In [ ]:

```
#SAMPLE PYTHON CODE
'''
- The following sample contains several python commands that show my ability to perform data transforma
tion, data aggregation, and data visualization using python functional programming.

THE DATA
---------
- The exercise is composed of two csv files that contain dummy marketing data:

Customer Segments Table: contains data about the customers accounts.
Billing Status Table : contains data about the billing.


EXTERNAL VARIABLES
------------------
Price Change in account subscriptions -> 2018-12
New Release with better performance and features -> 2018-12


Analysis:
---------
Effect of these variables on customers and ROI.

'''
```

In [1]:

```
#OPEN FILES AND IMPORT LIBRARIES
import pandas as pd
import re
import matplotlib.pyplot as plt

df0 = pd.read_csv("customer_segments.csv")
df1 = pd.read_csv("billing_status.csv")
```

In [2]:

```
#CUSTOMER SEGMENENTS TABLE ( TOP FIVE LINES )
df0.head(5)
```

Out[2]:

	Customer ID	Free Account Created Date	Last Seen	Session Count	Segment	Billing Account #
0	1	2017-05-29	2019-04-05	1379	Marketing Segment 2	AX#93195
1	2	2017-12-17	2019-04-27	531	Marketing Segment 1	AX#20438
2	3	2017-01-30	2019-04-22	317	Marketing Segment 1	AX#60779
3	4	2017-02-03	2019-04-23	349	Marketing Segment 1	AX#91209
4	5	2017-03-18	2019-04-05	1158	Marketing Segment 2	AX#57000

In [7]:

```
#BILLING STATUS TABLE ( TOP FIVE LINES )
df1.head(5)
```

Out[7]:

	Billing Account #	Account Status	Upgraded Account Created Date	Upgraded Account Cancelled Date	Total Charges
0	AX#10042	paying	2019-03-20	NaN	$332.80
1	AX#10114	paying	2019-01-17	NaN	$451.13
2	AX#10259	paying	2018-09-28	NaN	$1,886.80
3	AX#10695	paying	2018-12-21	NaN	$797.87
4	AX#10855	paying	2019-04-01	NaN	$189.00

In [8]:

```
#MAKE THE JOIN BETWEEN TABLES
dfjoin = pd.merge(df0,df1,how = 'left', left_on = 'Billing Account #', right_on='Billing Account #')
```

In [9]:

```
#JOIN TABLE ( TOP FIVE LINES )
dfjoin
```

Out[9]:

	Customer ID	Free Account Created Date	Last Seen	Session Count	Segment	Billing Account #	Account Status	Upgraded Account Created Date	Upgraded Account Cancelled Date	Total Charges
0	1	2017-05-29	2019-04-05	1379	Marketing Segment 2	AX#93195	paying	2019-04-02	NaN	$145.60
1	2	2017-12-17	2019-04-27	531	Marketing Segment 1	AX#20438	paying	2018-11-11	NaN	$728.00
2	3	2017-01-30	2019-04-22	317	Marketing Segment 1	AX#60779	paying	2019-03-15	NaN	$277.20
3	4	2017-02-03	2019-04-23	349	Marketing Segment 1	AX#91209	paying	2019-03-24	NaN	$292.83
4	5	2017-03-18	2019-04-05	1158	Marketing Segment 2	AX#57000	paying	2018-10-12	NaN	$462.00
...	...	...	...	...	...	...	...	...	...	...
1009	996	2019-04-25	2019-04-28	1	Marketing Segment 2	AX#88630	paying	2019-04-25	NaN	$21.20
1010	997	2019-04-26	2019-04-26	0	Marketing Segment 2	AX#53242	paying	2019-04-27	NaN	$6.83
1011	998	2019-04-27	2019-04-27	1	Marketing Segment 3	AX#50772	paying	2019-04-27	NaN	$4.73
1012	999	2019-04-28	2019-04-28	0	Marketing Segment 1	AX#75078	paying	2019-04-28	NaN	$-
1013	1000	2019-04-28	2019-04-28	0	Marketing Segment 2	AX#89342	paying	2019-04-28	NaN	$-

