Data Analysis Of BrandX Retail Promoter Led Sales Promotions

Author: Sohel Japanwala

Email: sohel.japanwala@gmail.com

LinkedIn: https://www.linkedin.com/in/soheljapanwala/ (https://www.linkedin.com/in/soheljapanwala/)

1. Introduction

A certain well known Indian Brand executed a promoter led activity in LFR stores in India. For confidentiality reasons, the brand has been renamed to BrandX.

2. Problem Statement

Brand X would like to :

- Crunch out numbers from the data
- Derive insights generated from the data
- Know where can the promotion can be carried out in the future and where it should be avoided to prevent losses
- Evaluate the factors affecting sales

3. Data Pre Processing

3.1 Import Packages

In [11]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

3.2 Import Data

In [12]:
```
brandX_df=pd.read_excel("BRANDX Sales Test data.xls")
```

3.3 Data Preview

In [13]:
```
brandX_df.head()
```

Out[13]:

	Sr No.	City	Sto... Nam...	Lo...tio...	D...	...eople ...pproached	BRANDX 60ML ...ening Sto...	BRANDX 60ML ... So...	BRANDX 60ML ...osing ...oc...	BRAN... 1... ...p... ...
0	1	Mumbai	Reliance Smart	Malad	2019-03-22 00:00:00	54	14	12	2	
1	2	Mumbai	Big Bazaar	Malad	2019-03-22 00:00:00	47	11	9	2	
2	3	Mumbai	Star Hyper	Andheri	2019-03-22 00:00:00	49	34	5	29	
3	4	Mumbai	Reliance Smart	Santacruz	2019-03-22 00:00:00	44	1	1	0	
4	5	Mumbai	Big Bazaar	Matunga	2019-03-22 00:00:00	51	44	4	40	

3.4 Data Shape

In [14]:
```
brandX_df.shape
```

Out[14]: (429, 14)

3.5 Data Description

In [15]:
```
brandX_df.describe()
```

Out[15]:

	Sr No.	BRANDX 60ML opening Stock	BRANDX 60ML Sold	BRANDX 60ML closing Stock	BRANDX 150ML opening Stock	BRANDX 150ML Sold	BRANDX 150ML closing Stock	B
count	429.000000	429.000000	429.000000	429.000000	429.000000	429.000000	429.000000	429
mean	300.200466	17.009324	4.351981	12.657343	20.766900	4.200466	16.552448	8
std	176.212838	22.132564	4.673411	20.361145	23.204228	4.478599	21.454041	7
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0
25%	160.000000	0.000000	0.000000	0.000000	6.000000	1.000000	1.000000	4
50%	305.000000	12.000000	3.000000	4.000000	14.000000	3.000000	9.000000	7
75%	404.000000	25.000000	6.000000	18.000000	27.000000	6.000000	23.000000	11
max	780.000000	167.000000	24.000000	154.000000	150.000000	31.000000	147.000000	45

3.6 Data Description

In [16]:
```
brandX_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 429 entries, 0 to 428
Data columns (total 14 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Sr No.                       429 non-null    int64 
 1   City                         429 non-null    object
 2   Store Name                   429 non-null    object
 3   Location                     429 non-null    object
 4   Date                         429 non-null    object
 5   People Approached            429 non-null    object
 6   BRANDX 60ML opening Stock    429 non-null    int64 
 7   BRANDX 60ML Sold             429 non-null    int64 
 8   BRANDX 60ML closing Stock    429 non-null    int64 
 9   BRANDX 150ML opening Stock   429 non-null    int64 
 10  BRANDX 150ML Sold            429 non-null    int64 
 11  BRANDX 150ML closing Stock   429 non-null    int64 
 12  Total BRANDX  sold           429 non-null    int64 
 13  Remarks                      429 non-null    object
dtypes: int64(8), object(6)
memory usage: 47.0+ KB
```

Observations On Data Description

- Date not identified as date coloumn
- People approached is not identified as a number column
- There is no missing data

