```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save &
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/mxmh-survey-results/mxmh_survey_results.csv
```

```
df = pd.read_csv('/kaggle/input/mxmh-survey-results/mxmh_survey_results.csv')
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
```

```
Nanvalues = df.isna().sum()
```

```
print(Nanvalues)
```

```
Timestamp                       0
Age                             1
Primary streaming service       1
Hours per day                   0
While working                   3
Instrumentalist                 4
Composer                        1
Fav genre                       0
Exploratory                     0
Foreign languages               4
BPM                           107
Frequency [Classical]           0
Frequency [Country]             0
Frequency [EDM]                 0
Frequency [Folk]                0
Frequency [Gospel]              0
Frequency [Hip hop]             0
Frequency [Jazz]                0
Frequency [K pop]               0
Frequency [Latin]               0
Frequency [Lofi]                0
Frequency [Metal]               0
Frequency [Pop]                 0
Frequency [R&B]                 0
Frequency [Rap]                 0
Frequency [Rock]                0
Frequency [Video game music]    0
Anxiety                         0
Depression                      0
Insomnia                        0
OCD                             0
Music effects                   8
Permissions                     0
dtype: int64
```

```
df_clean = df.dropna()
```

```
print(df_clean.isna())
```

```
     Timestamp    Age  Primary streaming service  Hours per day  \
2        False  False                      False          False
3        False  False                      False          False
4        False  False                      False          False
5        False  False                      False          False
6        False  False                      False          False
..         ...    ...                        ...            ...
731      False  False                      False          False
732      False  False                      False          False
733      False  False                      False          False
734      False  False                      False          False
735      False  False                      False          False

     While working  Instrumentalist  Composer  Fav genre  Exploratory  \
2            False            False     False      False        False
3            False            False     False      False        False
4            False            False     False      False        False
5            False            False     False      False        False
6            False            False     False      False        False
..             ...              ...       ...        ...          ...
731          False            False     False      False        False
732          False            False     False      False        False
733          False            False     False      False        False
734          False            False     False      False        False
735          False            False     False      False        False

     Foreign languages  ...  Frequency [R&B]  Frequency [Rap]  \
2                False  ...            False            False
3                False  ...            False            False
4                False  ...            False            False
5                False  ...            False            False
6                False  ...            False            False
..                 ...  ...              ...              ...
731              False  ...            False            False
732              False  ...            False            False
733              False  ...            False            False
734              False  ...            False            False
735              False  ...            False            False

     Frequency [Rock]  Frequency [Video game music]  Anxiety  Depression  \
2               False                         False    False       False
3               False                         False    False       False
4               False                         False    False       False
5               False                         False    False       False
6               False                         False    False       False
..                ...                           ...      ...         ...
731             False                         False    False       False
732             False                         False    False       False
733             False                         False    False       False
734             False                         False    False       False
735             False                         False    False       False

     Insomnia    OCD  Music effects  Permissions
2       False  False          False        False
3       False  False          False        False
4       False  False          False        False
5       False  False          False        False
6       False  False          False        False
```

```
df_clean = df_clean.reset_index(drop=True)
```

```
df_clean.to_csv('clean_MUSICdataset.csv', index=False)
```

```
df_clean.head(10)
```