1014 rows × 10 columns

In [10]:

```
#JOIN TABLE - METADATA ( SCHEEMA AND DATATYPES)
dfjoin.dtypes
```

Out[10]:

```
Customer ID                         int64
Free Account Created Date          object
Last Seen                          object
Session Count                       int64
Segment                            object
Billing Account #                  object
Account Status                     object
Upgraded Account Created Date      object
Upgraded Account Cancelled Date    object
Total Charges                      object
dtype: object
```

In [11]:

```
'''
CLEANING DATA .....
'''
#CONVERT CURRENCIES INTO FLOATS (remove [$,-\s])
dfjoin['Total Charges'] = dfjoin['Total Charges'].map(lambda x: x.lstrip('$ ').rstrip('$ ')) #strip
 ['\s','$']
dfjoin['Total Charges'] = dfjoin['Total Charges'].apply(lambda x: x.replace(',','').replace('-','0')).a
stype('float')
dfjoin['Total Charges']
```

Out[11]:

```
0       145.60
1       728.00
2       277.20
3       292.83
4       462.00
         ...
1009     21.20
1010      6.83
1011      4.73
1012      0.00
1013      0.00
Name: Total Charges, Length: 1014, dtype: float64
```

In [12]:

```
'''
ADDING SLICES FOR THE ANALYSIS
'''
#ADD year_Upgrated slice
#-------------------------
dfjoin['year_Upgraded'] = dfjoin['Upgraded Account Created Date'].str[:7]
dfjoin.dtypes
```

Out[12]:

```
Customer ID                          int64
Free Account Created Date           object
Last Seen                           object
Session Count                        int64
Segment                             object
Billing Account #                   object
Account Status                      object
Upgraded Account Created Date       object
Upgraded Account Cancelled Date     object
Total Charges                      float64
year_Upgraded                       object
dtype: object
```

In [13]:

```
'''
ANALYSIS
'''
#TOTAL SUM OF CHARGES ALL MARKETS
#-------------------------------
dfjoin.groupby(['year_Upgraded'])['Total Charges'].sum()
```

Out[13]:

```
year_Upgraded
2017-12    16861.13
2018-01    27355.33
2018-02    10866.98
2018-03    22589.81
2018-04    75067.60
2018-05    15558.43
2018-06    32922.12
2018-07    32622.26
2018-08    24998.21
2018-09    36122.05
2018-10    32823.37
2018-11    62145.74
2018-12    55113.17
2019-01    37862.08
2019-02    34583.31
2019-03    32525.63
2019-04    18681.54
Name: Total Charges, dtype: float64
```