3.7 Data Cleaning

3.7.1 Rectifying Errors In Data

In [18]:
```
brandX_df["Date "]=pd.to_datetime(brandX_df["Date "],format="%Y-%m-%d %H:%M:%
S")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\arrays\datetimes.py in
objects_to_datetime64ns(data, dayfirst, yearfirst, utc, errors, require_iso86
01, allow_object)
   1857         try:
-> 1858             values, tz_parsed = conversion.datetime_to_datetime64(dat
a)
   1859             # If tzaware, these values represent unix timestamps, so
 we

pandas\_libs\tslibs\conversion.pyx in pandas._libs.tslibs.conversion.datetime
_to_datetime64()

TypeError: Unrecognized value type: <class 'str'>

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
<ipython-input-18-4078e832fb1e> in <module>
----> 1 brandX_df["Date "]=pd.to_datetime(brandX_df["Date "],format="%Y-%m-%d
%H:%M:%S")

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\tools\datetimes.py in
to_datetime(arg, errors, dayfirst, yearfirst, utc, format, exact, unit, infer
_datetime_format, origin, cache)
    722                 result = result.tz_localize(tz)
    723     elif isinstance(arg, ABCSeries):
--> 724         cache_array = _maybe_cache(arg, format, cache, convert_listli
ke)
    725         if not cache_array.empty:
    726             result = arg.map(cache_array)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\tools\datetimes.py in
_maybe_cache(arg, format, cache, convert_listlike)
    150         unique_dates = unique(arg)
    151         if len(unique_dates) < len(arg):
--> 152             cache_dates = convert_listlike(unique_dates, format)
    153             cache_array = Series(cache_dates, index=unique_dates)
    154     return cache_array

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\tools\datetimes.py in
_convert_listlike_datetimes(arg, format, name, tz, unit, errors, infer_dateti
me_format, dayfirst, yearfirst, exact)
    438         assert format is None or infer_datetime_format
    439         utc = tz == "utc"
--> 440         result, tz_parsed = objects_to_datetime64ns(
    441             arg,
    442             dayfirst=dayfirst,

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\arrays\datetimes.py in
objects_to_datetime64ns(data, dayfirst, yearfirst, utc, errors, require_iso86
01, allow_object)
   1861             return values.view("i8"), tz_parsed
   1862         except (ValueError, TypeError):
-> 1863             raise e
   1864
```

```
   1865     if tz_parsed is not None:

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\arrays\datetimes.py in
objects_to_datetime64ns(data, dayfirst, yearfirst, utc, errors, require_iso86
01, allow_object)
   1846
   1847     try:
-> 1848         result, tz_parsed = tslib.array_to_datetime(
   1849             data,
   1850             errors=errors,

pandas\_libs\tslib.pyx in pandas._libs.tslib.array_to_datetime()

pandas\_libs\tslib.pyx in pandas._libs.tslib.array_to_datetime()

ValueError: time data 18-042019 doesn't match format specified
```

In [19]:
```
brandX_df["Date "]=brandX_df["Date "].replace(["18-042019"],"2019-04-18 00:00:
00")
```

In [20]:
```
brandX_df["Date "]=pd.to_datetime(brandX_df["Date "],format="%Y-%m-%d %H:%M:%
S")
```

In [21]:
```
brandX_df["People Approached"]=pd.to_numeric(brandX_df["People Approached"])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
pandas\_libs\lib.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "`42"

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
<ipython-input-21-4e0c3ff600a3> in <module>
----> 1 brandX_df["People Approached"]=pd.to_numeric(brandX_df["People Approa
ched"])

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\tools\numeric.py in to
_numeric(arg, errors, downcast)
    147         coerce_numeric = errors not in ("ignore", "raise")
    148         try:
--> 149             values = lib.maybe_convert_numeric(
    150                 values, set(), coerce_numeric=coerce_numeric
    151             )

pandas\_libs\lib.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string "`42" at position 204
```

In [22]:
```
brandX_df["People Approached"]=brandX_df["People Approached"].replace(["`42"],
"42")
```

In [23]:
```
brandX_df["People Approached"]=pd.to_numeric(brandX_df["People Approached"])
brandX_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 429 entries, 0 to 428
Data columns (total 14 columns):
 #   Column                      Non-Null Count  Dtype         
---  ------                      --------------  -----         
 0   Sr No.                      429 non-null    int64         
 1   City                        429 non-null    object        
 2   Store Name                  429 non-null    object        
 3   Location                    429 non-null    object        
 4   Date                        429 non-null    datetime64[ns]
 5   People Approached           429 non-null    int64         
 6   BRANDX 60ML opening Stock   429 non-null    int64         
 7   BRANDX 60ML Sold            429 non-null    int64         
 8   BRANDX 60ML closing Stock   429 non-null    int64         
 9   BRANDX 150ML opening Stock  429 non-null    int64         
 10  BRANDX 150ML Sold           429 non-null    int64         
 11  BRANDX 150ML closing Stock  429 non-null    int64         
 12  Total BRANDX  sold          429 non-null    int64         
 13  Remarks                     429 non-null    object        
dtypes: datetime64[ns](1), int64(9), object(4)
memory usage: 47.0+ KB
```

