In [100…

```python
import pandas as pd
import datetime as dt
import numpy as np
import matplotlib.pyplot as plt
```

In [40]:

```python
# Loading the dataset
flo = pd.read_csv("flo_data_20k.csv")
flo.head()
```

Out[40]:

	master_id	order_channel	last_order_channel	first_order_date	last_order_date	last_order_date_online	last_order_date_offline	order_num_
0	cc294636-19f0-11eb-8d74-000d3a38a36f	Android App	Offline	2020-10-30	2021-02-26	2021-02-21	2021-02-26	
1	f431bd5a-ab7b-11e9-a2fc-000d3a38a36f	Android App	Mobile	2017-02-08	2021-02-16	2021-02-16	2020-01-10	
2	69b69676-1a40-11ea-941b-000d3a38a36f	Android App	Android App	2019-11-27	2020-11-27	2020-11-27	2019-12-01	
3	1854e56c-491f-11eb-806e-000d3a38a36f	Android App	Android App	2021-01-06	2021-01-17	2021-01-17	2021-01-06	
4	d6ea1074-f1f5-11e9-9346-000d3a38a36f	Desktop	Desktop	2019-08-03	2021-03-07	2021-03-07	2019-08-03	

In [41]:

```python
# Checking the information of the dataset
flo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19945 entries, 0 to 19944
Data columns (total 12 columns):
 #   Column                             Non-Null Count  Dtype  
---  ------                             --------------  -----  
 0   master_id                          19945 non-null  object 
 1   order_channel                      19945 non-null  object 
 2   last_order_channel                 19945 non-null  object 
 3   first_order_date                   19945 non-null  object 
 4   last_order_date                    19945 non-null  object 
 5   last_order_date_online             19945 non-null  object 
 6   last_order_date_offline            19945 non-null  object 
 7   order_num_total_ever_online        19945 non-null  float64
 8   order_num_total_ever_offline       19945 non-null  float64
 9   customer_value_total_ever_offline  19945 non-null  float64
 10  customer_value_total_ever_online   19945 non-null  float64
 11  interested_in_categories_12        19945 non-null  object 
dtypes: float64(4), object(8)
memory usage: 1.8+ MB
```

There is no null in this dataset

In [43]:
```python
# Explore the data: average, min, max, statistics number
flo.describe()
```

Out[43]:

	order_num_total_ever_online	order_num_total_ever_offline	customer_value_total_ever_offline	customer_value_total_ever_online
count	19945.000000	19945.000000	19945.000000	19945.000000
mean	3.110855	1.913913	253.922597	497.321690
std	4.225647	2.062880	301.532853	832.601886
min	1.000000	1.000000	10.000000	12.990000
25%	1.000000	1.000000	99.990000	149.980000
50%	2.000000	1.000000	179.980000	286.460000
75%	4.000000	2.000000	319.970000	578.440000
max	200.000000	109.000000	18119.140000	45220.130000

In [44]:
```python
# Examining the variable types and changing the type of variables that express date to date
flo['first_order_date'] = pd.to_datetime(flo['first_order_date'])
flo['last_order_date'] = pd.to_datetime(flo['last_order_date'])
```

In [45]:
```
#reference date
ref_date = flo["last_order_date"].max() + pd.Timedelta(days=1)
```

In [46]:
```
#Calculating the Recency: Total number of days since last order Using the most recent order date (either online or offline)
flo["Recency"] = (ref_date - flo["last_order_date"]).dt.days

# Calculating the Frequency (sum of total orders)
flo['Frequency'] = flo['order_num_total_ever_online'] + flo['order_num_total_ever_offline']

# Calculating the Monetary (Total spenderutes of each customer)
flo['Monetary'] = flo['customer_value_total_ever_online'] + flo['customer_value_total_ever_offline']
```