	Timestamp	Age	Primary streaming service	Hours per day	While working	Instrumentalist	Composer	Fav genre	Exploratory	Foreign languages	...	Frequency [R&B]	Frequency [Rap]	Freq [
0	8/27/2022 21:28:18	18.0	Spotify	4.0	No	No	No	Video game music	No	Yes	...	Never	Rarely	
1	8/27/2022 21:40:40	61.0	YouTube Music	2.5	Yes	No	Yes	Jazz	Yes	Yes	...	Sometimes	Never	
2	8/27/2022 21:54:47	18.0	Spotify	4.0	Yes	No	No	R&B	Yes	No	...	Very frequently	Very frequently	
3	8/27/2022 21:56:50	18.0	Spotify	5.0	Yes	Yes	Yes	Jazz	Yes	Yes	...	Very frequently	Very frequently	freq
4	8/27/2022 22:00:29	18.0	YouTube Music	3.0	Yes	Yes	No	Video game music	Yes	Yes	...	Rarely	Never	
5	8/27/2022 22:18:59	21.0	Spotify	1.0	Yes	No	No	K pop	Yes	Yes	...	Sometimes	Rarely	

```
df_clean['BPM'].head(10)
```

```
0    132.0
1     84.0
2    107.0
3     86.0
4     66.0
5     95.0
6     94.0
7    155.0
8    125.0
9     88.0
Name: BPM, dtype: float64
```

```
df_clean.columns
```

```
Index(['Timestamp', 'Age', 'Primary streaming service', 'Hours per day',
       'While working', 'Instrumentalist', 'Composer', 'Fav genre',
       'Exploratory', 'Foreign languages', 'BPM', 'Frequency [Classical]',
       'Frequency [Country]', 'Frequency [EDM]', 'Frequency [Folk]',
       'Frequency [Gospel]', 'Frequency [Hip hop]', 'Frequency [Jazz]',
       'Frequency [K pop]', 'Frequency [Latin]', 'Frequency [Lofi]',
       'Frequency [Metal]', 'Frequency [Pop]', 'Frequency [R&B]',
       'Frequency [Rap]', 'Frequency [Rock]', 'Frequency [Video game music]',
       'Anxiety', 'Depression', 'Insomnia', 'OCD', 'Music effects',
       'Permissions'],
      dtype='object')
```

```
# find the key of the music effects columns
df_clean['Music effects'].unique()
```

```
array(['No effect', 'Improve', 'Worsen'], dtype=object)
```

```
df_ML = df_clean.drop(['Timestamp','Instrumentalist', 'Composer', 'Exploratory','Permissions'],axis=1)
```

```
df_ML.head(10)
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	Foreign languages	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	...	Frequency [Pop]	Frequency [R&B]	Frequency [Rap]	Fre
0	18.0	Spotify	4.0	No	Video game music	Yes	132.0	Never	Never	Very frequently	...	Rarely	Never	Rarely	
1	61.0	YouTube Music	2.5	Yes	Jazz	Yes	84.0	Sometimes	Never	Never	...	Sometimes	Sometimes	Never	
2	18.0	Spotify	4.0	Yes	R&B	No	107.0	Never	Never	Rarely	...	Sometimes	Very frequently	Very frequently	
3	18.0	Spotify	5.0	Yes	Jazz	Yes	86.0	Rarely	Sometimes	Never	...	Very frequently	Very frequently	Very frequently	fr
4	18.0	YouTube Music	3.0	Yes	Video game music	Yes	66.0	Sometimes	Never	Rarely	...	Rarely	Rarely	Never	
5	21.0	Spotify	1.0	Yes	K pop	Yes	95.0	Never	Never	Rarely	...	Sometimes	Sometimes	Rarely	

```
# map the music effects to numeric values
Effectmapping = {'No effect': 0, 'Improve': 1, 'Worsen': 2}


df_ML['Music effects'] = df_ML['Music effects'].map(Effectmapping)
```

8	19.0	[illegible]	8.0	Yes	EDM	No	125.0	Rarely	Never	[illegible]	...	Rarely	Rarely	Sometimes	

```
df_ML.head(10)
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	Foreign languages	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	...	Frequency [Pop]	Frequency [R&B]	Frequency [Rap]	Fre
0	18.0	Spotify	4.0	No	Video game music	Yes	132.0	Never	Never	Very frequently	...	Rarely	Never	Rarely	
1	61.0	YouTube Music	2.5	Yes	Jazz	Yes	84.0	Sometimes	Never	Never	...	Sometimes	Sometimes	Never	
2	18.0	Spotify	4.0	Yes	R&B	No	107.0	Never	Never	Rarely	...	Sometimes	Very frequently	Very frequently	
3	18.0	Spotify	5.0	Yes	Jazz	Yes	86.0	Rarely	Sometimes	Never	...	Very frequently	Very frequently	Very frequently	fr
4	18.0	YouTube Music	3.0	Yes	Video game music	Yes	66.0	Sometimes	Never	Rarely	...	Rarely	Rarely	Never	
5	21.0	Spotify	1.0	Yes	K pop	Yes	95.0	Never	Never	Rarely	...	Sometimes	Sometimes	Rarely	
6	19.0	Spotify	6.0	Yes	Rock	No	94.0	Never	Very frequently	Never	...	Never	Never	Never	fr
7	18.0	I do not use a streaming service.	1.0	Yes	R&B	Yes	155.0	Rarely	Rarely	Rarely	...	Sometimes	Sometimes	Rarely	So
8	19.0	YouTube Music	8.0	Yes	EDM	No	125.0	Rarely	Never	Very frequently	...	Rarely	Rarely	Sometimes	
9	19.0	Spotify	2.0	Yes	Country	No	88.0	Never	Very frequently	Rarely	...	Rarely	Never	Very frequently	

10 rows × 28 columns

```
df_clean['Frequency [EDM]'].unique()
```

```
array(['Very frequently', 'Never', 'Rarely', 'Sometimes'], dtype=object)
```

```
df_clean['Fav genre'].unique()
```

```
array(['Video game music', 'Jazz', 'R&B', 'K pop', 'Rock', 'EDM',
       'Country', 'Hip hop', 'Rap', 'Pop', 'Classical', 'Metal', 'Folk',
       'Lofi', 'Gospel', 'Latin'], dtype=object)
```

```
Effectmapping = {
    'Video game music': 0, 'Jazz': 1, 'R&B': 2, 'K pop': 3, 'Rock': 4, 'EDM': 5,
    'Country': 6, 'Hip hop': 7, 'Rap': 8, 'Pop': 9, 'Classical': 10, 'Metal': 11,
    'Folk': 12, 'Lofi': 13, 'Gospel': 14, 'Latin': 15
}

# Map 'Fav genre' column
df_ML['Fav genre'] = df_ML['Fav genre'].map(Effectmapping)
```

```
# must create a ml for the  frequence of the columns to be able to predict the type of music a person would listen to
# can not hot encode must map instead
Effectmapping = {'Very frequently': 0, 'Never': 1, 'Rarely': 2, 'Sometimes': 3}

encodedColumn = ['Frequency [Classical]', 'Frequency [Country]', 'Frequency [EDM]',
                 'Frequency [Folk]', 'Frequency [Gospel]', 'Frequency [Hip hop]',
                 'Frequency [Jazz]', 'Frequency [K pop]', 'Frequency [Latin]',
                 'Frequency [Lofi]', 'Frequency [Metal]', 'Frequency [Pop]',
                 'Frequency [R&B]', 'Frequency [Rap]', 'Frequency [Rock]', 'Frequency [Video game music]']

# Iterate through columns and map values
for column in encodedColumn:
    df_ML[column] = df_ML[column].map(Effectmapping)
```

```
df_clean['Primary streaming service'].unique()
```

```
array(['Spotify', 'YouTube Music', 'I do not use a streaming service.',
       'Apple Music', 'Other streaming service', 'Pandora'], dtype=object)
```

```
# map the music effects to numeric values
Effectmapping = {'Spotify': 0, 'YouTube Music': 1, 'I do not use a streaming service': 2,'Apple Music': 3,'Other streaming service': 4,'Pando
df_ML['Primary streaming service'] = df_ML['Primary streaming service'].map(Effectmapping)
```

```
Effectmapping = {'No':0 , 'Yes': 1}
df_ML['While working'] = df_ML['While working'].map(Effectmapping)
```

```
df_ML.sample()
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	Foreign languages	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	...	Frequency [Pop]	Frequency [R&B]	Frequency [Rap]	Fr
141	49.0	4.0	1.0	1	5	No	156.0	3	1	0	...	3	2	2	

1 rows × 28 columns

```
# answers provided to the survey may not be totally an accurate picture of the facts
```