In [24]:
```
brandX_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 429 entries, 0 to 428
Data columns (total 14 columns):
 #   Column                      Non-Null Count  Dtype         
---  ------                      --------------  -----         
 0   Sr No.                      429 non-null    int64         
 1   City                        429 non-null    object        
 2   Store Name                  429 non-null    object        
 3   Location                    429 non-null    object        
 4   Date                        429 non-null    datetime64[ns]
 5   People Approached           429 non-null    int64         
 6   BRANDX 60ML opening Stock   429 non-null    int64         
 7   BRANDX 60ML Sold            429 non-null    int64         
 8   BRANDX 60ML closing Stock   429 non-null    int64         
 9   BRANDX 150ML opening Stock  429 non-null    int64         
 10  BRANDX 150ML Sold           429 non-null    int64         
 11  BRANDX 150ML closing Stock  429 non-null    int64         
 12  Total BRANDX  sold          429 non-null    int64         
 13  Remarks                     429 non-null    object        
dtypes: datetime64[ns](1), int64(9), object(4)
memory usage: 47.0+ KB
```


Observations:

- Data has been cleaned and formatted

4. Data Analysis

4.1 Adding Month and Day Columns

In [26]:
```
brandX_df["Month"]=brandX_df["Date "].dt.month_name()
brandX_df["Day"]=brandX_df["Date "].dt.day_name()
brandX_df.head()
```

Out[26]:

	Sr No.	City	Store Name	Location	Date	People Approached	BRANDX 60ML opening Stock	BRANDX 60ML Sold	BRANDX 60ML closing Stock	BRANDX 150ML opening Stock
0	1	Mumbai	Reliance Smart	Malad	2019-03-22	54	14	12	2	11
1	2	Mumbai	Big Bazaar	Malad	2019-03-22	47	11	9	2	3
2	3	Mumbai	Star Hyper	Andheri	2019-03-22	49	34	5	29	6
3	4	Mumbai	Reliance Smart	Santacruz	2019-03-22	44	1	1	0	9
4	5	Mumbai	Big Bazaar	Matunga	2019-03-22	51	44	4	40	44

4.2 Analyze Total Number Of Days Of Promotion

In [85]:
```
print("The total number of days of promotion:")
brandX_df["Day"].nunique()
```

```
The total number of days of promotion:
```

Out[85]: 7

In [86]:
```
print("The total number of days of promotion:")
brandX_df["Day"].unique()
```

```
The total number of days of promotion:

array(['Friday', 'Saturday', 'Sunday', 'Tuesday', 'Wednesday', 'Thursday',
       'Monday'], dtype=object)
```

In [87]:
```
print("The total number of months of promotion:")
brandX_df["Month"].unique()
```

```
The total number of months of promotion:

array(['March', 'April', 'May'], dtype=object)
```

In [88]:
```
print("The total number of months of promotion:")
brandX_df["Month"].nunique()
```

```
The total number of months of promotion:

3
```

Observations:

- The promotion was run for 3 months (March, April and May)
- It was run for all 7 days of the week

4.3 Check Total Number Of Locations

In [89]:
```
print("The total number of cities of promotion:")
brandX_df["City"].unique()
```

```
The total number of cities of promotion:

array(['Mumbai', 'Pune'], dtype=object)
```

In [90]:
```
print("The total number of cities of promotion:")
brandX_df["City"].nunique()
```

```
The total number of cities of promotion:

2
```

In [91]:
```
print("The total number of locations of promotion:")
brandX_df["Location"].unique()
```

```
The total number of locations of promotion:

array(['Malad', 'Andheri', 'Santacruz', 'Matunga', 'Vashi', 'Thane',
       'Kharadi,', 'Magarpatta', 'Chinchwad', 'Kothrud', 'Ghatkopar',
       'Powai', 'Aundh', 'Kurla'], dtype=object)
```

In [92]:
```
print("The total number of locations of promotion:")
brandX_df["Location"].nunique()
```

```
The total number of locations of promotion:
```

Out[92]: 14

In [93]:
```
print("The total number of Stores of promotion:")
brandX_df["Store Name "].unique()
```

```
The total number of Stores of promotion:
```

Out[93]:
```
array(['Reliance Smart', 'Big Bazaar', 'Star Hyper', 'Haiko',
       'ABRL Hyper'], dtype=object)
```

In [94]:
```
print("The total number of Stores of promotion:")
brandX_df["Store Name "].nunique()
```

```
The total number of Stores of promotion:
```

Out[94]: 5

Observations:

- The promotion was run in 2 cities: Mumbai and Pune
- It was run in a total of 14 locations : 'Malad', 'Andheri', 'Santacruz', 'Matunga', 'Vashi', 'Thane', 'Kharadi,', 'Magarpatta', 'Chinchwad', 'Kothruc 'Gha pa ', 'Pow ' 'A ndh 'K a
- It was run in 5 major stores : 'Reliance S ha , Big Ba zaar' Star Hy er 'H ik ', AB RL Hy er

4.4 Analyze Total Sales

In [37]:
```
print("The total sales:")
brandX_df["Total BRANDX  sold"].sum()
```

```
The total sales:
```

Out[37]: 3669

In [38]:
```
print("The average sales:")
brandX_df["Total BRANDX  sold"].mean()
```

```
The average sales:
```

Out[38]: 8.552447552447552

In [95]:
```
print("The median sales:")
brandX_df["Total BRANDX  sold"].median()
```

```
The median sales:
```

Out[95]: 7.0

4.5 Check Bifurcated Sales

4.5.1 Check Total Sales City Wise

In [40]:
```
brandX_dfCityGroup=brandX_df.groupby("City")
```

In [41]:
```
brandXMumbai = brandX_dfCityGroup.get_group("Mumbai")
print("The total sales in Mumbai:")
brandXMumbai["Total BRANDX  sold"].sum()
```

```
The total sales in Mumbai:
```

Out[41]: 2603

In [42]:
```
print("The average sales in Mumbai:")
brandXMumbai["Total BRANDX  sold"].mean()
```

```
The average sales in Mumbai:
```

Out[42]: 8.676666666666666

In [43]:
```
brandXPune = brandX_dfCityGroup.get_group("Pune")
print("The total sales in Pune:")
brandXPune["Total BRANDX  sold"].sum()
```

```
The total sales in Pune:
```

Out[43]: 1066

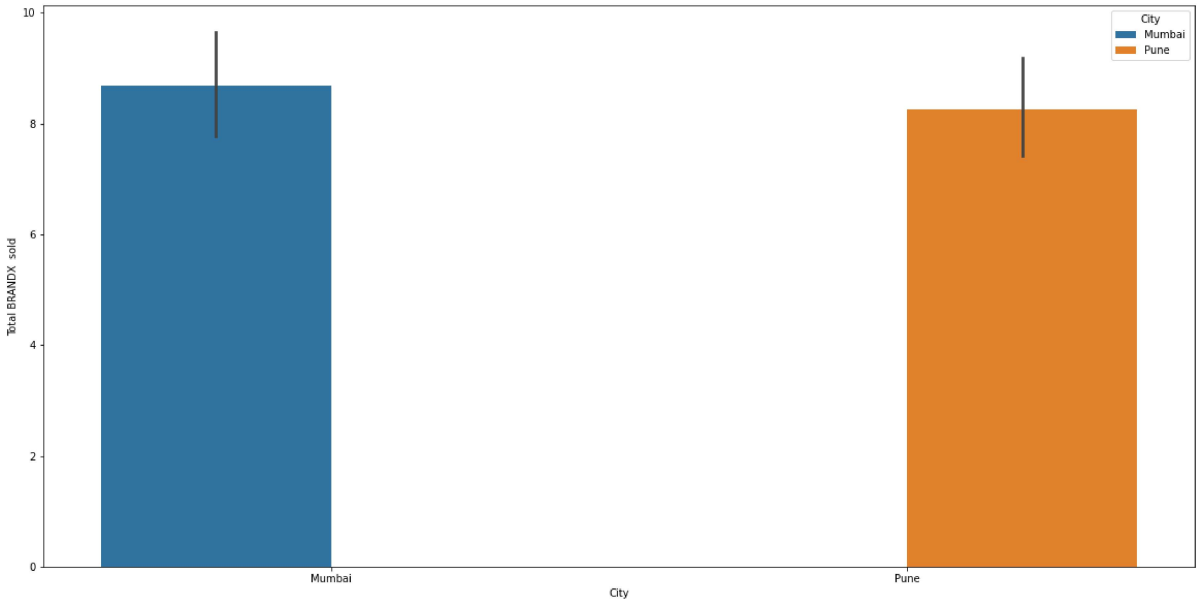
In [44]:
```
print("The average sales in Pune:")
brandXPune["Total BRANDX  sold"].mean()
```

```
The average sales in Pune:
```