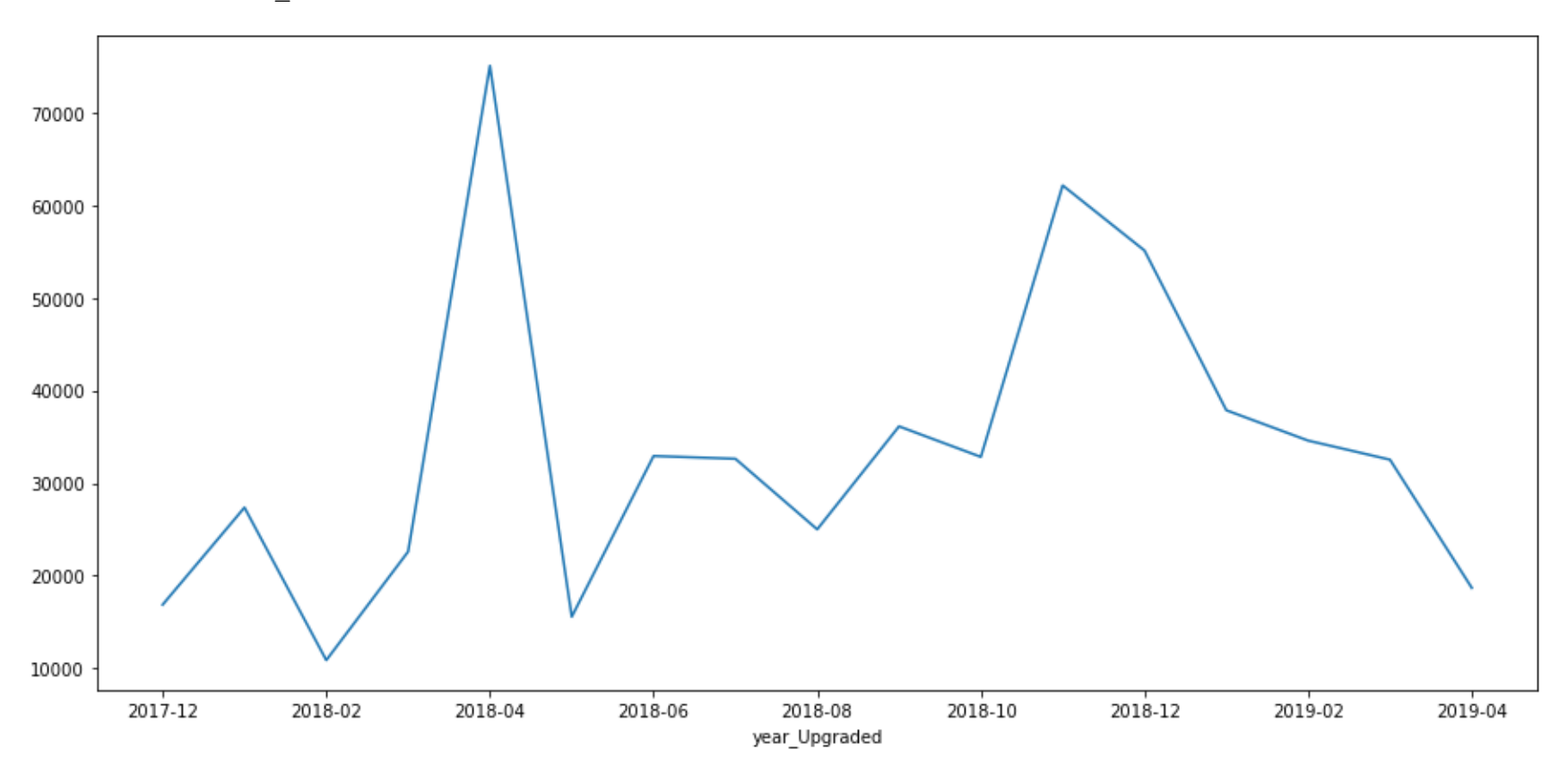
In [14]:

```
'''
ANALYSIS
'''
#TREND TOTAL SUM OF CHARGES ALL MARKETS
#-------------------------------------
fig, ax = plt.subplots(figsize=(15,7))
dfjoin.groupby(['year_Upgraded'])['Total Charges'].sum().plot(ax=ax)


'''
========
FINDINGS
========

- The collective state of the income generated has a declining trend.
- The latest month has a total income similar to the launch of the product in2017-12.
- The income peaks since the opening of the firm were in Q1 and Q4 of 2018
       2017-12    16861.13
       2018-04    75067.60
       2018-11    62145.74
       2019-04    18681.54
'''
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x18d0fd22130>



In [25]:

```
'''
ANALYSIS
'''
#TOTAL SUM OF CHARGES ALL MARKETS BY SEGMENT
#-----------------------------------------
dfjoin.groupby(['Segment','year_Upgraded'])['Total Charges'].sum()
```

Out[25]:

```
Segment              year_Upgraded
Marketing Segment 1  2017-12           9243.23
                     2018-01          13893.86
                     2018-02          10616.11
                     2018-03          10665.33
                     2018-04          34014.66
                     2018-05           5907.97
                     2018-06          12804.16
                     2018-07           3344.49
                     2018-08           3558.54
                     2018-09           8137.21
                     2018-10          10667.24
                     2018-11          13401.80
                     2018-12          11551.77
                     2019-01           8660.69
                     2019-02           3381.60
                     2019-03           7227.62
                     2019-04           5781.08
Marketing Segment 2  2017-12           5938.30
                     2018-01           8782.90
                     2018-03           6829.48
                     2018-04          27312.03
                     2018-05           4331.77
                     2018-06           8633.83
                     2018-07          20155.17
                     2018-08          17498.27
                     2018-09          15470.97
                     2018-10          14590.97
                     2018-11          24591.71
                     2018-12          22993.06
                     2019-01          15692.61
                     2019-02          17377.08
                     2019-03          14977.00
                     2019-04           5166.25
Marketing Segment 3  2017-12           1679.60
                     2018-01           4678.57
                     2018-02            250.87
                     2018-03           5095.00
                     2018-04          13740.91
                     2018-05           5318.69
                     2018-06          11484.13
                     2018-07           9122.60
                     2018-08           3941.40
                     2018-09          12513.87
                     2018-10           7565.16
                     2018-11          24152.23
                     2018-12          20568.34
                     2019-01          13508.78
                     2019-02          13824.63
                     2019-03          10321.01
                     2019-04           7734.21
Name: Total Charges, dtype: float64
```

In [26]:

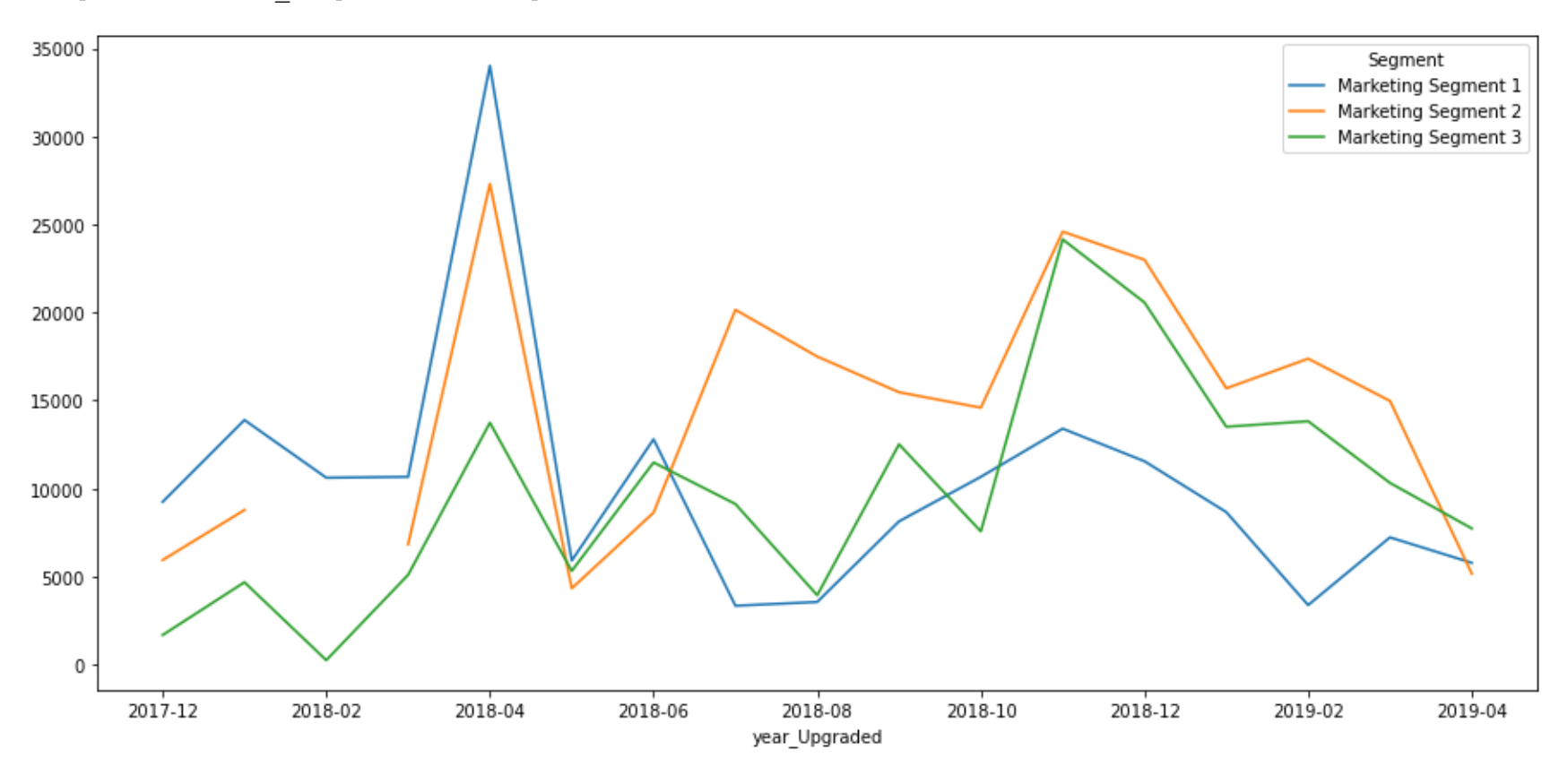
```
#TREND TOTAL SUM OF CHARGES ALL MARKETS BY SEGMENT
#-----------------------------------------------
fig, ax = plt.subplots(figsize=(15,7))
dfjoin.groupby(['year_Upgraded','Segment'])['Total Charges'].sum().unstack().plot(ax=ax)