In [47]:
```
flo[["master_id", "Recency", "Frequency", "Monetary"]].head()
```

Out[47]:

	master_id	Recency	Frequency	Monetary
0	cc294636-19f0-11eb-8d74-000d3a38a36f	94	5.0	939.37
1	f431bd5a-ab7b-11e9-a2fc-000d3a38a36f	104	21.0	2013.55
2	69b69676-1a40-11ea-941b-000d3a38a36f	185	5.0	585.32
3	1854e56c-491f-11eb-806e-000d3a38a36f	134	2.0	121.97
4	d6ea1074-f1f5-11e9-9346-000d3a38a36f	85	2.0	209.98

In [48]:
```
rfm = flo[["master_id", "Recency", "Frequency", "Monetary"]].copy()
rfm.set_index("master_id", inplace=True)
```

In [49]:
```
rfm["R_score"] = pd.qcut(rfm["Recency"], 5, labels=[5, 4, 3, 2, 1]).astype(int) #Lowset recency get highest R score
rfm["F_score"] = pd.qcut(rfm["Frequency"].rank(method="first"), 5, labels=[1, 2, 3, 4, 5]).astype(int)
rfm["M_score"] = pd.qcut(rfm["Monetary"], 5, labels=[1, 2, 3, 4, 5]).astype(int)

rfm['RFM_Score']= rfm[["R_score", "F_score", "M_score"]].astype(int).sum(axis=1)
rfm["RFM_Segment"] = rfm["R_score"].astype(str) + rfm["F_score"].astype(str) + rfm["M_score"].astype(str)

rfm.head()
```

Out[49]:

master_id	Recency	Frequency	Monetary	R_score	F_score	M_score	RFM_Score	RFM_Segment
cc294636-19f0-11eb-8d74-000d3a38a36f	94	5.0	939.37	3	4	4	11	344
f431bd5a-ab7b-11e9-a2fc-000d3a38a36f	104	21.0	2013.55	3	5	5	13	355
69b69676-1a40-11ea-941b-000d3a38a36f	185	5.0	585.32	2	4	3	9	243
1854e56c-491f-11eb-806e-000d3a38a36f	134	2.0	121.97	3	1	1	5	311
d6ea1074-f1f5-11e9-9346-000d3a38a36f	85	2.0	209.98	3	1	1	5	311

In [64]:

```
# Classifying the customers based on RFM scores
def segment_rfm_score(score):
    if score >= 12:
        return 'High_Value'
    elif score >= 7:
        return 'Mid_Value'
    else:
        return 'Low'

# Applying the segmentation
rfm['Segment'] = rfm['RFM_Score'].apply(segment_rfm_score)

# Showing the results
rfm.head()
```

Out[64]:

master_id	Recency	Frequency	Monetary	R_score	F_score	M_score	RFM_Score	RFM_Segment	Segment
cc294636-19f0-11eb-8d74-000d3a38a36f	94	5.0	939.37	3	4	4	11	344	Mid_Value
f431bd5a-ab7b-11e9-a2fc-000d3a38a36f	104	21.0	2013.55	3	5	5	13	355	High_Value
69b69676-1a40-11ea-941b-000d3a38a36f	185	5.0	585.32	2	4	3	9	243	Mid_Value
1854e56c-491f-11eb-806e-000d3a38a36f	134	2.0	121.97	3	1	1	5	311	Low
d6ea1074-f1f5-11e9-9346-000d3a38a36f	85	2.0	209.98	3	1	1	5	311	Low