Out[44]: 8.263565891472869

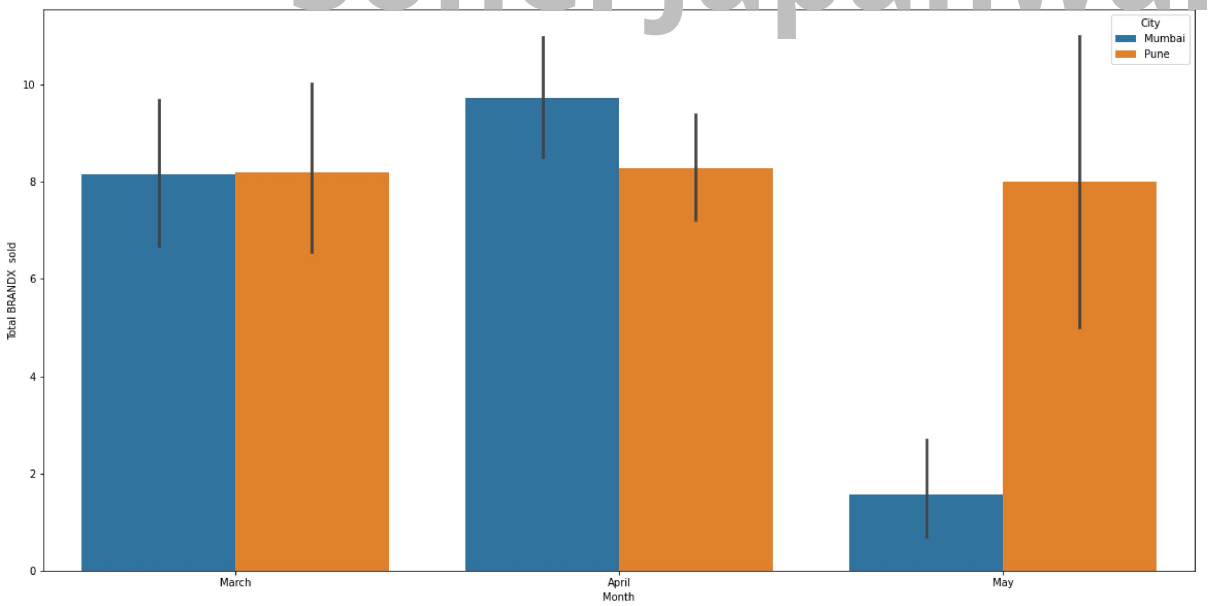
In [75]:
```
plt.figure(figsize=(20,10))
sns.barplot(y=brandX_df["Total BRANDX  sold"], x=brandX_df["City"], hue=brandX_df["City"], dodge=True)
```

Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x1e4445db760>



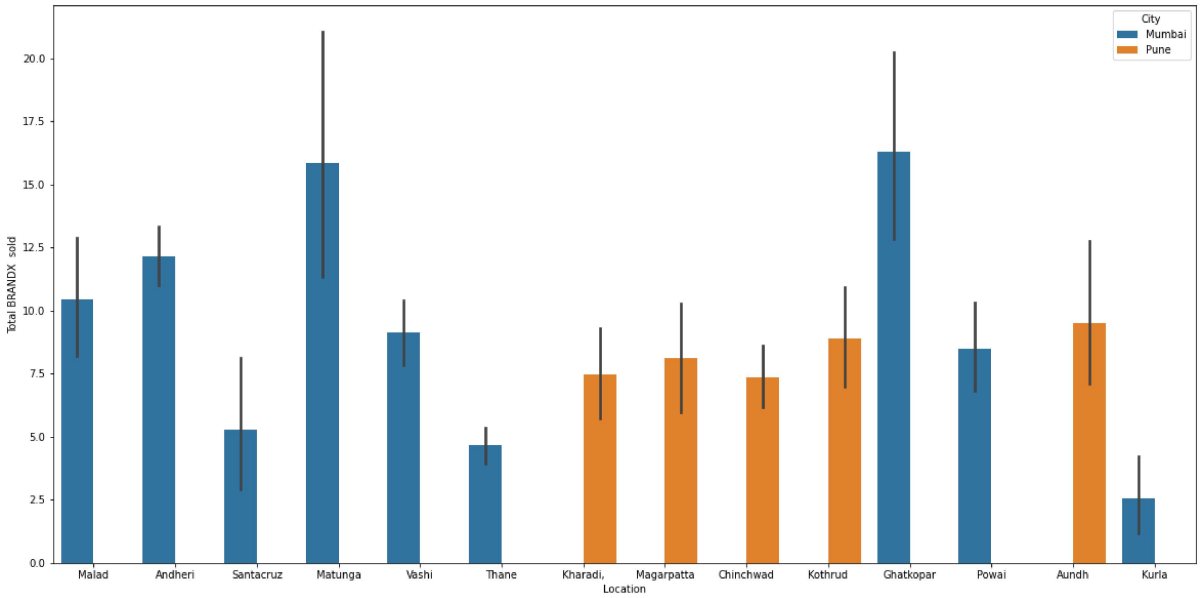
In [76]:
```
plt.figure(figsize=(20,10))
sns.barplot(y=brandX_df["Total BRANDX  sold"],x=brandX_df["Month"], hue=brandX_df["City"])
```