'''
========
FINDINGS
========

- All markets are experiencing income decline. This can be  due a reduction in prices or a reduction in
 paying accounts.

'''
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x22385a7ddf0>



In [40]:

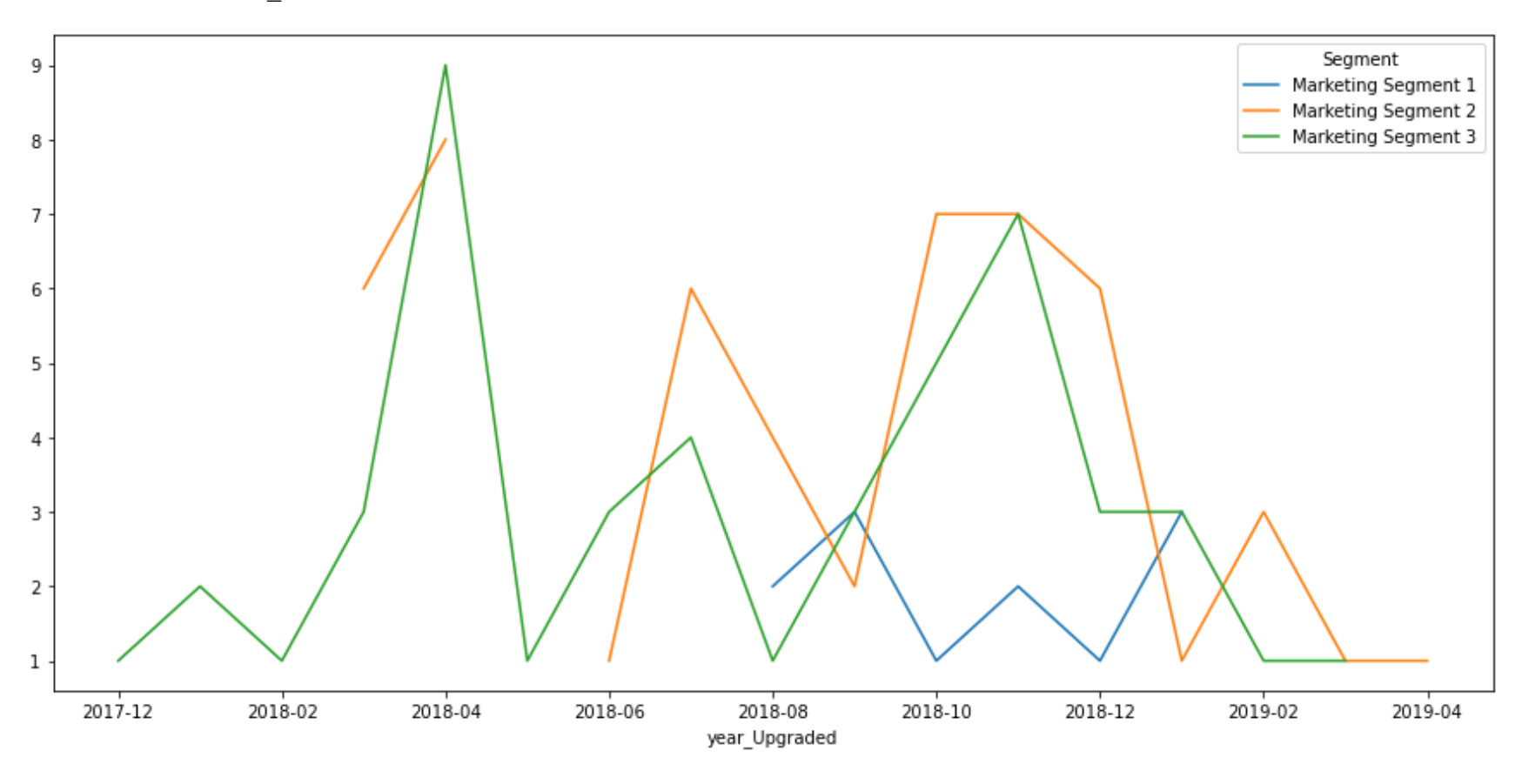
```
#TOTAL COUNT OF ACCOUNTS BY ACCOUNT STATUS CANCELLED
#-----------------------------------------
index_cancelled = dfjoin.pipe(lambda x: x['Account Status'] != "paying")
index_paying = dfjoin.pipe(lambda x: x['Account Status'] == "paying")
fig, ax = plt.subplots(figsize=(15,7))
dfjoin[index_cancelled].groupby(['year_Upgraded','Segment'])['Customer ID'].count().unstack().plot(ax=a
x)

'''
========
FINDINGS
========

- After the price change and the new release, the status of the accounts canceled started to decline.

'''
```

Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x223867e6370>



In [41]:

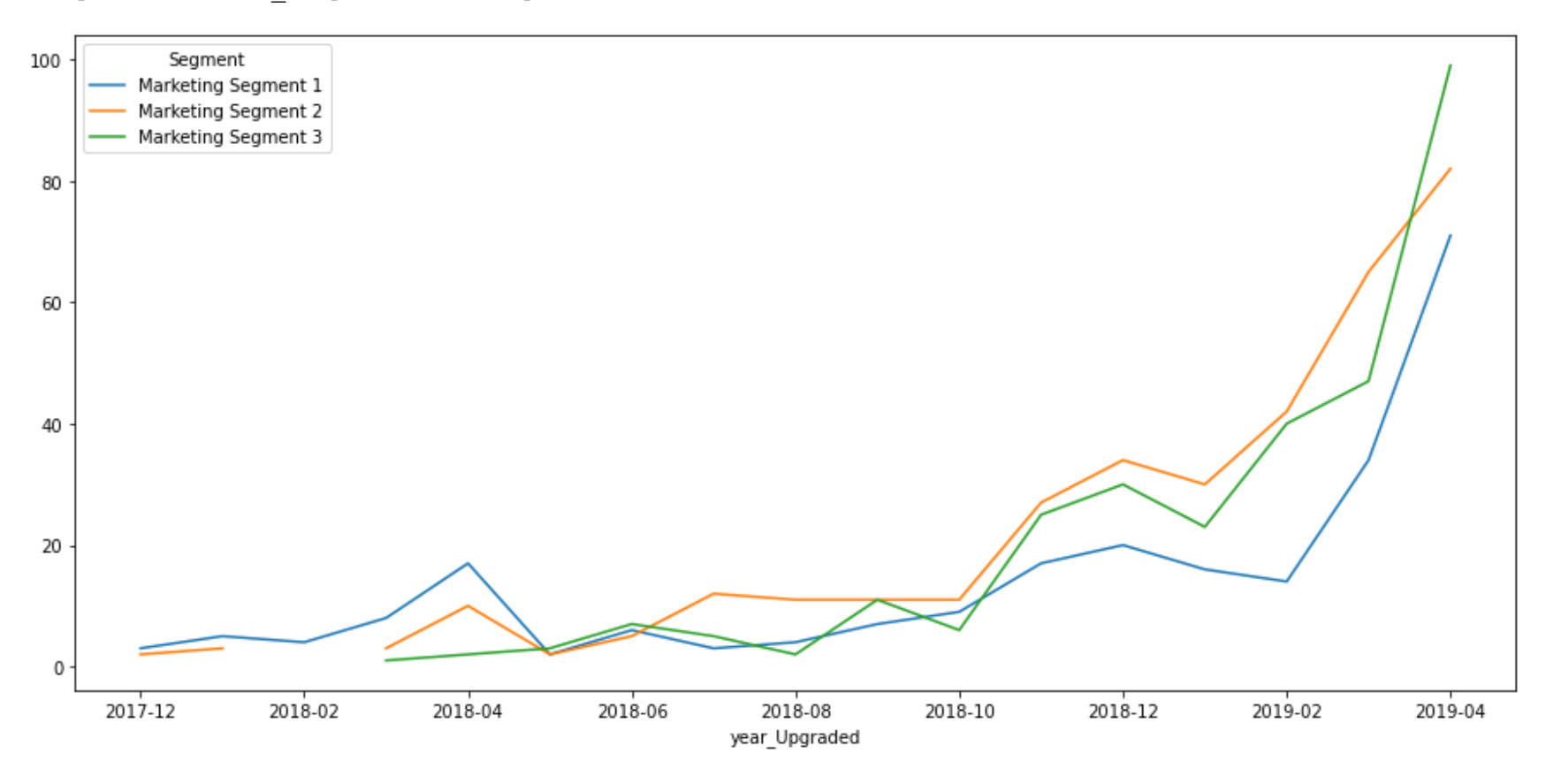
```
#TOTAL COUNT OF ACCOUNTS BY ACCOUNT STATUS PAYING
#-----------------------------------------------
fig, ax = plt.subplots(figsize=(15,7))
dfjoin[index_paying].groupby(['year_Upgraded','Segment'])['Customer ID'].count().unstack().plot(ax=ax)

'''
========
FINDINGS
========

- After the price change and the new release, the status of the accounts paying started to increase rai
dly.

'''
```