```
df_ML.drop(['Foreign languages'],axis = 1)
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	Frequency [Folk]	...	Frequency [Pop]	Frequency [R&B]	Frequency [Rap]	Fr
0	18.0	0.0	4.0	0	0	132.0	1	1	0	1	...	2	1	2	
1	61.0	1.0	2.5	1	1	84.0	3	1	1	2	...	3	3	1	
2	18.0	0.0	4.0	1	2	107.0	1	1	2	1	...	3	0	0	
3	18.0	0.0	5.0	1	1	86.0	2	3	1	1	...	0	0	0	
4	18.0	1.0	3.0	1	0	66.0	3	1	2	3	...	2	2	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	

```
df_ML.columns
```

```
Index(['Age', 'Primary streaming service', 'Hours per day', 'While working',
       'Fav genre', 'Foreign languages', 'BPM', 'Frequency [Classical]',
       'Frequency [Country]', 'Frequency [EDM]', 'Frequency [Folk]',
       'Frequency [Gospel]', 'Frequency [Hip hop]', 'Frequency [Jazz]',
       'Frequency [K pop]', 'Frequency [Latin]', 'Frequency [Lofi]',
       'Frequency [Metal]', 'Frequency [Pop]', 'Frequency [R&B]',
       'Frequency [Rap]', 'Frequency [Rock]', 'Frequency [Video game music]',
       'Anxiety', 'Depression', 'Insomnia', 'OCD', 'Music effects'],
      dtype='object')
```

```
df_ML.describe(exclude="number")
```

	Foreign languages
count	616
unique	2
top	Yes
freq	347

```
df_ML = df_ML.drop(['Foreign languages'],axis=1)
```

```
df_ML.describe()
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	Frequency [Folk]	...	Frequ [
count	616.000000	560.000000	616.000000	616.000000	616.000000	6.160000e+02	616.000000	616.000000	616.000000	616.000000	...	616.00
mean	24.792208	0.721429	3.702435	0.795455	6.555195	1.623500e+06	1.767857	1.547078	1.517857	1.612013	...	1.50
std	11.658515	1.353814	3.071961	0.403697	3.653292	4.029114e+07	1.002277	0.832896	0.960863	0.925054	...	1.31
min	10.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000000	0.000000	...	0.00
25%	18.000000	0.000000	2.000000	1.000000	4.000000	1.000000e+02	1.000000	1.000000	1.000000	1.000000	...	0.00
50%	21.000000	0.000000	3.000000	1.000000	6.000000	1.200000e+02	2.000000	1.000000	1.000000	2.000000	...	2.00
75%	27.000000	1.000000	5.000000	1.000000	10.000000	1.440000e+02	3.000000	2.000000	2.000000	2.000000	...	3.00
max	89.000000	5.000000	24.000000	1.000000	15.000000	1.000000e+09	3.000000	3.000000	3.000000	3.000000	...	3.00

8 rows × 27 columns

```
NanvaluesML = df_ML.isna().sum()
```

```
print(NanvaluesML)
```

```
Age                           0
Primary streaming service    56
Hours per day                 0
While working                 0
Fav genre                     0
```

```
BPM                           0
Frequency [Classical]         0
Frequency [Country]           0
Frequency [EDM]               0
Frequency [Folk]              0
Frequency [Gospel]            0
Frequency [Hip hop]           0
Frequency [Jazz]              0
Frequency [K pop]             0
Frequency [Latin]             0
Frequency [Lofi]              0
Frequency [Metal]             0
Frequency [Pop]               0
Frequency [R&B]               0
Frequency [Rap]               0
Frequency [Rock]              0
Frequency [Video game music]  0
Anxiety                       0
Depression                    0
Insomnia                      0
OCD                           0
Music effects                 0
dtype: int64
```

```
df_ML.dropna
```

```
<bound method DataFrame.dropna of       Age  Primary streaming service  Hours per day  While working  Fav genre  \
0    18.0                        0.0            4.0              0          0
1    61.0                        1.0            2.5              1          1
2    18.0                        0.0            4.0              1          2
3    18.0                        0.0            5.0              1          1
4    18.0                        1.0            3.0              1          0
..    ...                        ...            ...            ...        ...
611  17.0                        0.0            2.0              1          4
612  18.0                        0.0            1.0              1          9
613  19.0                        4.0            6.0              1          8
614  19.0                        0.0            5.0              1         10
615  29.0                        1.0            2.0              1          7

       BPM  Frequency [Classical]  Frequency [Country]  Frequency [EDM]  \
0    132.0                      1                    1                0
1     84.0                      3                    1                1
2    107.0                      1                    1                2
3     86.0                      2                    3                1
4     66.0                      3                    1                2
..     ...                    ...                  ...              ...
611  120.0                      0                    2                1
612  160.0                      2                    2                1
613  120.0                      2                    3                3
614  170.0                      0                    1                1
615   98.0                      3                    2                0

     Frequency [Folk]  ...  Frequency [Pop]  Frequency [R&B]  Frequency [Rap]  \
0                   1  ...                2                1                2
1                   2  ...                3                3                1
2                   1  ...                3                0                0
3                   1  ...                0                0                0
4                   3  ...                2                2                1
..                ...  ...              ...              ...              ...
611                 3  ...                0                1                2
612                 1  ...                0                1                1
613                 2  ...                3                3                3
614                 1  ...                1                1                1
615                 3  ...                3                0                0

     Frequency [Rock]  Frequency [Video game music]  Anxiety  Depression  \
0                   2                             0      7.0         7.0
1                   1                             1      9.0         7.0
2                   1                             2      7.0         2.0
3                   0                             1      8.0         8.0
4                   1                             3      4.0         8.0
..                ...                           ...      ...         ...
611                 0                             1      7.0         6.0
612                 3                             3      3.0         2.0
613                 2                             2      2.0         2.0
614                 1                             3      2.0         3.0
615                 0                             2      2.0         2.0

     Insomnia  OCD  Music effects
0        10.0  2.0              0
1         3.0  3.0              1
2         5.0  9.0              1
3         7.0  7.0              1
```

```
4          6.0  0.0              1
```

```
df_ML = df_ML.dropna()
```

```
df_MLC= df_ML.isna().sum()
df_MLC
```

```
Age                            0
Primary streaming service      0
Hours per day                  0
While working                  0
Fav genre                      0
BPM                            0
Frequency [Classical]          0
Frequency [Country]            0
Frequency [EDM]                0
Frequency [Folk]               0
Frequency [Gospel]             0
Frequency [Hip hop]            0
Frequency [Jazz]               0
Frequency [K pop]              0
Frequency [Latin]              0
Frequency [Lofi]               0
Frequency [Metal]              0
Frequency [Pop]                0
Frequency [R&B]                0
Frequency [Rap]                0
Frequency [Rock]               0
Frequency [Video game music]   0
Anxiety                        0
Depression                     0
Insomnia                       0
OCD                            0
Music effects                  0
dtype: int64
```

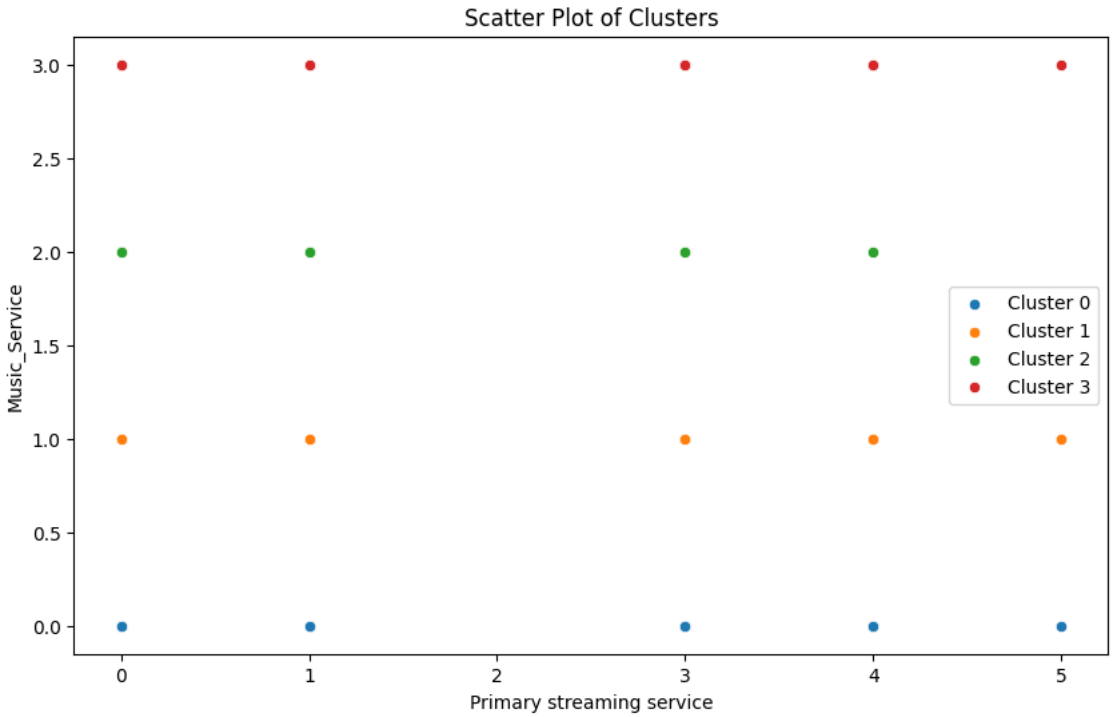
```
df_ML.head(50)
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	Frequency [Folk]	...	Frequency [Pop]	Frequency [R&B]	Frequency [Rap]	Fre
0	18.0	0.0	4.00	0	0	132.0	1	1	0	1	...	2	1	2	
1	61.0	1.0	2.50	1	1	84.0	3	1	1	2	...	3	3	1	
2	18.0	0.0	4.00	1	2	107.0	1	1	2	1	...	3	0	0	
3	18.0	0.0	5.00	1	1	86.0	2	3	1	1	...	0	0	0	
4	18.0	1.0	3.00	1	0	66.0	3	1	2	3	...	2	2	1	
5	21.0	0.0	1.00	1	3	95.0	1	1	2	1	...	3	3	2	
6	19.0	0.0	6.00	1	4	94.0	1	0	1	3	...	1	1	1	
8	19.0	1.0	8.00	1	5	125.0	2	1	0	1	...	2	2	3	
9	19.0	0.0	2.00	1	6	88.0	1	0	2	3	...	2	1	0	
10	18.0	0.0	4.00	1	1	148.0	0	2	1	1	...	3	1	1	
11	16.0	0.0	8.00	1	7	103.0	1	1	1	1	...	1	3	0	
12	16.0	0.0	12.00	1	7	120.0	2	1	3	2	...	3	2	3	
13	17.0	0.0	24.00	1	8	99.0	2	1	1	1	...	2	3	0	
14	15.0	0.0	3.00	0	7	120.0	1	1	1	1	...	2	3	0	
15	15.0	3.0	8.00	1	7	120.0	2	1	3	2	...	0	2	0	
16	17.0	0.0	4.00	1	8	125.0	1	2	2	1	...	3	1	0	
18	18.0	0.0	2.00	1	9	79.0	2	2	3	1	...	0	1	1	
19	16.0	4.0	3.00	1	4	84.0	2	2	1	2	...	1	3	2	
20	18.0	0.0	2.00	0	9	169.0	3	2	2	0	...	0	3	3	
21	14.0	0.0	12.00	1	4	136.0	3	3	2	2	...	0	0	0	
22	18.0	1.0	6.00	1	9	101.0	3	2	2	2	...	0	3	3	
23	17.0	0.0	2.00	1	9	126.0	1	0	2	2	...	0	2	1	
24	17.0	3.0	1.00	1	9	183.0	2	1	1	2	...	3	1	1	
25	19.0	0.0	2.00	1	10	120.0	0	1	1	1	...	0	1	1	
26	17.0	0.0	4.00	0	4	142.0	2	2	2	0	...	0	2	3	
27	16.0	0.0	1.00	1	10	75.0	0	1	1	2	...	1	1	1	
28	18.0	0.0	5.00	1	9	120.0	3	2	3	2	...	0	3	3	

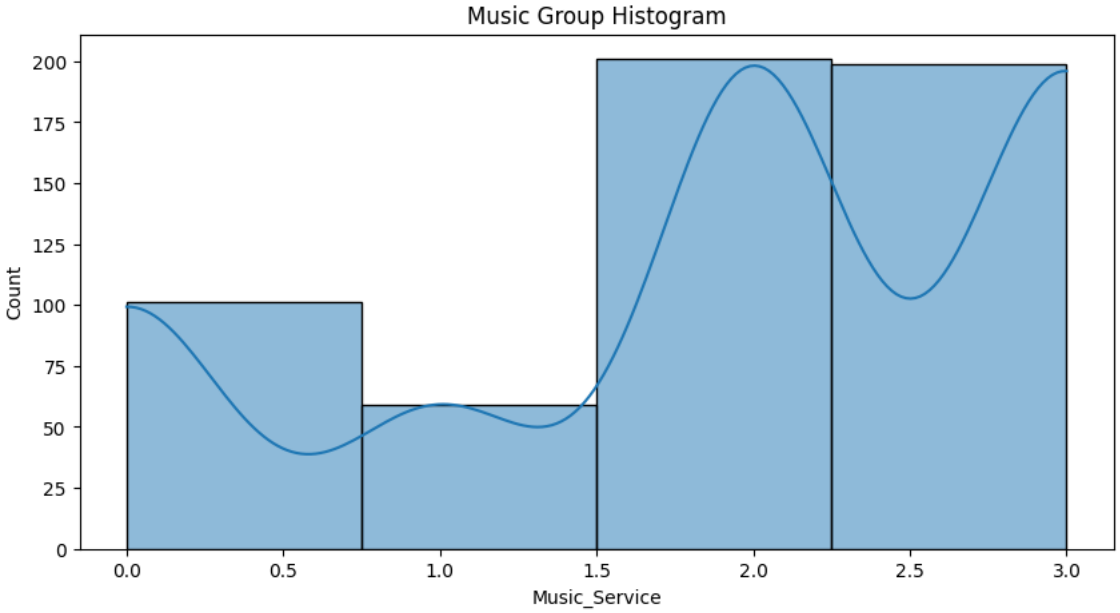
```
selected_features = df_ML[['Age','Hours per day', 'While working',
       'BPM', 'Frequency [Classical]',
       'Frequency [Country]', 'Frequency [EDM]', 'Frequency [Folk]',
       'Frequency [Gospel]', 'Frequency [Hip hop]', 'Frequency [Jazz]',
       'Frequency [K pop]', 'Frequency [Latin]', 'Frequency [Lofi]',
       'Frequency [Metal]', 'Frequency [Pop]', 'Frequency [R&B]',
       'Frequency [Rap]', 'Frequency [Rock]', 'Frequency [Video game music]']]


scaler = StandardScaler()
scaled_features = scaler.fit_transform(selected_features)


num_clusters = 4


kmeans = KMeans(n_clusters=num_clusters)
df_ML['Music_Service'] = kmeans.fit_predict(scaled_features)


plt.figure(figsize=(10, 6))

for cluster in range(num_clusters):
    cluster_data = df_ML[df_ML['Music_Service'] == cluster]
    sns.scatterplot(x='Primary streaming service', y='Music_Service', data=cluster_data, label=f'Cluster {cluster}')
```

```
plt.title('Scatter Plot of Clusters')
plt.legend()
plt.show()


plt.figure(figsize=(10, 5))
sns.histplot(data=df_ML, x='Music_Service', bins=num_clusters, kde=True, palette='viridis')
plt.title('Music Group Histogram')
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from
  warnings.warn(
/tmp/ipykernel_20/4098458283.py:27: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
  df_ML['Music_Service'] = kmeans.fit_predict(scaled_features)
```




```
/tmp/ipykernel_20/4098458283.py:42: UserWarning: Ignoring `palette` because no `hue` variable has been assigned.
  sns.histplot(data=df_ML, x='Music_Service', bins=num_clusters, kde=True, palette='viridis')
```



```
df_ML.head(10)
```