Out[76]: <matplotlib.axes._subplots.AxesSubplot at 0x1e44e5fa0>



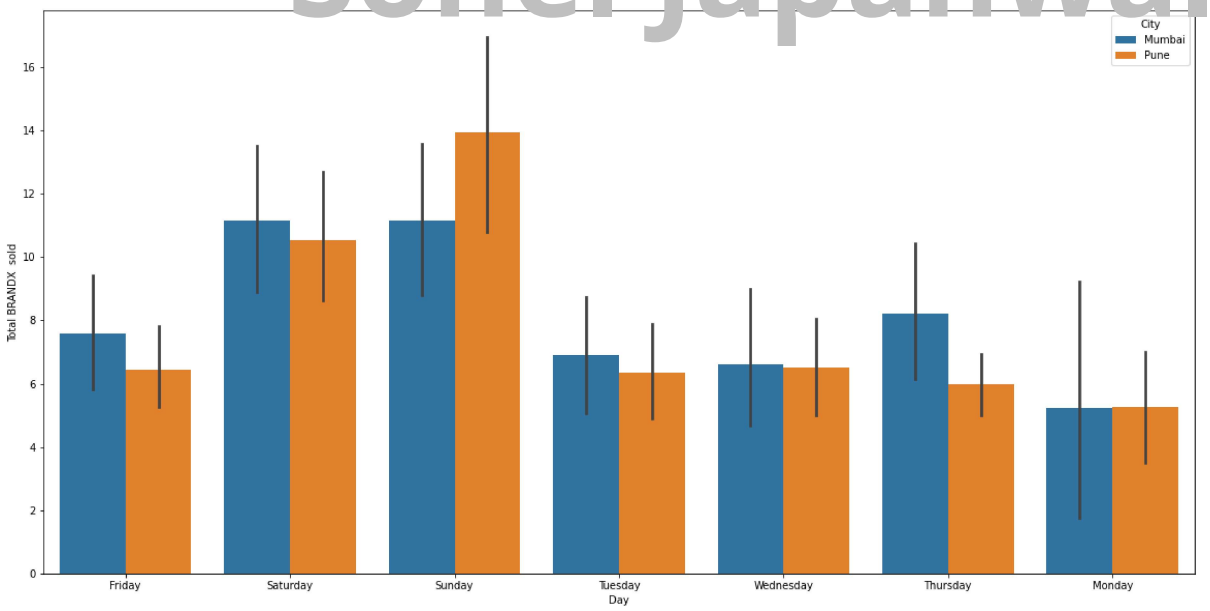
In [47]:
```
plt.figure(figsize=(20,10))
sns.barplot(y=brandX_df["Total BRANDX  sold"], x=brandX_df["Location"], hue=br
andX_df["City"])
```

Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1e4413d31c0>



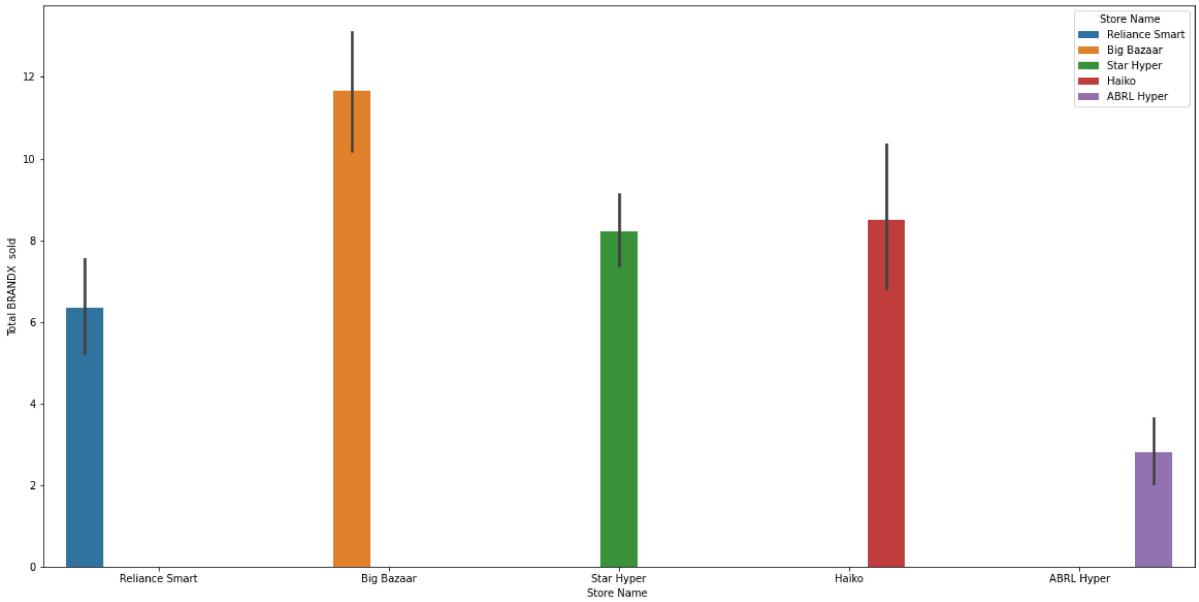
In [48]:
```
plt.figure(figsize=(20,10))
sns.barplot(y=brandX_df["Total BRANDX  sold"], x=brandX_df["Day"], hue=brandX_
df["City"])
```

Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x1e44[illegible]>



In [117]:
```
plt.figure(figsize=(20,10))
sns.barplot(y=brandX_df["Total BRANDX  sold"], x=brandX_df["Store Name "], hue
=brandX_df["Store Name "])
```

Out[117]: <matplotlib.axes._subplots.AxesSubplot at 0x1e4486edfd0>



Observations:

- The total sales in Mumbai was mor[illegible]26[illegible]
- The sales in Pune was lesser :106[illegible]
- The average sales was only a little higher in Mumbai at 8.6 units
- The sales were higher in April followed by March and then May
- The sales were greatest on Sunday and least on Monday
- Big Bazaar outlets had the most sales whereas ABRL was the least

4.6 Factors Affecting Sales

4.6.1 Add Total Opening Stock Column

In [58]:
```
brandX_df['Total Opening Stock'] =brandX_df["BRANDX 60ML opening Stock "]+bran
dX_df["BRANDX 150ML opening Stock "]
```

4.6.2 Changing Month and Weekday To Number

In [68]:
```
brandX_df["Month_Number"]=brandX_df["Date "].dt.month
brandX_df["Day_Number"]=brandX_df["Date "].dt.weekday
```

In [69]:
```
brandX_df.columns
```

Out[69]:
```
Index(['Sr No. ', 'City', 'Store Name ', 'Location', 'Date ',
       'People Approached', 'BRANDX 60ML opening Stock ', 'BRANDX 60ML Sold',
       'BRANDX 60ML closing Stock ', 'BRANDX 150ML opening Stock ',
       'BRANDX 150ML Sold', 'BRANDX 150ML closing Stock ',
       'Total BRANDX  sold', 'Remarks', 'Month', 'Day', 'Total Opening Stoc
k',
       'Month_number', 'Month_Number', 'Day_Number'],
      dtype='object')
```

In [60]:
```
brandX_df.head(2)
```

Out[60]:

	Sr No.	City	Store Name	Location	Date	People Approached	BRANDX 60ML opening Stock	BRANDX 60ML Sold	BRANDX 60ML closing Stock	BRANDX 150ML opening Stock
0	1	Mumbai	Reliance Smart	Malad	2019-03-22	54	14	12	2	11
1	2	Mumbai	Big Bazaar	Malad	2019-03-22	47	11	9	2	3

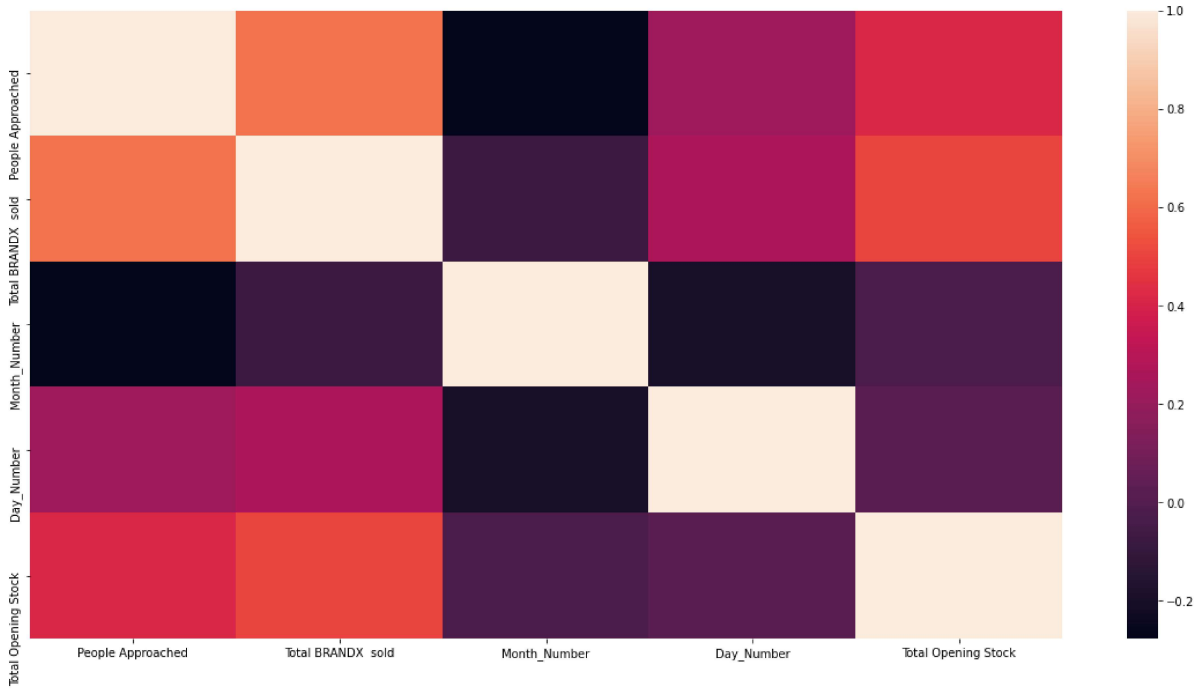
In [96]:
```
brandX_df[salesFact rs].corr()
```