In [126…

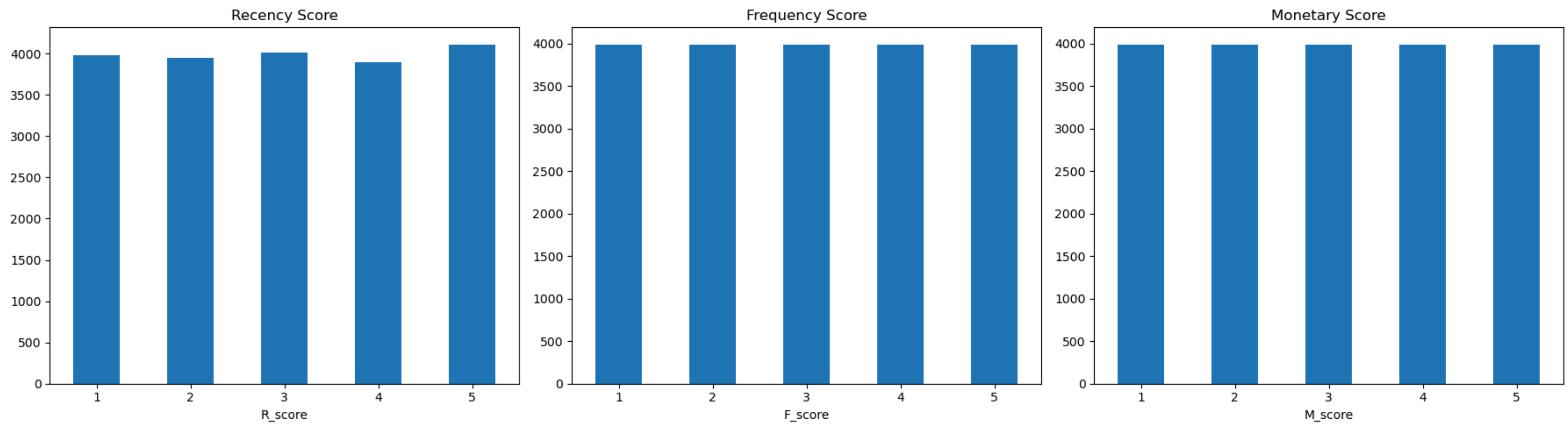
```
#Counting the customers in each segment
segment_counts = rfm["Segment"].value_counts().sort_index()
segment_counts
```

Out[126...

```
Segment
High_Value     4959
Low            4943
Mid_Value     10043
Name: count, dtype: int64
```