	Age	Primary streaming service	Hours per day	While working	Fav genre	BPM	Frequency [Classical]	Frequency [Country]	Frequency [EDM]	Frequency [Folk]	...	Frequency [R&B]	Frequency [Rap]	Frequency [Rock]	Fre
0	18.0	0.0	4.0	0	0	132.0	1	1	0	1	...	1	2	2	
1	61.0	1.0	2.5	1	1	84.0	3	1	1	2	...	3	1	1	
2	18.0	0.0	4.0	1	2	107.0	1	1	2	1	...	0	0	1	
3	18.0	0.0	5.0	1	1	86.0	2	3	1	1	...	0	0	0	
4	18.0	1.0	3.0	1	0	66.0	3	1	2	3	...	2	1	1	
5	21.0	0.0	1.0	1	3	95.0	1	1	2	1	...	3	2	1	
6	19.0	0.0	6.0	1	4	94.0	1	0	1	3	...	1	1	0	
8	19.0	1.0	8.0	1	5	125.0	2	1	0	1	...	2	3	2	
9	19.0	0.0	2.0	1	6	88.0	1	0	2	3	...	1	0	1	
10	18.0	0.0	4.0	1	1	148.0	0	2	1	1	...	1	1	3	

10 rows × 28 columns

```
df_ML.nunique()
```

```
Age                             52
Primary streaming service        5
Hours per day                   23
While working                    2
Fav genre                       16
BPM                            133
Frequency [Classical]            4
Frequency [Country]              4
Frequency [EDM]                  4
Frequency [Folk]                 4
Frequency [Gospel]               4
Frequency [Hip hop]              4
Frequency [Jazz]                 4
Frequency [K pop]                4
Frequency [Latin]                4
Frequency [Lofi]                 4
Frequency [Metal]                4
Frequency [Pop]                  4
Frequency [R&B]                  4
Frequency [Rap]                  4
Frequency [Rock]                 4
Frequency [Video game music]     4
Anxiety                         12
Depression                      12
Insomnia                        12
OCD                             13
Music effects                    3
Music_Service                    4
dtype: int64
```

```
df_ML.columns
```

```
Index(['Age', 'Primary streaming service', 'Hours per day', 'While working',
       'Fav genre', 'BPM', 'Frequency [Classical]', 'Frequency [Country]',
       'Frequency [EDM]', 'Frequency [Folk]', 'Frequency [Gospel]',
       'Frequency [Hip hop]', 'Frequency [Jazz]', 'Frequency [K pop]',
       'Frequency [Latin]', 'Frequency [Lofi]', 'Frequency [Metal]',
       'Frequency [Pop]', 'Frequency [R&B]', 'Frequency [Rap]',
       'Frequency [Rock]', 'Frequency [Video game music]', 'Anxiety',
       'Depression', 'Insomnia', 'OCD', 'Music effects', 'Music_Service'],
      dtype='object')
```

```
X = df_ML[['Age', 'Hours per day', 'While working', 'BPM', 'Frequency [Classical]', 'Frequency [Country]', 'Frequency [EDM]', 'Frequency [Fol
y = df_ML['Fav genre']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

```
Mean Squared Error: 11.977270153506216
```

```
X = df_ML[['Age', 'Hours per day', 'While working', 'BPM', ]]
y = df_ML['Fav genre']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


model = RandomForestRegressor(n_estimators=10, random_state=42)
model.fit(X_train, y_train)


y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")


last_20_data = df_ML.tail(20)
X_last_20 = last_20_data[['Age', 'Hours per day', 'While working', 'BPM', ]]
y_last_20_true = last_20_data['Fav genre']
y_last_20_pred = model.predict(X_last_20)
mse_last_20 = mean_squared_error(y_last_20_true, y_last_20_pred)
print(f"Mean Squared Error (Last 20 Data Points): {mse_last_20}")
```

```
Mean Squared Error: 15.75663125
Mean Squared Error (Last 20 Data Points): 4.397222222222221
```