Out[96]:

	People Approached	Total BRANDX sold	Month_Number	Day_Number	Total Opening Stock
People Approached	1.000000	0.622920	-0.277241	0.224913	0.418209
Total BRANDX sold	0.622920	1.000000	-0.070664	0.270324	0.505675
Month_Number	-0.277241	-0.070664	1.000000	-0.195659	-0.019448
Day_Number	0.224913	0.270324	-0.195659	1.000000	0.025052
Total Opening Stock	0.418209	0.505675	-0.019448	0.025052	1.000000

In [78]:
```
salesFactors=['People Approached','Total BRANDX  sold','Month_Number', 'Day_Number','Total Opening Stock']
plt.figure(figsize=(20,10))
sns.heatmap(brandX_df[salesFactors].corr())
```

Out[78]: <matplotlib.axes._subplots.AxesSubplot at 0x1e445c46760>



Observations:

- Total opening stocks affect the promoters confidence to approach people
- Total opening stocks affect the total sales
- The day of the week affect total people approached
- The day of the week affect sales

4.7 Listing Cities/ Locations/ Store Basis Sales Figures

In [106]:
```
print("Total Sales:")
brandX_df.groupby("City")["Total BRANDX  sold"].sum()
```

```
Total Sales:

City
Mumbai    2603
Pune      1066
Name: Total BRANDX  sold, dtype: int64
```

In [112]:
```
print("Total Sales:")
locationWiseSales=brandX_df.groupby("Location")["Total BRANDX  sold"].sum()
locationWiseSales.sort_values()
```

```
Total Sales:

Location
Kurla          71
Powai         102
Santacruz     137
Chinchwad     191
Kharadi,      194
Magarpatta    203
Kothrud       231
Vashi         237
Aundh         247
Andheri       316
Thane         363
Matunga       412
Ghatkopar     423
Malad         542
Name: Total BRANDX  sold, dtype: int64
```

In [114]:
```
print("Total Sales:")
storeWiseSales=brandX_df.groupby("Store Name ")["Total BRANDX  sold"].sum()
storeWiseSales.sort_values()
```

```
Total Sales:

Store Name
ABRL Hyper          7[illegible]
Haiko               102
Reliance Smart      837
Star Hyper          847
Big Bazaar         1807
Name: Total BRANDX  sold, dtype: int64
```


In [115]:
```
print("Total Sales:")
weekdayWiseSales=brandX_df.groupby("Day")["Total BRANDX  sold"].sum()
weekdayWiseSales.sort_values()
```

```
Total Sales:

Day
Monday         68
Tuesday       411
Wednesday     427
Thursday      484
Friday        519
Sunday        871
Saturday      889
Name: Total BRANDX  sold, dtype: int64
```

5. Actionable Insights

To prevent losses, promotions can be avoided:

- At locations where numbers are the least
- At LFR Stores where numbers are the least
- At the start of the week

Sales will improve:

- if promoters approach more people
- if stocks are replenished
- If sales are pushed near the weekends

Data Analysis Of BrandX Retail Promoter Led Sales Promotions

Author: Sohel Japanwala

Email: sohel.japanwala@gmail.com

LinkedIn: https://www.linkedin.com/in/sohelljapanwala/ (https://www.linkedin.com/in/sohelljapanwala/)