Maven Music Data Analysis

Data Gathering

In [2]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import pandas as pd
import numpy as np
```

In [3]:
```
# Read in the customer data

customers = pd.read_csv(r"D:\DA\Done\Python\Udemy Course\Data\CSV\Dummy Data\Maven_Music_Customers.csv")
customers.head()
```

Out[3]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	3/13/23	Basic (Ads)	$2.99	NaN	NaN
1	5002	Aria Keys	Email: melodious.aria@email.edu	3/13/23	NaN	$2.99	NaN	NaN
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	3/13/23	NaN	$2.99	NaN	6/1/23
3	5267	Rock Bassett	Email: groovy.rock@email.com	3/20/23	Basic (Ads)	$2.99	NaN	NaN
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	3/20/23	NaN	$2.99	NaN	NaN

In [4]:
```
# Read in the listening_history

listening_history = pd.read_excel(r"D:\DA\Done\Python\Udemy Course\Data\CSV\Dummy Data\Maven_Music_Listening_history
listening_history.head()
```

Out[4]:

	Customer ID	Session ID	Audio Order	Audio ID	Audio Type
0	5001	100520	1	101	Song
1	5001	100520	2	102	Song
2	5001	100520	3	103	Song
3	5001	100520	4	104	Song
4	5001	100520	5	105	Song

In [5]:
```
# Read in the audio data

audio = pd.read_excel(r"D:\DA\Done\Python\Udemy Course\Data\CSV\Dummy Data\Maven_Music_Listening_history.xlsx", she
audio.head()
```

Out[5]:

	ID	Name	Genre	Popularity
0	Song-101	Dance All Night	Pop	1
1	Song-102	Unbreakable Beat	Pop	2
2	Song-103	Sunset Boulevard	Pop Music	5
3	Song-104	Glowing Hearts	Pop Music	10
4	Song-105	Pop Rocks	Pop Music	52

In [6]:
```
# Read in the session data

sessions = pd.read_excel(r"D:\DA\Done\Python\Udemy Course\Data\CSV\Dummy Data\Maven_Music_Listening_history.xlsx", 
sessions.head()
```

Out[6]:

	Session ID	Session Log In Time
0	100520	2023-03-13 18:29:00
1	100522	2023-03-13 22:15:00
2	100525	2023-03-14 10:01:00
3	100527	2023-03-13 14:14:00
4	100538	2023-03-21 12:23:00

Data Cleaning

In [7]:
```
# check the customers data type
customers.dtypes
```

Out[7]:
```
Customer ID            int64
Customer Name         object
Email                 object
Member Since          object
Subscription Plan     object
Subscription Rate     object
Discount?             object
Cancellation Date     object
dtype: object
```

In [8]:
```
# check the listening_history data type
listening_history.dtypes
```

Out[8]:
```
Customer ID      int64
Session ID       int64
Audio Order      int64
Audio ID         int64
Audio Type      object
dtype: object
```

In [9]:
```
# check the audio data type
audio.dtypes
```

Out[9]:
```
ID             object
Name           object
Genre          object
Popularity      int64
dtype: object
```

In [10]:
```
# check the sessions data type
sessions.dtypes
```

Out[10]:
```
Session ID                          int64
Session Log In Time    datetime64[ns]
dtype: object
```

Converting Data Types

In [11]:
```
customers.head()
```

Out[11]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	3/13/23	Basic (Ads)	$2.99	NaN	NaN
1	5002	Aria Keys	Email: melodious.aria@email.edu	3/13/23	NaN	$2.99	NaN	NaN
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	3/13/23	NaN	$2.99	NaN	6/1/23
3	5267	Rock Bassett	Email: groovy.rock@email.com	3/20/23	Basic (Ads)	$2.99	NaN	NaN
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	3/20/23	NaN	$2.99	NaN	NaN

In [12]:
```
#convert objects to numeric & datetime fields

customers['Member Since'] = pd.to_datetime(customers['Member Since'])
customers['Subscription Rate'] = pd.to_numeric(customers['Subscription Rate'].str.replace('$', ''))
customers['Cancellation Date'] = pd.to_datetime(customers['Cancellation Date'])
```

```
C:\Users\RONI\AppData\Local\Temp\ipykernel_8672\3005490285.py:4: FutureWarning: The default value of regex will ch
ange from True to False in a future version. In addition, single character regular expressions will *not* be treat
ed as literal strings when regex=True.
  customers['Subscription Rate'] = pd.to_numeric(customers['Subscription Rate'].str.replace('$', ''))
```

In [13]:
```
# check the new customers data type
customers.dtypes
```

Out[13]:
```
Customer ID                   int64
Customer Name                object
Email                        object
Member Since         datetime64[ns]
Subscription Plan            object
Subscription Rate           float64
Discount?                    object
Cancellation Date    datetime64[ns]
dtype: object
```

Resolve Data Issues

Data Minning

In [14]:
```
# find the Nan values in customers
customers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Customer ID        30 non-null     int64
 1   Customer Name      30 non-null     object
 2   Email              30 non-null     object
 3   Member Since       30 non-null     datetime64[ns]
 4   Subscription Plan  25 non-null     object
 5   Subscription Rate  30 non-null     float64
 6   Discount?          7 non-null      object
 7   Cancellation Date  13 non-null     datetime64[ns]
dtypes: datetime64[ns](2), float64(1), int64(1), object(4)
memory usage: 2.0+ KB
```

In [15]:
```
# find the Nan values in listening_history
listening_history.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 505 entries, 0 to 504
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Customer ID  505 non-null    int64
 1   Session ID   505 non-null    int64
 2   Audio Order  505 non-null    int64
 3   Audio ID     505 non-null    int64
 4   Audio Type   505 non-null    object
dtypes: int64(4), object(1)
memory usage: 19.9+ KB
```

In [16]:
```
# find the Nan values in audio
audio.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17 entries, 0 to 16
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   ID          17 non-null     object
 1   Name        17 non-null     object
 2   Genre       17 non-null     object
 3   Popularity  17 non-null     int64 
dtypes: int64(1), object(3)
memory usage: 672.0+ bytes
```

In [17]:
```
# find the Nan values in sessions
sessions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 2 columns):
 #   Column               Non-Null Count  Dtype         
---  ------               --------------  -----         
 0   Session ID           90 non-null     int64         
 1   Session Log In Time  90 non-null     datetime64[ns]
dtypes: datetime64[ns](1), int64(1)
memory usage: 1.5 KB
```

In [18]:
```
# customers df has null values in Subscription Plan, Discount, Cancellation Date
customers.head()
```

Out[18]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	NaN	NaT
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	NaN	2.99	NaN	NaT
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	NaN	2.99	NaN	2023-06-01
3	5267	Rock Bassett	Email: groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	NaN	NaT
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	NaN	2.99	NaN	NaT

In [19]:
```
# look into Subscription Plan all NaN Subscription Plan are $2.99
customers[customers['Subscription Plan'].isna()]
```

Out[19]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	NaN	2.99	NaN	NaT
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	NaN	2.99	NaN	2023-06-01
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	NaN	2.99	NaN	NaT
5	5404	Jazz Saxton	Email: jazzy.sax@email.com	2023-03-20	NaN	2.99	NaN	2023-06-03
11	5827	Rhythm Franklin	Email: rhythmic.franklin@email.edu	2023-03-28	NaN	2.99	NaN	NaT

In [20]:
```
# check the unique subscription rate
customers[['Subscription Plan', 'Subscription Rate']].drop_duplicates()
```

Out[20]:

	Subscription Plan	Subscription Rate
0	Basic (Ads)	2.99
1	NaN	2.99
6	Premium (No Ads)	9.99
15	Premium (No Ads)	99.99
21	Premium (No Ads)	7.99

In [21]:
```
# It look likes $2.99 is basic Subscription Plan, hence Nan values fill with basic Subscription Plan

customers['Subscription Plan'] = customers['Subscription Plan'].fillna('Basic,(Ads)')
customers.head()
```