Out[41]: <matplotlib.axes._subplots.AxesSubplot at 0x22386840340>



In [42]:

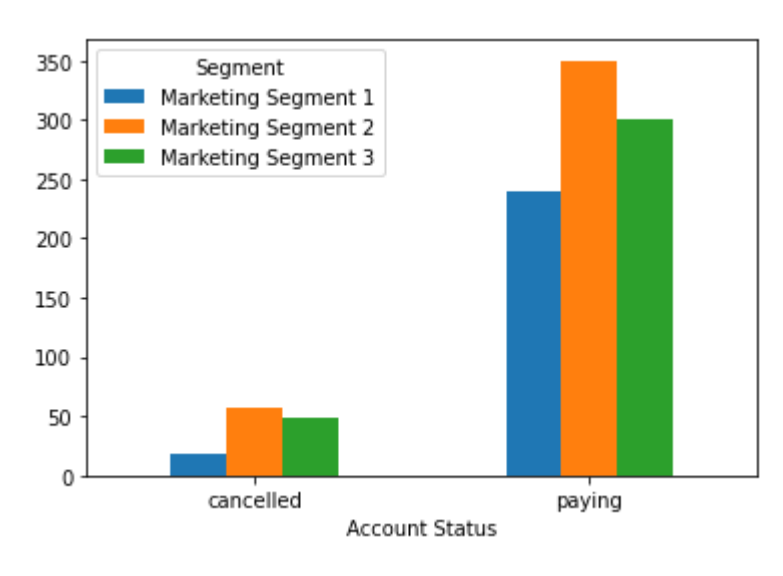
```
#WHICH SEGMENT PAYS THE MOST
#---------------------------
dfjoin.groupby(['Account Status','Segment'])['Customer ID'].count().unstack().plot.bar(rot=0)


'''
========
FINDINGS
========

- Overall in all periods, Market Segment 2 have paid and cancelled more.

'''
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x223868b9ee0>



In [43]:

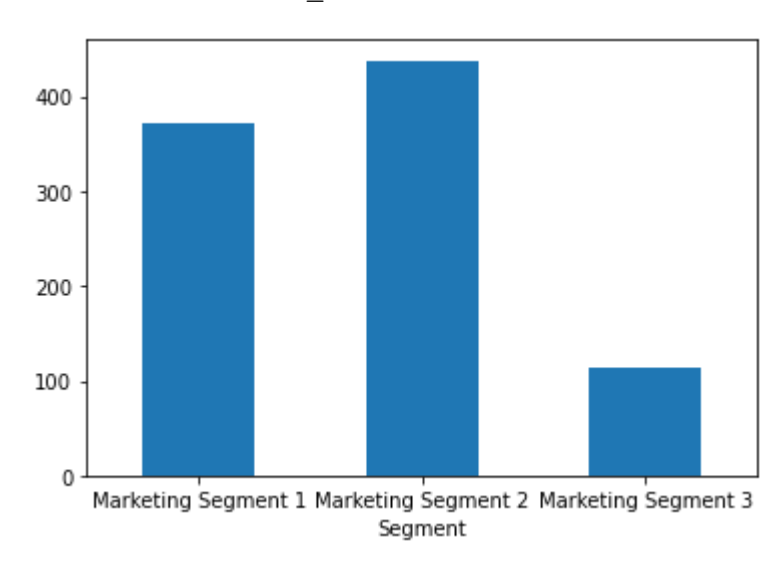
```
#WHICH SEGMENT USES OUR APPLICATION MORE (OVERALL ALL PERIODS)
#-----------------------------------------------------------
dfjoin.groupby(['Segment'])['Session Count'].mean().plot.bar(rot=0)