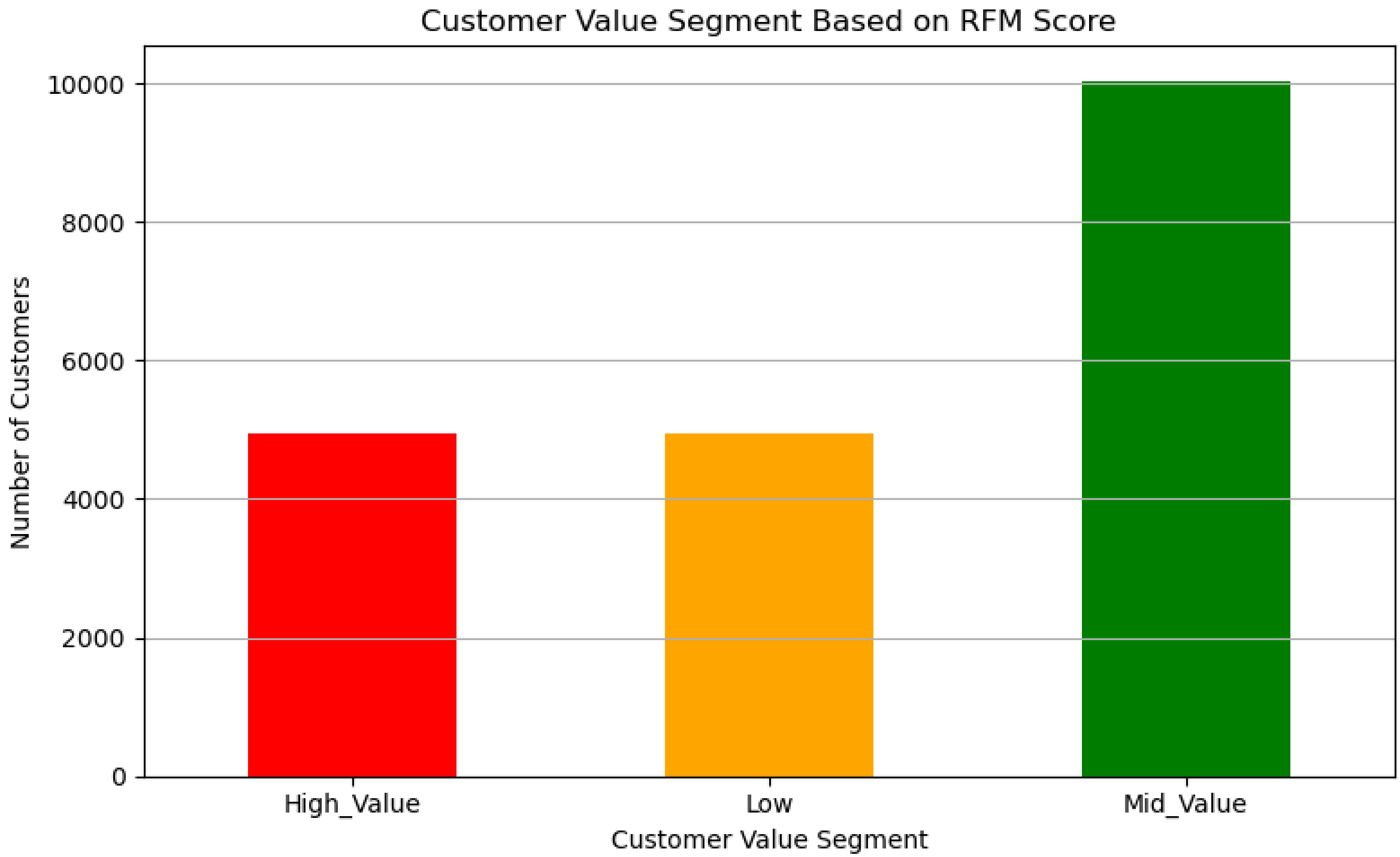
In [114...

```
# RFM Score Bar Chart
fig, axs = plt.subplots(1, 3, figsize=(18,5))
rfm["R_score"].value_counts().sort_index().plot(kind="bar", ax=axs[0], title="Recency Score")
axs[0].set_xticklabels(axs[0].get_xticklabels(), rotation=0)
rfm["F_score"].value_counts().sort_index().plot(kind="bar", ax=axs[1], title="Frequency Score")
axs[1].set_xticklabels(axs[1].get_xticklabels(), rotation=0)
rfm["M_score"].value_counts().sort_index().plot(kind="bar", ax=axs[2], title="Monetary Score")
axs[2].set_xticklabels(axs[2].get_xticklabels(), rotation=0)
plt.tight_layout()
plt.show()
```



In [128...

```
# Segment Bar Chart
plt.figure(figsize= (8, 5))
segment_counts.plot(kind="bar", color=["red", "orange", "green"])
plt.title("Customer Value Segment Based on RFM Score")
plt.xlabel("Customer Value Segment")
plt.ylabel("Number of Customers")
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```



Step 1: RFM Analysis

1. Calculate Recency, Frequency, and Monetary Values

For each customer, we calculated the following metrics:

Recency: Number of days since the customer's last purchase (compared to a reference date). Frequency: Total number of transactions made by the customer (both online and offline). Monetary: Total amount of money spent by the customer (online and offline combined).

2. Assign RFM Scores Using Quintile-Based Scoring (1–5)

Each of the RFM metrics was scored based on quintiles:

Recency: Lower recency (more recent purchase) received a higher score (5 = most recent). Frequency & Monetary: Higher values received higher scores (5 = most frequent or highest spending). These scores were then concatenated into a 3-digit string called RFM_Segment (e.g., "355", "421").

3. Segment Customers Based on Total RFM Score

We summed the individual RFM scores to get a total RFM score per customer. Based on this total score, customers were segmented as follows:

High Value: Customers with high RFM scores (11–15). They buy frequently, recently, and spend more. Mid Value: Customers with moderate RFM scores (7–10). Good buyers but not top-tier. Low Value: Customers with low RFM scores (≤6). They are either inactive, low spenders, or infrequent buyers. Final customer distribution by segment:

High Value: 4,959 customers Mid Value: 10,043 customers Low Value: 4,943 customers

In [ ]: