file.



109726	22906	What I Tell Myself Vol. 2	Forget the Whale

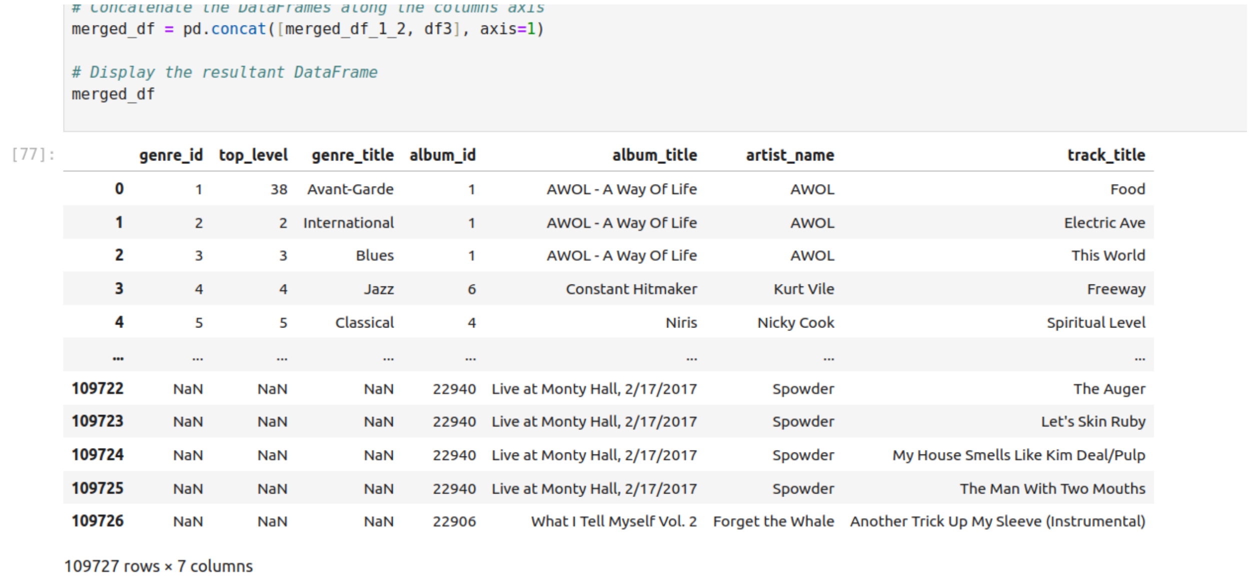
109727 rows × 4 columns



```python
import pandas as pd

# Assuming df1 and df2 are your DataFrames

# Merge DataFrames based on 'genre_id' column
merged_df_1_2 = pd.merge(df1, df2, on='genre_id')

# Display the resultant DataFrame
print(merged_df_1_2)
```

```
    genre_id top_level     genre_title
0          1        38     Avant-Garde
1          2         2   International
2          3         3           Blues
3          4         4            Jazz
4          5         5       Classical
..       ...       ...             ...
158     1032         2         Turkish
159     1060         2           Tango
160     1156         2            Fado
161     1193        38       Christmas
162     1235      1235    Instrumental
```

The next we made a resultant data frame that is here



```
# concatenate the DataFrames along the columns axis
merged_df = pd.concat([merged_df_1_2, df3], axis=1)

# Display the resultant DataFrame
merged_df
```

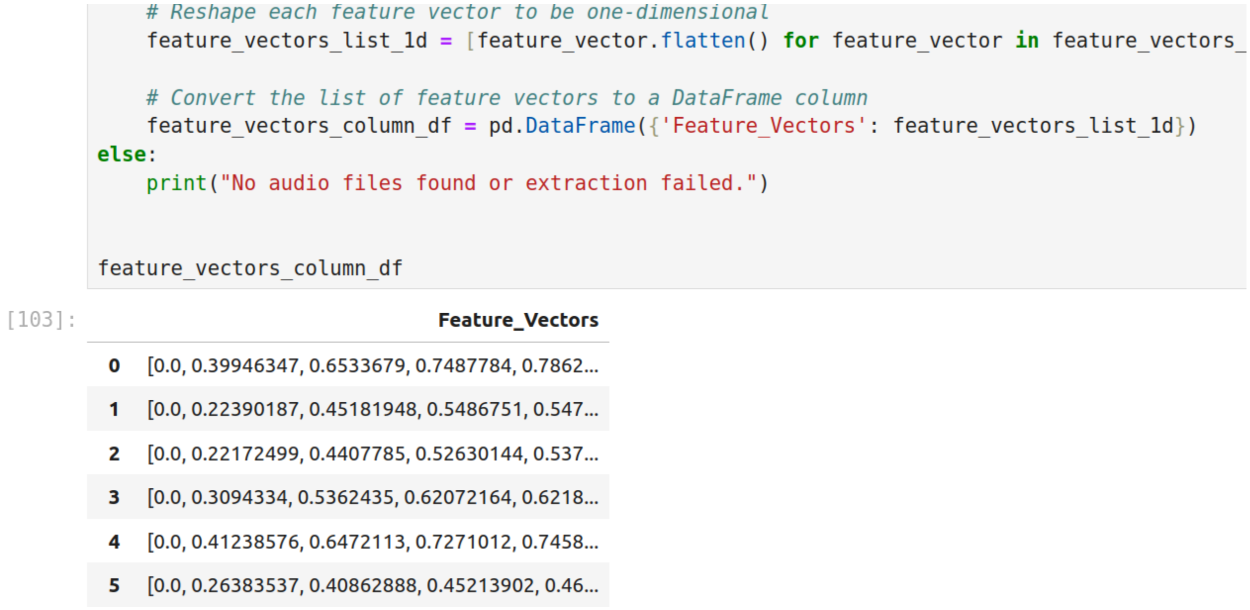
[77]:

	genre_id	top_level	genre_title	album_id	album_title	artist_name	track_title
0	1	38	Avant-Garde	1	AWOL - A Way Of Life	AWOL	Food
1	2	2	International	1	AWOL - A Way Of Life	AWOL	Electric Ave
2	3	3	Blues	1	AWOL - A Way Of Life	AWOL	This World
3	4	4	Jazz	6	Constant Hitmaker	Kurt Vile	Freeway
4	5	5	Classical	4	Niris	Nicky Cook	Spiritual Level
...	...	...	...	...	...	...	...
109722	NaN	NaN	NaN	22940	Live at Monty Hall, 2/17/2017	Spowder	The Auger
109723	NaN	NaN	NaN	22940	Live at Monty Hall, 2/17/2017	Spowder	Let's Skin Ruby
109724	NaN	NaN	NaN	22940	Live at Monty Hall, 2/17/2017	Spowder	My House Smells Like Kim Deal/Pulp
109725	NaN	NaN	NaN	22940	Live at Monty Hall, 2/17/2017	Spowder	The Man With Two Mouths
109726	NaN	NaN	NaN	22906	What I Tell Myself Vol. 2	Forget the Whale	Another Trick Up My Sleeve (Instrumental)

109727 rows × 7 columns

Then all the preprocessing is being done which was necessary for the data.
The next part was to make a key value pair.

The next this we did was to convert 2d np array to 1 d and into list

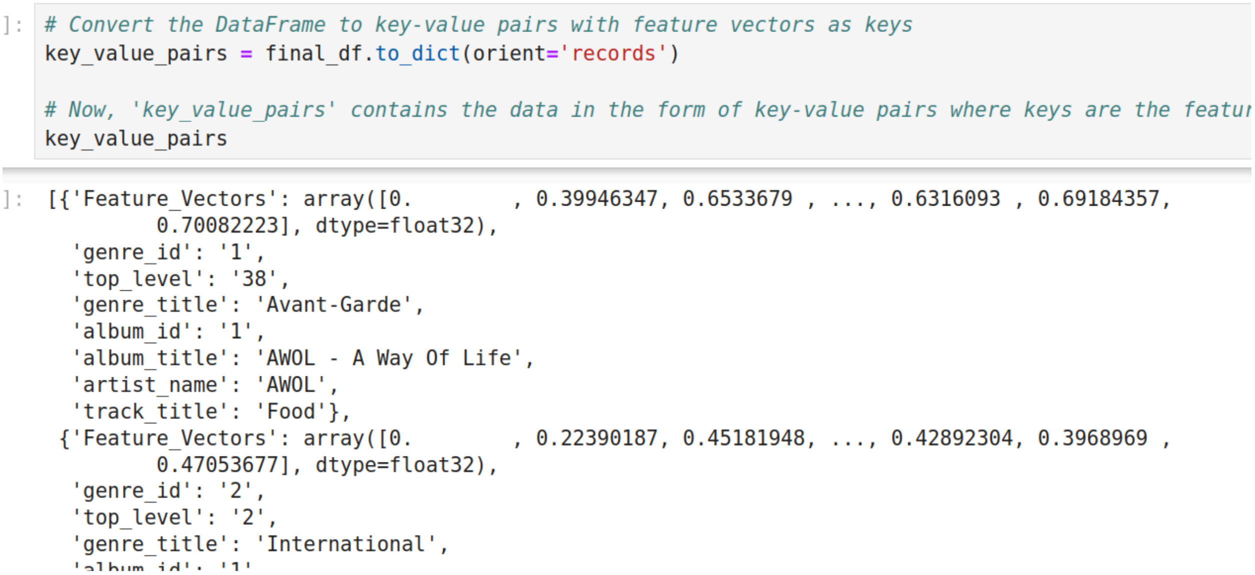


```
    # Reshape each feature vector to be one-dimensional
    feature_vectors_list_1d = [feature_vector.flatten() for feature_vector in feature_vectors_

    # Convert the list of feature vectors to a DataFrame column
    feature_vectors_column_df = pd.DataFrame({'Feature_Vectors': feature_vectors_list_1d})
else:
    print("No audio files found or extraction failed.")


feature_vectors_column_df
```

[103]:

	Feature_Vectors
0	[0.0, 0.39946347, 0.6533679, 0.7487784, 0.7862...
1	[0.0, 0.22390187, 0.45181948, 0.5486751, 0.547...
2	[0.0, 0.22172499, 0.4407785, 0.52630144, 0.537...
3	[0.0, 0.3094334, 0.5362435, 0.62072164, 0.6218...
4	[0.0, 0.41238576, 0.6472113, 0.7271012, 0.7458...
5	[0.0, 0.26383537, 0.40862888, 0.45213902, 0.46...

Key value pairs



```
]: # Convert the DataFrame to key-value pairs with feature vectors as keys
   key_value_pairs = final_df.to_dict(orient='records')

   # Now, 'key_value_pairs' contains the data in the form of key-value pairs where keys are the featur
   key_value_pairs
```

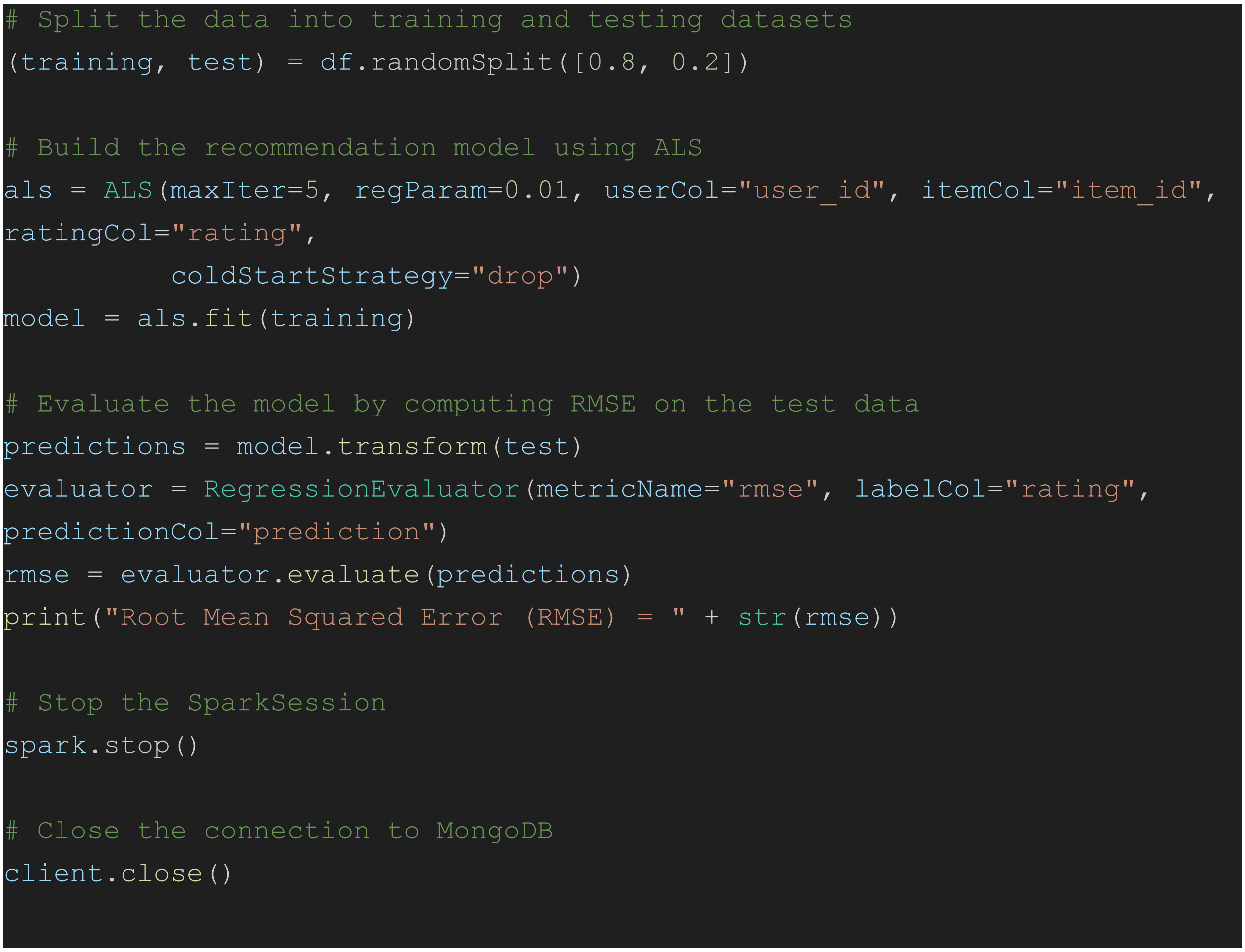
```
]: [{'Feature_Vectors': array([0.        , 0.39946347, 0.6533679 , ..., 0.6316093 , 0.69184357,
            0.70082223], dtype=float32),
     'genre_id': '1',
     'top_level': '38',
     'genre_title': 'Avant-Garde',
     'album_id': '1',
     'album_title': 'AWOL - A Way Of Life',
     'artist_name': 'AWOL',
     'track_title': 'Food'},
    {'Feature_Vectors': array([0.        , 0.22390187, 0.45181948, ..., 0.42892304, 0.3968969 ,
            0.47053677], dtype=float32),
     'genre_id': '2',
     'top_level': '2',
     'genre_title': 'International',
     'album_id': '1'
```

Now we will upload data to mongo db

```
from t4 import key_value_pairs
from pymongo import MongoClient
from pyspark.sql import SparkSession
from pyspark.ml.recommendation import ALS
from pyspark.ml.evaluation import RegressionEvaluator

# Connect to MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['your_database']
collection = db['key_value_pairs']  # Collection to store key-value pairs
```

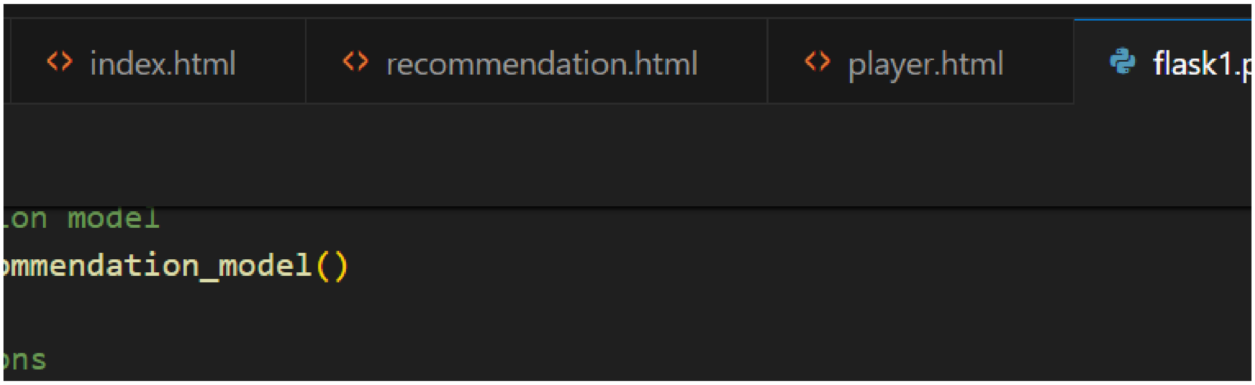
Applying pyspark ML algorithm

```
# Read data from MongoDB into a DataFrame
spark = SparkSession.builder \
    .appName("Music Recommendation") \
    .config("spark.mongodb.input.uri",
"mongodb://localhost:27017/your_database.key_value_pairs") \
    .config("spark.mongodb.output.uri",
"mongodb://localhost:27017/your_database.key_value_pairs") \
    .getOrCreate()

df = spark.read.format("com.mongodb.spark.sql.DefaultSource").load()
```



```
# Split the data into training and testing datasets
(training, test) = df.randomSplit([0.8, 0.2])

# Build the recommendation model using ALS
als = ALS(maxIter=5, regParam=0.01, userCol="user_id", itemCol="item_id",
ratingCol="rating",
          coldStartStrategy="drop")
model = als.fit(training)

# Evaluate the model by computing RMSE on the test data
predictions = model.transform(test)
evaluator = RegressionEvaluator(metricName="rmse", labelCol="rating",
predictionCol="prediction")
rmse = evaluator.evaluate(predictions)
print("Root Mean Squared Error (RMSE) = " + str(rmse))

# Stop the SparkSession
spark.stop()

# Close the connection to MongoDB
client.close()
```

We built the recommendation model using ALS
Later we created html css webpage and used Flask to display the outputs



```
.on model
ommendation_model()

ons
```

The main difficulties we faced during this project was with Apache Spark and its installation as it kept throwing the same error.