'''
========
FINDINGS
========

- Overall in all periods, Market Segment 2 have used the application more.

'''
```

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x22386930100>



In [46]:

```
#WHICH SEGMENT USES OUR APPLICATION MORE BEFORE THE RELEASE
#-----------------------------------------------------------
regex_NOT_2019 = "2017.*|2018.*"
series = dfjoin.iloc[:,10]
index_NOT_2019 = series.str.contains(regex_NOT_2019)

dfjoin[index_NOT_2019].groupby(['Segment'])['Session Count'].mean()


'''
========
FINDINGS
========

- Before the release, Market Segment 2 have used the application more.

'''
```

Out[46]:

```
Segment
Marketing Segment 1    674.193277
Marketing Segment 2    855.076923
Marketing Segment 3     93.296296
Name: Session Count, dtype: float64
```

In [45]:

```
#WHICH SEGMENT USES OUR APPLICATION MORE AFTER THE LAST RELEASE
#-----------------------------------------------------------
regex_2019 = "2019.*"
series = dfjoin.iloc[:,10]
index_2019 = series.str.contains(regex_2019)

dfjoin[index_2019].groupby(['Segment'])['Session Count'].mean()

'''
========
FINDINGS
========

- After the release, Market Segment 3 have used the application more.

'''
```

Out[45]:

```
Segment
Marketing Segment 1    111.870504
Marketing Segment 2    100.382222
Marketing Segment 3    127.733645
Name: Session Count, dtype: float64
```

In [ ]:

```
#FINDINGS SUMMARY
'''
OVERALL:
- The collective state of the income generated has a declining trend.
- The latest month has a total income similar to the launch of the product in2017-12.
- The income peaks since the opening of the firm were in Q1 and Q4 of 2018
       2017-12    16861.13
       2018-04    75067.60
       2018-11    62145.74
       2019-04    18681.54

STATUS:
- The overall status of the accounts seems to have an increased trend of payingaccounts and a decreased
 trend on cancellations.
- Therefore, this means that the income problem must come from the productpricing change introduced on
 2018-11-13
- There are more paying customers, however the company is making less money.

MARKET SEGMENTS:
- The overall Market Segment II has the highest count of paid accounts;however, the trend shows Market
 Segment III as current leader in New AccountsPaying.
- As per use, the Marketing Segment II was leading before the release, but in2019 Market Segment II has
 the lowest use.  After the release, MarketSegment III has the highest use, followed by Market Segment
 I.

RECOMMENDATIONS:
- I would recommend to increase the pricing on the subscriptions to correct thedrop in income, however,
 if the goal is to attract new paid customers, thereduction in pricing seems to be working.

Facebook Ads Campaign:
- In order to check if the campaign on Facebook is a success, I would payattention to the Free Account
 Created Date vector.
- If in the following months there is an increase of new free accounts created,that means that customer
s are seeing the adds and trying the product.

Please find attached my analysis in the pdf .

Thank you very much for giving me the opportunity to prove my skills, thisexercise has been a lot of fu
n.
'''
```