Out[21]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	NaN	NaT
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	NaN	NaT
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	NaN	2023-06-01
3	5267	Rock Bassett	Email: groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	NaN	NaT
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	NaN	NaT

In [22]:
```
# find discount
customers['Discount?'].value_counts()
```

Out[22]:
```
Yes    7
Name: Discount?, dtype: int64
```

In [23]:
```
# change the Discount into numeric
customers['Discount?']=np.where(customers['Discount?']== 'Yes',1,0)
customers.head()
```

Out[23]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01
3	5267	Rock Bassett	Email: groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT

Inconsistent Text & Typos

In [24]:
```
# Look at customers - the Subscription Rate looks really high
customers.describe()
```

Out[24]:

	Customer ID	Subscription Rate	Discount?
count	30.000000	30.000000	30.000000
mean	6276.333333	8.556667	0.233333
std	814.255587	17.517840	0.430183
min	5001.000000	2.990000	0.000000
25%	5759.500000	2.990000	0.000000
50%	6196.000000	2.990000	0.000000
75%	6823.500000	7.990000	0.000000
max	7583.000000	99.990000	1.000000

In [25]:
```
# look into the $99.990000  ...looks like a typo
customers[customers['Subscription Rate'] > 7.99]
```

Out[25]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
6	5581	Reed Sharp	Email: sharp.tunes@email.com	2023-03-21	Premium (No Ads)	9.99	0	NaT
7	5759	Carol Kingbird	Email: songbird.carol@email.com	2023-03-22	Premium (No Ads)	9.99	0	2023-06-02
8	5761	Sonata Nash	Email: musical.sonata@email.com	2023-03-28	Premium (No Ads)	9.99	0	NaT
12	6029	Chord Campbell	Email: campbell.chordify@email.com	2023-03-29	Premium (No Ads)	9.99	0	2023-06-02
14	6163	Melody Parks	Email: park.of.melodies@email.com	2023-04-05	Premium (No Ads)	9.99	0	NaT
15	6229	Symphony Rhodes	Email: rhodes.symphony@email.com	2023-04-06	Premium (No Ads)	99.99	0	2023-06-02

In [26]:
```
# fix the 99.99 typo
customers.iloc[15,5] = 9.99
```

In [27]:
```
#check the range of customers
customers['Member Since'].max()
```

Out[27]:
```
Timestamp('2023-05-16 00:00:00')
```

In [28]:
```
# Look at listening_history
listening_history.head()
```

Out[28]:

	Customer ID	Session ID	Audio Order	Audio ID	Audio Type
0	5001	100520	1	101	Song
1	5001	100520	2	102	Song
2	5001	100520	3	103	Song
3	5001	100520	4	104	Song
4	5001	100520	5	105	Song

In [29]:
```
# count of audio type values
listening_history['Audio Type'].value_counts()
```

Out[29]:
```
Song       463
Podcast     42
Name: Audio Type, dtype: int64
```

In [30]:
```
# Look at audio
audio.head()
```

Out[30]:

	ID	Name	Genre	Popularity
0	Song-101	Dance All Night	Pop	1
1	Song-102	Unbreakable Beat	Pop	2
2	Song-103	Sunset Boulevard	Pop Music	5
3	Song-104	Glowing Hearts	Pop Music	10
4	Song-105	Pop Rocks	Pop Music	52

In [31]:
```
# count of Genre
audio.Genre.value_counts()
```

Out[31]:
```
Pop Music     3
Hip Hop       3
Comedy        3
Pop           2
Country       2
Jazz          2
True Crime    2
Name: Genre, dtype: int64
```

In [32]:
```
# Pop & Pop Music should be mapped into the same value
audio.Genre = np.where(audio.Genre == 'Pop Music', 'Pop', audio.Genre)
audio.Genre.value_counts()
```

Out[32]:
```
Pop           5
Hip Hop       3
Comedy        3
Country       2
Jazz          2
True Crime    2
Name: Genre, dtype: int64
```

In [33]:
```
# Look at log in time range
sessions['Session Log In Time'].max()
```

Out[33]:
```
Timestamp('2023-05-31 06:03:00')
```

Find Duplicate Rows

In [34]:
```
customers[customers.duplicated()]
```

Out[34]:

Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date

In [35]:
```
listening_history[listening_history.duplicated()]
```

Out[35]:

Customer ID	Session ID	Audio Order	Audio ID	Audio Type

In [36]:
```
audio[audio.duplicated()]
```

Out[36]:

ID	Name	Genre	Popularity

In [37]:
```
sessions[sessions.duplicated()]
```

Out[37]:

Session ID	Session Log In Time

Create New Columns

In [38]:
```
customers.head()
```

Out[38]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01
3	5267	Rock Bassett	Email: groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT

In [39]:
```
# Create a Cancelled column
customers['Cancelled'] = np.where(customers['Cancellation Date'].notna(), 1, 0)
customers.head()
```

Out[39]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
0	5001	Harmony Greene	Email: harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT	0
1	5002	Aria Keys	Email: melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT	0
2	5004	Lyric Bell	Email: rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01	1
3	5267	Rock Bassett	Email: groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT	0
4	5338	Rhythm Dixon	Email: beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT	0

In [40]:
```
# Create an Updated Email column without the 'email' word
customers['Email'] = customers.Email.str[6:]
customers.head()
```

Out[40]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
0	5001	Harmony Greene	harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT	0
1	5002	Aria Keys	melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT	0
2	5004	Lyric Bell	rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01	1
3	5267	Rock Bassett	groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT	0
4	5338	Rhythm Dixon	beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT	0

Exploratory Data Analysis

In [41]:
```
customers.head()
```

Out[41]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
0	5001	Harmony Greene	harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT	0
1	5002	Aria Keys	melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT	0
2	5004	Lyric Bell	rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01	1
3	5267	Rock Bassett	groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT	0
4	5338	Rhythm Dixon	beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT	0

In [42]:
```
# view the customers who cancelled
customers[customers['Cancellation Date'].notna()].head()
```

Out[42]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
2	5004	Lyric Bell	rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01	1
5	5404	Jazz Saxton	jazzy.sax@email.com	2023-03-20	Basic,(Ads)	2.99	0	2023-06-03	1
7	5759	Carol Kingbird	songbird.carol@email.com	2023-03-22	Premium (No Ads)	9.99	0	2023-06-02	1
12	6029	Chord Campbell	campbell.chordify@email.com	2023-03-29	Premium (No Ads)	9.99	0	2023-06-02	1
13	6092	Benny Beat	rhythmic.benny@email.com	2023-04-01	Basic (Ads)	2.99	0	2023-06-01	1

In [43]:
```
#customers tenurity before the cancellation
(customers['Cancellation Date'] - customers['Member Since']).mean()
```

Out[43]:
```
Timedelta('46 days 07:23:04.615384615')
```

In [44]:
```
# calculate the cancellation rate for those who had discount
discount_yes = customers[customers['Discount?']==1]
discount_yes
```

Out[44]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
21	6822	Kiki Keys	kiki.keys.piano@email.com	2023-05-01	Premium (No Ads)	7.99	1	NaT	0
22	6824	Greta Groove	groovy.greta@email.com	2023-05-01	Premium (No Ads)	7.99	1	2023-06-02	1
23	7087	Harmony Heart	heartfelt.harmony@email.com	2023-05-01	Premium (No Ads)	7.99	1	2023-06-02	1
25	7224	Melody Fitzgerald	fitzgerald.melody@email.com	2023-05-08	Premium (No Ads)	7.99	1	2023-06-01	1
26	7401	Reed Murphy	murphy.reed.music@email.com	2023-05-08	Premium (No Ads)	7.99	1	2023-06-01	1
28	7581	Lyric Keys	keysoflyric@email.com	2023-05-16	Premium (No Ads)	7.99	1	2023-06-03	1
29	7583	Melody Singer	melodic.singer@email.com	2023-05-16	Premium (No Ads)	7.99	1	2023-06-01	1

In [45]:
```
# calculate the cancellation rate in % for those who had discount
round(discount_yes.Cancelled.sum()/discount_yes.Cancelled.count()*100)
```

Out[45]:
```
86
```

In [46]:
```
# calculate the cancellation rate for those who did not have a discount
discount_no = customers[customers['Discount?']==0]
discount_no.head()
```

Out[46]:

	Customer ID	Customer Name	Email	Member Since	Subscription Plan	Subscription Rate	Discount?	Cancellation Date	Cancelled
0	5001	Harmony Greene	harmonious.vibes@email.com	2023-03-13	Basic (Ads)	2.99	0	NaT	0
1	5002	Aria Keys	melodious.aria@email.edu	2023-03-13	Basic,(Ads)	2.99	0	NaT	0
2	5004	Lyric Bell	rhythmical.lyric@email.com	2023-03-13	Basic,(Ads)	2.99	0	2023-06-01	1
3	5267	Rock Bassett	groovy.rock@email.com	2023-03-20	Basic (Ads)	2.99	0	NaT	0
4	5338	Rhythm Dixon	beats.by.rhythm@email.edu	2023-03-20	Basic,(Ads)	2.99	0	NaT	0

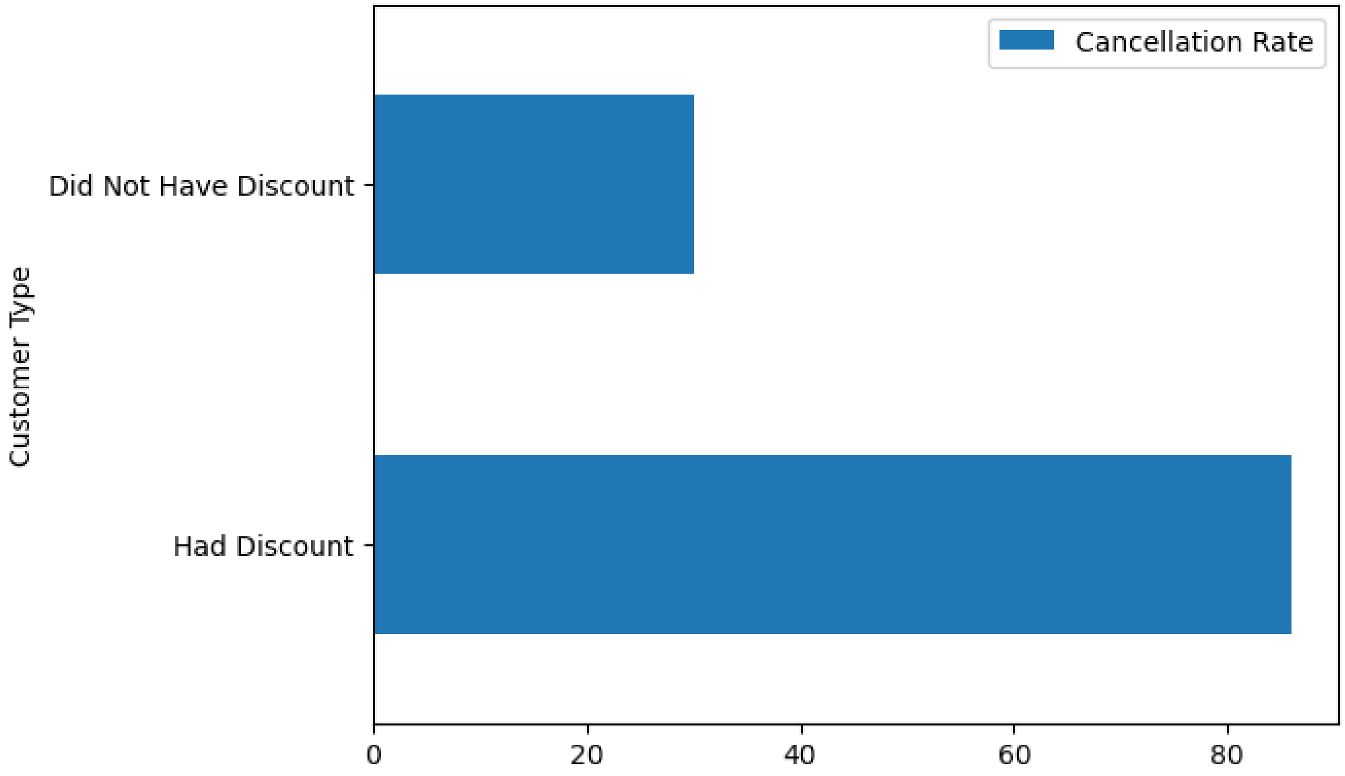
In [47]:
```
# calculate the cancellation rate for those who did not have a discount
round(discount_no.Cancelled.sum()/discount_no.Cancelled.count()*100)
```

Out[47]:
```
30
```

In [48]:
```
# Visulaize the cancellation rate for those who got a dixount vs who did not get any discount through bar plot

pd.DataFrame([['Had Discount', 86],
              ['Did Not Have Discount', 30]],
             columns = ['Customer Type', 'Cancellation Rate']).plot.barh(x ='Customer Type', y='Cancellation Rate',
```

Out[48]:
```
<Axes: ylabel='Customer Type'>
```



In [ ]:
```
# Better understand the customers 'listening history' - join together the listening_history & audio table & sessions
```

In [49]:
```
listening_history.head()
```

Out[49]:

	Customer ID	Session ID	Audio Order	Audio ID	Audio Type
0	5001	100520	1	101	Song
1	5001	100520	2	102	Song
2	5001	100520	3	103	Song
3	5001	100520	4	104	Song
4	5001	100520	5	105	Song

In [50]:
```
audio.head()
```

Out[50]:

	ID	Name	Genre	Popularity
0	Song-101	Dance All Night	Pop	1
1	Song-102	Unbreakable Beat	Pop	2
2	Song-103	Sunset Boulevard	Pop	5
3	Song-104	Glowing Hearts	Pop	10
4	Song-105	Pop Rocks	Pop	52

In [51]:
```
sessions.head()
```

Out[51]:

	Session ID	Session Log In Time
0	100520	2023-03-13 18:29:00
1	100522	2023-03-13 22:15:00
2	100525	2023-03-14 10:01:00
3	100527	2023-03-13 14:14:00
4	100538	2023-03-21 12:23:00

In [52]:
```
# split the 'ID' the audio so the column can be joined to other tables
audio_clean = pd.DataFrame(audio.ID.str.split('-').to_list()).rename(columns={0:'Type', 1:'Audio ID'})
audio_clean.head()
```

Out[52]:

	Type	Audio ID
0	Song	101
1	Song	102
2	Song	103
3	Song	104
4	Song	105

In [53]:
```
# Add the new fields into the original audio table
audio_all = pd.concat([audio_clean, audio], axis =1)
audio_all.head()
```

Out[53]:

	Type	Audio ID	ID	Name	Genre	Popularity
0	Song	101	Song-101	Dance All Night	Pop	1
1	Song	102	Song-102	Unbreakable Beat	Pop	2
2	Song	103	Song-103	Sunset Boulevard	Pop	5
3	Song	104	Song-104	Glowing Hearts	Pop	10
4	Song	105	Song-105	Pop Rocks	Pop	52

In [54]:
```
# check the data types of Audio ID in the audio table
audio_all.dtypes
```

Out[54]:
```
Type           object
Audio ID       object
ID             object
Name           object
Genre          object
Popularity      int64
dtype: object
```

In [55]:
```
# change the datatype of Audio ID form object to integer
audio_all['Audio ID'] = audio_all['Audio ID'].astype(int)
audio_all.dtypes
```

Out[55]:
```
Type           object
Audio ID        int32
ID             object
Name           object
Genre          object
Popularity      int64
dtype: object
```

In [56]:
```
# merge the Audio ID with listening_history

df = listening_history.merge(audio_all, how = 'left', on='Audio ID')
df
```

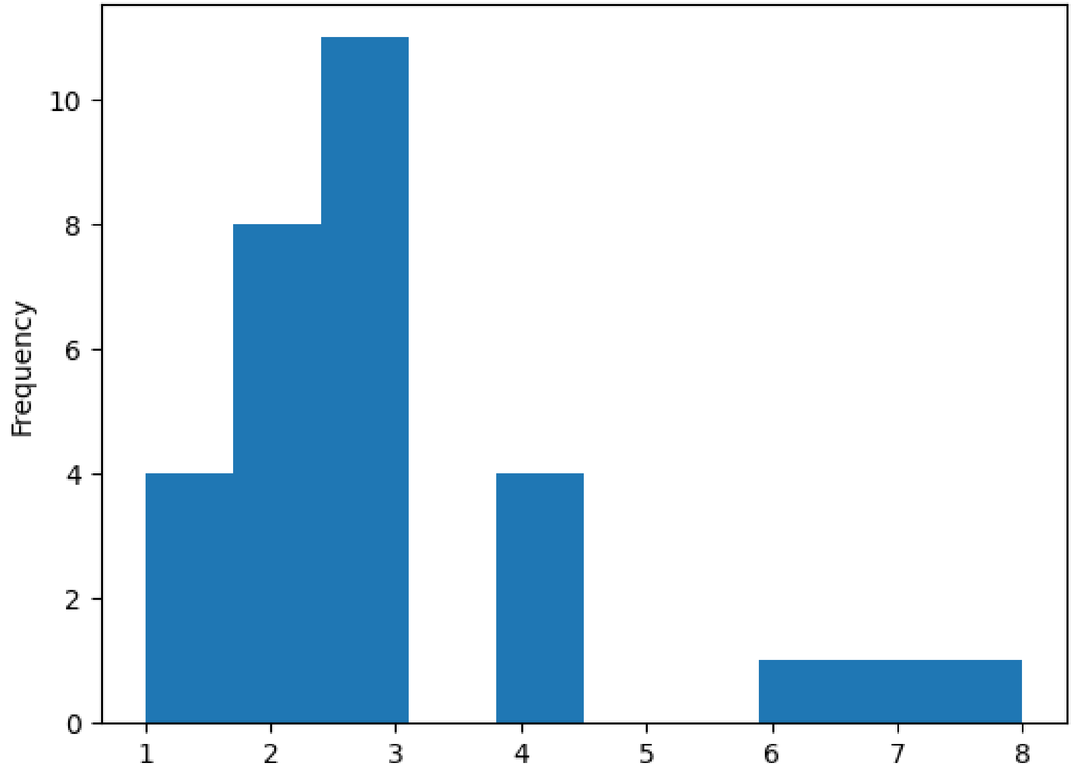
Out[56]:

	Customer ID	Session ID	Audio Order	Audio ID	Audio Type	Type	ID	Name	Genre	Popularity
0	5001	100520	1	101	Song	Song	Song-101	Dance All Night	Pop	1
1	5001	100520	2	102	Song	Song	Song-102	Unbreakable Beat	Pop	2
2	5001	100520	3	103	Song	Song	Song-103	Sunset Boulevard	Pop	5
3	5001	100520	4	104	Song	Song	Song-104	Glowing Hearts	Pop	10
4	5001	100520	5	105	Song	Song	Song-105	Pop Rocks	Pop	52
...	...	...	...	...	...	...	...	...	...	...
500	7579	111282	4	111	Song	Song	Song-111	Moonlit Serenade	Jazz	63
501	6588	111286	1	201	Podcast	Podcast	Podcast-201	Jokes on Jokes	Comedy	2
502	5763	111333	1	110	Song	Song	Song-110	Boss Moves	Hip Hop	28
503	5763	111333	2	108	Song	Song	Song-108	Chase the Dream	Hip Hop	4
504	5763	111333	3	110	Song	Song	Song-110	Boss Moves	Hip Hop	28

505 rows × 10 columns

In [57]:
```
# The number of listening sessions that each customer had in the past 3 months
df.groupby('Customer ID') ['Session ID'].nunique().plot.hist()
```

Out[57]:
```
<Axes: ylabel='Frequency'>
```



In [58]:
```
# The most popular Genre that customers listened to
df.Genre.value_counts()
```

Out[58]:
```
Pop           267
Hip Hop        88
Country        68
Jazz           48
Comedy         19
True Crime     15
Name: Genre, dtype: int64
```

Prep for Data Modeling¶

Create a DataFrame that is ready to modeling with each now represnting a customer and the following numeric & non-null columns:

Customer ID

Whether a cancelled or not

Whether a customer received discount or not

The number of listening sessions

% of listening history consisting in POP

% of listening history consisting in Podcast

In [59]:
```
# Create DataFrame ready for modeling
model_df = customers[['Customer ID' , 'Cancelled', 'Discount?']]
model_df.head()
```

Out[59]:

	Customer ID	Cancelled	Discount?
0	5001	0	0
1	5002	0	0
2	5004	1	0
3	5267	0	0
4	5338	0	0

In [60]:
```
# calculate the number of listening sessions for each customers
number_of_sessions = df.groupby('Customer ID')['Session ID'].nunique().rename('Number of Sessions').to_frame().rese
number_of_sessions.head()
```

Out[60]:

	Customer ID	Number of Sessions
0	5001	8
1	5002	4
2	5004	1
3	5267	7
4	5338	4

In [61]:
```
# add the 'number_of_session' into the modeling df
model_df = model_df.merge(number_of_sessions, how = 'left', on = 'Customer ID')
model_df.head()
```

Out[61]:

	Customer ID	Cancelled	Discount?	Number of Sessions
0	5001	0	0	8
1	5002	0	0	4
2	5004	1	0	1
3	5267	0	0	7
4	5338	0	0	4

In [62]:
```
# calculate dummy variables for each genre
pd.get_dummies(df.Genre)
```

Out[62]:

	Comedy	Country	Hip Hop	Jazz	Pop	True Crime
0	0	0	0	0	1	0
1	0	0	0	0	1	0
2	0	0	0	0	1	0
3	0	0	0	0	1	0
4	0	0	0	0	1	0
...	...	...	...	...	...	...
500	0	0	0	1	0	0
501	1	0	0	0	0	0
502	0	0	1	0	0	0
503	0	0	1	0	0	0
504	0	0	1	0	0	0

505 rows × 6 columns

In [63]:

```
# Combine it with Customer ID
pd.concat([df['Customer ID'], pd.get_dummies(df.Genre)], axis=1)
```

Out[63]:

	Customer ID	Comedy	Country	Hip Hop	Jazz	Pop	True Crime
0	5001	0	0	0	0	1	0
1	5001	0	0	0	0	1	0
2	5001	0	0	0	0	1	0
3	5001	0	0	0	0	1	0
4	5001	0	0	0	0	1	0
...	...	...	...	...	...	...	...
500	7579	0	0	0	1	0	0
501	6588	1	0	0	0	0	0
502	5763	0	0	1	0	0	0
503	5763	0	0	1	0	0	0
504	5763	0	0	1	0	0	0

505 rows × 7 columns

In [64]:

```
# group it by cutomer
genres = pd.concat([df['Customer ID'], pd.get_dummies(df.Genre)], axis=1).groupby('Customer ID').sum().reset_index(
genres.head()
```

Out[64]:

	Customer ID	Comedy	Country	Hip Hop	Jazz	Pop	True Crime
0	5001	0	0	26	0	34	0
1	5002	0	22	0	0	0	0
2	5004	0	0	0	0	9	0
3	5267	0	0	22	0	23	0
4	5338	0	18	0	0	0	0

In [65]:
```
# add a column for total songs/podcasts listened to
total_audio = listening_history.groupby('Customer ID')['Audio ID'].count().rename('Total Audio').to_frame().reset_i
total_audio.head()
```

Out[65]:

	Customer ID	Total Audio
0	5001	60
1	5002	22
2	5004	9
3	5267	45
4	5338	18

In [66]:
```
# Create a master audio table to calculate the %
df_audio = genres.merge(total_audio, how ='left', on = 'Customer ID')
df_audio
```

Out[66]:

	Customer ID	Comedy	Country	Hip Hop	Jazz	Pop	True Crime	Total Audio
0	5001	0	0	26	0	34	0	60
1	5002	0	22	0	0	0	0	22
2	5004	0	0	0	0	9	0	9
3	5267	0	0	22	0	23	0	45
4	5338	0	18	0	0	0	0	18
5	5404	0	0	0	0	8	0	8
6	5581	0	0	0	0	0	5	5
7	5759	0	0	0	0	15	0	15
8	5761	0	0	0	0	0	5	5
9	5763	0	0	11	0	20	0	31
10	5826	0	17	0	0	0	0	17
11	5827	0	0	0	0	7	0	7
12	6029	0	0	0	0	12	0	12
13	6092	4	0	3	0	3	0	10
14	6163	0	0	0	0	0	4	4
15	6229	0	0	0	0	13	0	13
16	6406	4	0	2	0	3	0	9
17	6584	0	4	6	4	13	0	27
18	6586	0	4	4	4	10	0	22
19	6588	3	0	3	0	4	0	10
20	6821	0	3	5	3	10	0	21
21	6822	0	0	0	15	0	0	15
22	6824	0	0	0	0	31	0	31
23	7087	3	0	3	0	5	0	11
24	7158	0	0	0	13	0	0	13
25	7224	0	0	0	0	29	0	29
26	7401	3	0	3	0	5	0	11

	Customer ID	Comedy	Country	Hip Hop	Jazz	Pop	True Crime	Total Audio
27	7579	0	0	0	9	0	0	9
28	7581	0	0	0	0	13	1	14
29	7583	2	0	0	0	0	0	2

In [67]:
```
# % of POP
model_df['Percentage Pop'] = df_audio.Pop/df_audio['Total Audio'] *100
model_df.head()
```

Out[67]:

	Customer ID	Cancelled	Discount?	Number of Sessions	Percentage Pop
0	5001	0	0	8	56.666667
1	5002	0	0	4	0.000000
2	5004	1	0	1	100.000000
3	5267	0	0	7	51.111111
4	5338	0	0	4	0.000000

In [68]:
```
# % of Podcast
model_df['Percentage Podcast'] = ((df_audio['Comedy'] + df_audio['True Crime']) / df_audio['Total Audio']) *100
model_df.head()
```

Out[68]:

	Customer ID	Cancelled	Discount?	Number of Sessions	Percentage Pop	Percentage Podcast
0	5001	0	0	8	56.666667	0.0
1	5002	0	0	4	0.000000	0.0
2	5004	1	0	1	100.000000	0.0
3	5267	0	0	7	51.111111	0.0
4	5338	0	0	4	0.000000	0.0

In [69]:
```
model_df
```

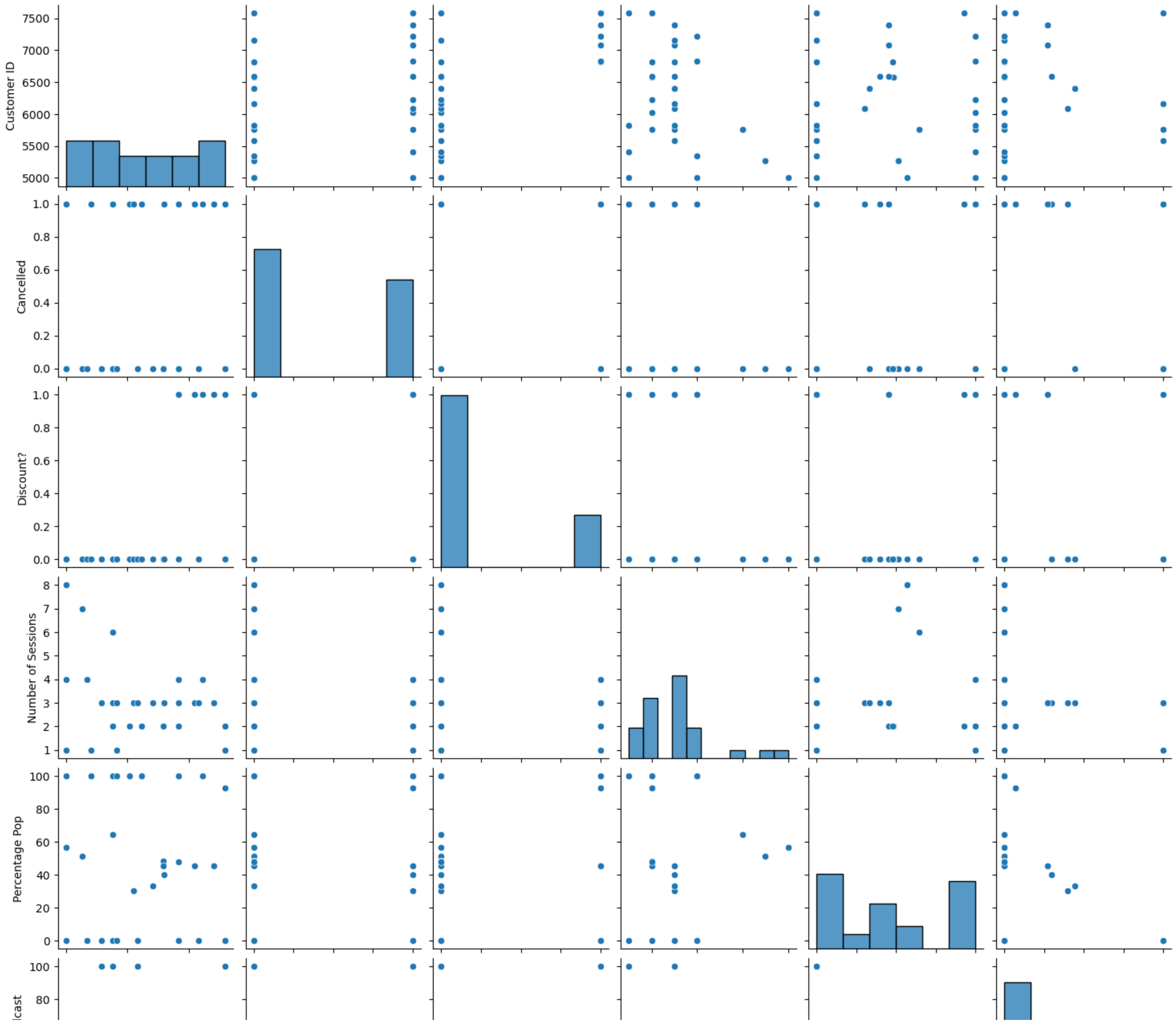
Out[69]:

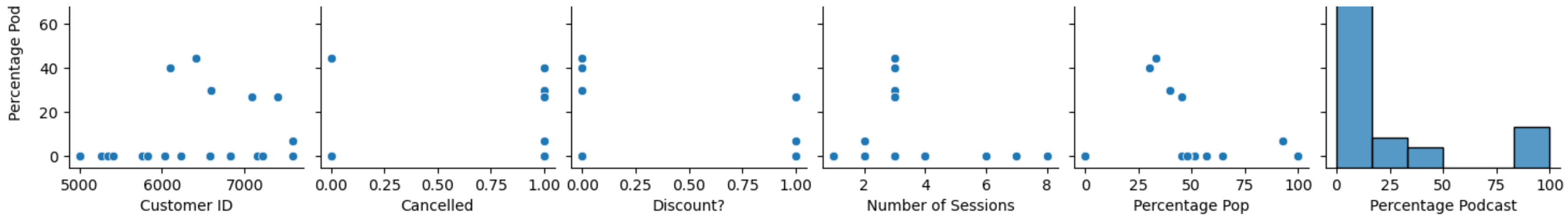
	Customer ID	Cancelled	Discount?	Number of Sessions	Percentage Pop	Percentage Podcast
0	5001	0	0	8	56.666667	0.000000
1	5002	0	0	4	0.000000	0.000000
2	5004	1	0	1	100.000000	0.000000
3	5267	0	0	7	51.111111	0.000000
4	5338	0	0	4	0.000000	0.000000
5	5404	1	0	1	100.000000	0.000000
6	5581	0	0	3	0.000000	100.000000
7	5759	1	0	2	100.000000	0.000000
8	5761	0	0	3	0.000000	100.000000
9	5763	0	0	6	64.516129	0.000000
10	5826	0	0	3	0.000000	0.000000
11	5827	0	0	1	100.000000	0.000000
12	6029	1	0	2	100.000000	0.000000
13	6092	1	0	3	30.000000	40.000000
14	6163	0	0	3	0.000000	100.000000
15	6229	1	0	2	100.000000	0.000000
16	6406	0	0	3	33.333333	44.444444
17	6584	0	0	2	48.148148	0.000000
18	6586	0	0	2	45.454545	0.000000
19	6588	1	0	3	40.000000	30.000000
20	6821	0	0	2	47.619048	0.000000
21	6822	0	1	3	0.000000	0.000000
22	6824	1	1	4	100.000000	0.000000
23	7087	1	1	3	45.454545	27.272727
24	7158	0	0	3	0.000000	0.000000
25	7224	1	1	4	100.000000	0.000000
26	7401	1	1	3	45.454545	27.272727

	Customer ID	Cancelled	Discount?	Number of Sessions	Percentage Pop	Percentage Podcast
27	7579	0	0	2	0.000000	0.000000
28	7581	1	1	2	92.857143	7.142857
29	7583	1	1	1	0.000000	100.000000

In [71]:
```
# Visualize the relationships in the modeling DataFrame using a pairplot

import seaborn as sns
sns.pairplot(model_df);
```


Customer ID
Cancelled
Discount?
Number of Sessions
Percentage Pop
lcast
7500
7000
6500
6000
5500
5000
1.0
0.8
0.6
0.4
0.2
0.0
8
7
6
5
4
3
2
1
100
80
60
40
20
0



In [72]:
```
# Look at the correlations
model_df.corr()
```

Out[72]:

	Customer ID	Cancelled	Discount?	Number of Sessions	Percentage Pop	Percentage Podcast
Customer ID	1.000000	0.269942	0.648514	-0.337083	-0.076129	0.083083
Cancelled	0.269942	1.000000	0.471825	-0.333739	0.585630	-0.035414
Discount?	0.648514	0.471825	1.000000	-0.048877	0.112675	0.062938
Number of Sessions	-0.337083	-0.333739	-0.048877	1.000000	-0.131156	-0.125459
Percentage Pop	-0.076129	0.585630	0.112675	-0.131156	1.000000	-0.487193
Percentage Podcast	0.083083	-0.035414	0.062938	-0.125459	-0.487193	1.000000

In [73]:
```
# Final Observations
## A dicount is correlated with a cancellation
## The more listening session, the fewer cancellation
## Max Cancelling is comimg from Pop music
## Podcast seems unrelated to cancellation
```