Impact of Promotions on Sales and Customers

Table of Content

Introduction

The following are descriptions of the columns in the dataset.

Id - an Id that represents a (Store, Date) duple within the test set Store - a unique Id for each store Sales - the turnover for any given day Customers - the number of customers on a given day Open - an indicator for whether the store was open: 0 = closed, 1 = open StateHoliday - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None SchoolHoliday - indicates if the (Store, Date) was affected by the closure of public schools StoreType - differentiates between 4 different store models: a, b, c, d Assortment - describes an assortment level: a = basic, b = extra, c = extended CompetitionDistance - distance in meters to the nearest competitor store CompetitionOpenSince[Month/Year] - gives the approximate year and month of the time the nearest competitor was opened Promo - indicates whether a store is running a promo on that day Promo2 - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating Promo2Since[Year/Week] - describes the year and calendar week when the store started participating in Promo2 PromoInterval - describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

Data Wrangling

In [1]:

```
#import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```
#load the dataset with pandas
df = pd.read_csv('store.csv')
train_df = pd.read_csv('train.csv', parse_dates=True,low_memory=False)
df.head()
```

Out[2]:

	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	Competitior
0	1	c	a	1270.0	9.0	
1	2	a	a	570.0	11.0	
2	3	a	a	14130.0	12.0	
3	4	c	c	620.0	9.0	
4	5	a	a	29910.0	4.0	

In [3]:

```
train_df.head()
```

Out[3]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-31	8314	821	1	1	0	1
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

In [4]:

```
# check for the shape of the dataset
df.shape
```

Out[4]:

```
(1115, 10)
```

In [5]:

```
train_df.shape
```

Out[5]:

```
(1017209, 9)
```

In [6]:

```
# check for the dataset summary
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1115 entries, 0 to 1114
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Store                      1115 non-null   int64  
 1   StoreType                  1115 non-null   object 
 2   Assortment                 1115 non-null   object 
 3   CompetitionDistance        1112 non-null   float64
 4   CompetitionOpenSinceMonth  761 non-null    float64
 5   CompetitionOpenSinceYear   761 non-null    float64
 6   Promo2                     1115 non-null   int64  
 7   Promo2SinceWeek            571 non-null    float64
 8   Promo2SinceYear            571 non-null    float64
 9   PromoInterval              571 non-null    object 
dtypes: float64(5), int64(2), object(3)
memory usage: 87.2+ KB
```

In [7]:

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1017209 entries, 0 to 1017208
Data columns (total 9 columns):
 #   Column         Non-Null Count    Dtype 
---  ------         --------------    ----- 
 0   Store          1017209 non-null  int64 
 1   DayOfWeek      1017209 non-null  int64 
 2   Date           1017209 non-null  object
 3   Sales          1017209 non-null  int64 
 4   Customers      1017209 non-null  int64 
 5   Open           1017209 non-null  int64 
 6   Promo          1017209 non-null  int64 
 7   StateHoliday   1017209 non-null  object
 8   SchoolHoliday  1017209 non-null  int64 
dtypes: int64(7), object(2)
memory usage: 69.8+ MB
```

In [8]:

```
#check the summary statistics
df.describe()
```

Out[8]:

	Store	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear
count	1115.00000	1112.000000	761.000000	761.000000
mean	558.00000	5404.901079	7.224704	2008.668857
std	322.01708	7663.174720	3.212348	6.195983
min	1.00000	20.000000	1.000000	1900.000000
25%	279.50000	717.500000	4.000000	2006.000000
50%	558.00000	2325.000000	8.000000	2010.000000
75%	836.50000	6882.500000	10.000000	2013.000000
max	1115.00000	75860.000000	12.000000	2015.000000

In [9]:

```
train_df.describe()
```

Out[9]:

	Store	DayOfWeek	Sales	Customers	Open	Promo	S
count	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	
mean	5.584297e+02	3.998341e+00	5.773819e+03	6.331459e+02	8.301067e-01	3.815145e-01	
std	3.219087e+02	1.997391e+00	3.849926e+03	4.644117e+02	3.755392e-01	4.857586e-01	
min	1.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	
25%	2.800000e+02	2.000000e+00	3.727000e+03	4.050000e+02	1.000000e+00	0.000000e+00	
50%	5.580000e+02	4.000000e+00	5.744000e+03	6.090000e+02	1.000000e+00	0.000000e+00	
75%	8.380000e+02	6.000000e+00	7.856000e+03	8.370000e+02	1.000000e+00	1.000000e+00	
max	1.115000e+03	7.000000e+00	4.155100e+04	7.388000e+03	1.000000e+00	1.000000e+00	

In [10]:

```
#check the data types
df.dtypes
```

Out[10]:

```
Store                         int64
StoreType                    object
Assortment                   object
CompetitionDistance         float64
CompetitionOpenSinceMonth   float64
CompetitionOpenSinceYear    float64
Promo2                        int64
Promo2SinceWeek             float64
Promo2SinceYear             float64
PromoInterval                object
dtype: object
```

In [11]:

```
train_df.dtypes
```

Out[11]:

```
Store             int64
DayOfWeek         int64
Date             object
Sales             int64
Customers         int64
Open              int64
Promo             int64
StateHoliday     object
SchoolHoliday     int64
dtype: object
```

In [12]:

```
#check for duplicated values
df.duplicated().sum()
```

Out[12]:

```
0
```

In [13]:

```
train_df.duplicated().sum()
```

Out[13]:

```
0
```

In [14]:

```
#check for null values
df.isna().sum()
```

Out[14]:

```
Store                         0
StoreType                     0
Assortment                    0
CompetitionDistance           3
CompetitionOpenSinceMonth   354
CompetitionOpenSinceYear    354
Promo2                        0
Promo2SinceWeek             544
Promo2SinceYear             544
PromoInterval               544
dtype: int64
```

In [15]:

```
train_df.isna().sum()
```

Out[15]:

```
Store            0
DayOfWeek        0
Date             0
Sales            0
Customers        0
Open             0
Promo            0
StateHoliday     0
SchoolHoliday    0
dtype: int64
```

In [16]:

```
#check for unique values
df.nunique()
```

Out[16]:

```
Store                       1115
StoreType                      4
Assortment                     3
CompetitionDistance          654
CompetitionOpenSinceMonth     12
CompetitionOpenSinceYear      23
Promo2                         2
Promo2SinceWeek               24
Promo2SinceYear                7
PromoInterval                  3
dtype: int64
```

In [17]:

```
train_df.nunique()
```

Out[17]:

```
Store             1115
DayOfWeek            7
Date               942
Sales            21734
Customers         4086
Open                 2
Promo                2
StateHoliday         4
SchoolHoliday        2
dtype: int64
```

In [18]:

```
#check unique values for each column
df.StoreType.unique()
```

Out[18]:

```
array(['c', 'a', 'd', 'b'], dtype=object)
```

In [19]:

```
df.Assortment.unique()
```

Out[19]:

```
array(['a', 'c', 'b'], dtype=object)
```

In [20]:

```
df.CompetitionOpenSinceMonth.unique()
```

Out[20]:

```
array([ 9., 11., 12.,  4., 10.,  8., nan,  3.,  6.,  5.,  1.,  2.,  7.])
```

In [21]:

```
df.CompetitionOpenSinceYear.unique()
```

Out[21]:

```
array([2008., 2007., 2006., 2009., 2015., 2013., 2014., 2000., 2011.,
         nan, 2010., 2005., 1999., 2003., 2012., 2004., 2002., 1961.,
       1995., 2001., 1990., 1994., 1900., 1998.])
```

In [22]:

```
df.Promo2SinceWeek.unique()
```

Out[22]:

```
array([nan, 13., 14.,  1., 45., 40., 26., 22.,  5.,  6., 10., 31., 37.,
        9., 39., 27., 18., 35., 23., 48., 36., 50., 44., 49., 28.])
```

In [23]:

```
df.Promo2SinceYear.unique()
```

Out[23]:

```
array([  nan, 2010., 2011., 2012., 2009., 2014., 2015., 2013.])
```

In [24]:

```
df.PromoInterval.unique()
```

Out[24]:

```
array([nan, 'Jan,Apr,Jul,Oct', 'Feb,May,Aug,Nov', 'Mar,Jun,Sept,Dec'],
      dtype=object)
```

In [25]:

```
df.Promo2.unique()
```

Out[25]:

```
array([0, 1], dtype=int64)
```

In [26]:

```
train_df.DayOfWeek.unique()
```

Out[26]:

```
array([5, 4, 3, 2, 1, 7, 6], dtype=int64)
```

In [27]:

```
train_df.Open.unique()
```

Out[27]:

```
array([1, 0], dtype=int64)
```

In [28]:

```
train_df.Promo.unique()
```

Out[28]:

```
array([1, 0], dtype=int64)
```

In [29]:

```
train_df.StateHoliday.unique()
```

Out[29]:

```
array(['0', 'a', 'b', 'c'], dtype=object)
```

In [30]:

```
train_df.SchoolHoliday.unique()
```

Out[30]:

```
array([1, 0], dtype=int64)
```

Summary of the analysis carried out above

Store Dataset

- There are 1115 rows and 10 columns in the dataset
- Competition Open Since Month column should be converted to object
- Competition Open Since Year column should be converted to integer
- Promo 2 column should be converted to object
- Promo 2 Since Week should be converted to datetime/object
- Promo 2 Since Year should be converted to datetime/object
- There are no duplicate values
- There are missing values which will be dropped or filled
- There are 4 types of store models
- There are 3 promo intervals
- Stores either participated in promos or not
- There are 3 assortment store levels; level: a = basic, b = extra, c = extended

Train Dataset

- There are 1017209 rows and 9 columns in the dataset
- Date should be converted to Datetime
- Convert Open, Date, Promo and SchoolHoliday to objects because they are categorical data

Cleaning

In [31]:

```
# change wrong data types to the correct one
df['CompetitionOpenSinceMonth'] = df['CompetitionOpenSinceMonth'].astype('object')
df['Promo2SinceWeek'] = df['Promo2SinceWeek'].astype('object')



train_df['Date'] = pd.to_datetime(train_df['Date'])
```

In [32]:

```
train_df.head()
```

Out[32]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-31	8314	821	1	1	0	1
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

In [33]:

```
df['Promo2'] = df['Promo2'].astype('object')
train_df['Open'] = train_df['Open'].astype('object')
train_df['Promo'] = train_df['Promo'].astype('object')
train_df['SchoolHoliday'] = train_df['SchoolHoliday'].astype('object')
```

In [34]:

```
#check for change
df.dtypes
```

Out[34]:

```
Store                          int64
StoreType                     object
Assortment                    object
CompetitionDistance          float64
CompetitionOpenSinceMonth     object
CompetitionOpenSinceYear     float64
Promo2                        object
Promo2SinceWeek               object
Promo2SinceYear              float64
PromoInterval                 object
dtype: object
```

In [35]:

```
train_df.dtypes
```

Out[35]:

```
Store                    int64
DayOfWeek                int64
Date            datetime64[ns]
Sales                    int64
Customers                int64
Open                    object
Promo                   object
StateHoliday            object
SchoolHoliday           object
dtype: object
```

In [36]:

```
#fill all null values with empty
df.fillna('Empty', axis=1, inplace=True)
```

In [37]:

```
#check for change
df.isna().sum()
```

Out[37]:

```
Store                       0
StoreType                   0
Assortment                  0
CompetitionDistance         0
CompetitionOpenSinceMonth   0
CompetitionOpenSinceYear    0
Promo2                      0
Promo2SinceWeek             0
Promo2SinceYear             0
PromoInterval               0
dtype: int64
```

In [38]:

```
#extract the year, month and day from date
train_df['Year']=train_df['Date'].dt.year
train_df['Month']=train_df['Date'].dt.month
train_df['Day']=train_df['Date'].dt.day
```

In [39]:

```
#check for change
train_df.head()
```

Out[39]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	Ye
0	1	5	2015-07-31	5263	555	1	1	0	1	20
1	2	5	2015-07-31	6064	625	1	1	0	1	20
2	3	5	2015-07-31	8314	821	1	1	0	1	20
3	4	5	2015-07-31	13995	1498	1	1	0	1	20
4	5	5	2015-07-31	4822	559	1	1	0	1	20

In [40]:

```
# check to see the unique years
train_df.Year.unique()
```

Out[40]:

```
array([2015, 2014, 2013], dtype=int64)
```

In [41]:

```
#merge both datasets together
train_store=pd.merge(train_df, df, how='inner', on='Store')
```

In [42]:

```
#view
train_store.tail()
```

Out[42]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHolid
1017204	1115	6	2013-01-05	4771	339	1	0	0	
1017205	1115	5	2013-01-04	4540	326	1	0	0	
1017206	1115	4	2013-01-03	4297	300	1	0	0	
1017207	1115	3	2013-01-02	3697	305	1	0	0	
1017208	1115	2	2013-01-01	0	0	0	0	a	

5 rows × 21 columns

Assessment

In [43]:

```
train_store.shape
```

Out[43]:

```
(1017209, 21)
```

In [44]:

```
train_store.isna().sum()
```

Out[44]:

```
Store                        0
DayOfWeek                    0
Date                         0
Sales                        0
Customers                    0
Open                         0
Promo                        0
StateHoliday                 0
SchoolHoliday                0
Year                         0
Month                        0
Day                          0
StoreType                    0
Assortment                   0
CompetitionDistance          0
CompetitionOpenSinceMonth    0
CompetitionOpenSinceYear     0
Promo2                       0
Promo2SinceWeek              0
Promo2SinceYear              0
PromoInterval                0
dtype: int64
```

In [45]:

```
# find out how many assorments each store has
train_store.groupby(['StoreType','Assortment']).Assortment.count()
```

Out[45]:

```
StoreType  Assortment
a          a             346389
           c             205238
b          a               6594
           b               8294
           c                942
c          a              70878
           c              65962
d          a             113584
           c             199328
Name: Assortment, dtype: int64
```

Store A has both a and c assortments with more of a than c Store B has a, b and c assortments with more of b than others Store C has a and c assortments with more of a than c Store D has both a and c assortments with more of c than a

Store A has the most A assortments

Stores A and D account for the most assortment levels

What day of the week do stores have more activity

In [46]:

```
train_store.DayOfWeek.value_counts()
```

Out[46]:

```
5    145845
4    145845
3    145665
2    145664
1    144730
7    144730
6    144730
Name: DayOfWeek, dtype: int64
```

Stores have more activities Tuesdays to Fridays, while less activities from Saturdays to Mondays

What year had the most promotions done?

In [47]:

```
train_store.groupby(['Year','Promo']).Promo.count()
```

Out[47]:

```
Year  Promo
2013  0        256449
      1        150525
2014  0        231075
      1        142780
2015  0        141605
      1         94775
Name: Promo, dtype: int64
```

2013 had the most promotions done Next, the impact of promotions in year 2013 and others will be looked at

What Impact of Promotions on Sales over the months

Use a catplot to derive the insight of this

In [48]:

```
# only select stores that were open
store_open = train_store[train_store['Open'] == 1]
store_open.head()
```

Out[48]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	Ye
0	1	5	2015-07-31	5263	555	1	1	0	1	20
1	1	4	2015-07-30	5020	546	1	1	0	1	20
2	1	3	2015-07-29	4782	523	1	1	0	1	20
3	1	2	2015-07-28	5011	560	1	1	0	1	20
4	1	1	2015-07-27	6102	612	1	1	0	1	20

5 rows × 21 columns

In [49]:

```
# get data for just 2013
store_open13 = store_open[store_open.Year == 2013]
store_open13.head()
```

Out[49]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
577	1	2	2013-12-31	2362	298	1	0	0	1
578	1	1	2013-12-30	7193	796	1	0	0	1
580	1	6	2013-12-28	5659	716	1	0	0	1
581	1	5	2013-12-27	6110	737	1	0	0	1
584	1	2	2013-12-24	3204	385	1	0	0	1

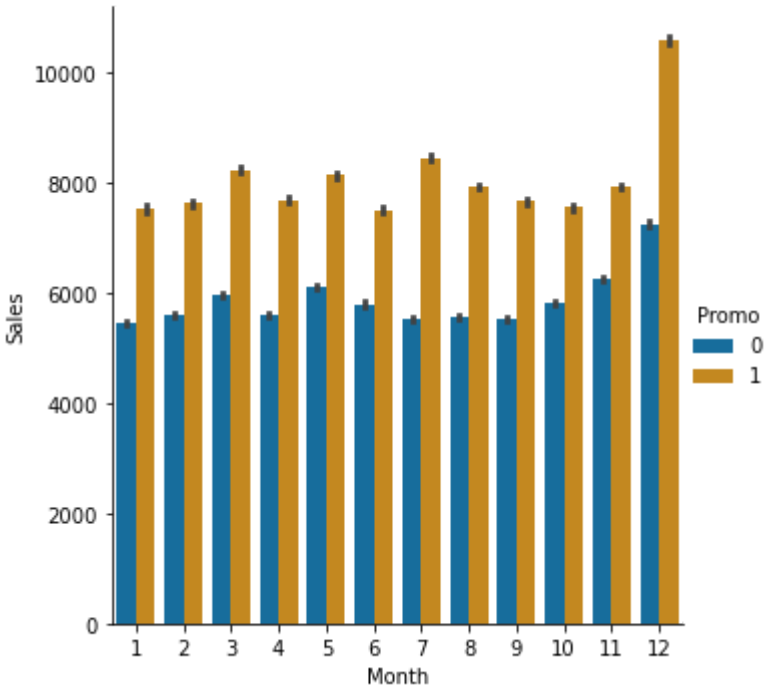
5 rows × 21 columns

In [50]:

```
#impact of promotion on sales in 2013
sns.catplot(data=store_open13, x='Month', y='Sales', palette='colorblind', hue='Promo', kin
```

Out[50]:

```
<seaborn.axisgrid.FacetGrid at 0x1f887ef5e80>
```



In [51]:

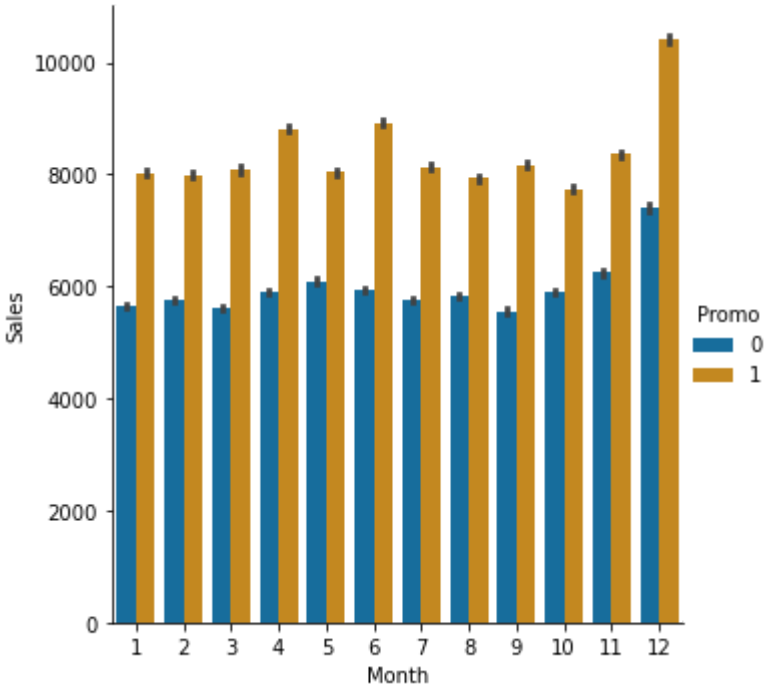
```
# get data for 2014

store_open14 = store_open[store_open.Year == 2014]


sns.catplot(data=store_open14, x='Month', y='Sales', palette='colorblind', hue='Promo', kin
```

Out[51]:

```
<seaborn.axisgrid.FacetGrid at 0x1f887fc9700>
```



In [52]:

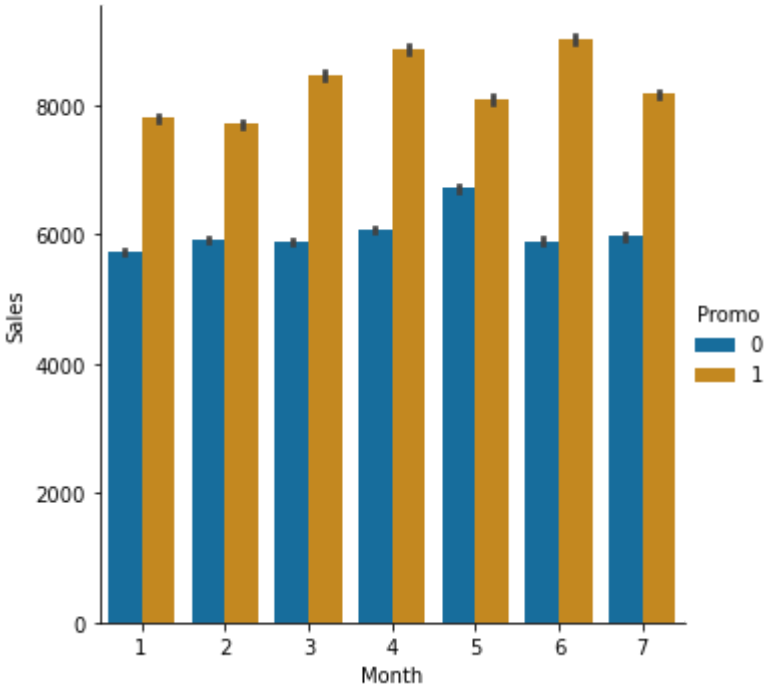
```
# get data for 2015

store_open15 = store_open[store_open.Year == 2015]


sns.catplot(data=store_open15, x='Month', y='Sales', palette='colorblind', hue='Promo', kin
```

Out[52]:

```
<seaborn.axisgrid.FacetGrid at 0x1f8887128e0>
```



From the above analysis, In 2013, sales were more during promotions in December, this could be due to the fact that december is a festive month and people will be prone to shopping more during december. Next month will be July, this is because most people will want to shop for summer.

In 2014, sales were more in December too, same reaosn with 2013

In 2015, the data stops in July and July also has the highest sale during promotions

Without promotions, customers bought more in May 2015

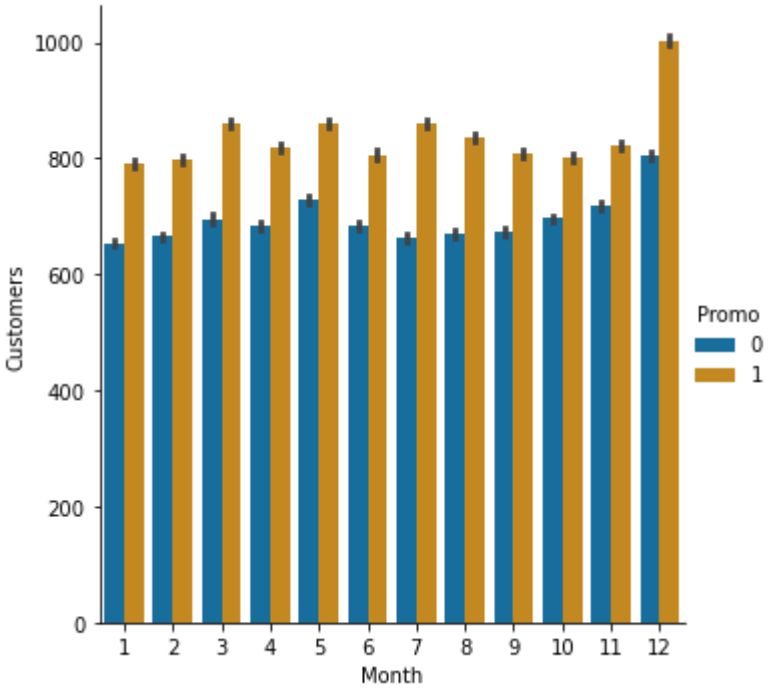
Impact of Promtions on Customers over the months

In [53]:

```
# for 2013
sns.catplot(data=store_open13, x='Month', y='Customers', palette='colorblind', hue='Promo',
```

Out[53]:

```
<seaborn.axisgrid.FacetGrid at 0x1f887fd1130>
```

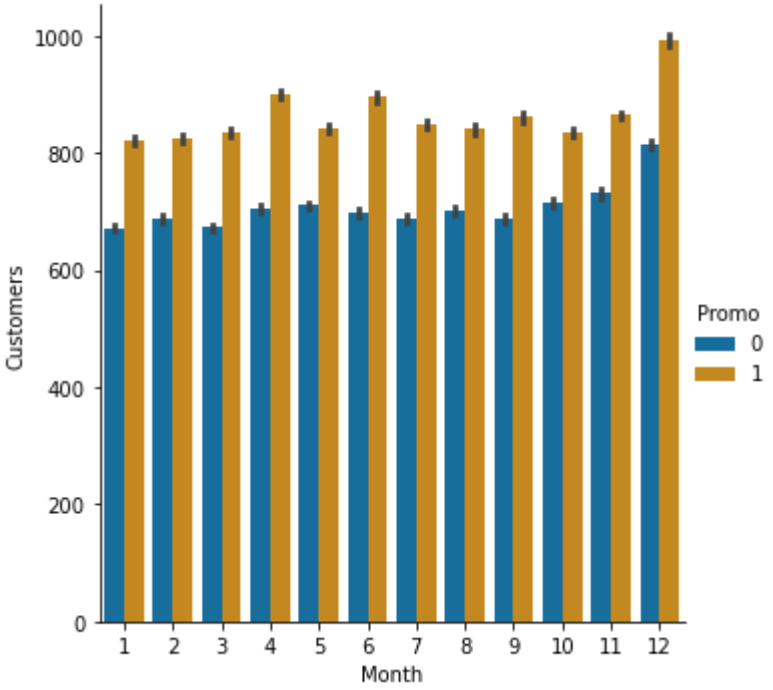


In [54]:

```
#for 2014
sns.catplot(data=store_open14, x='Month', y='Customers', palette='colorblind', hue='Promo',
```

Out[54]:

```
<seaborn.axisgrid.FacetGrid at 0x1f88812c280>
```

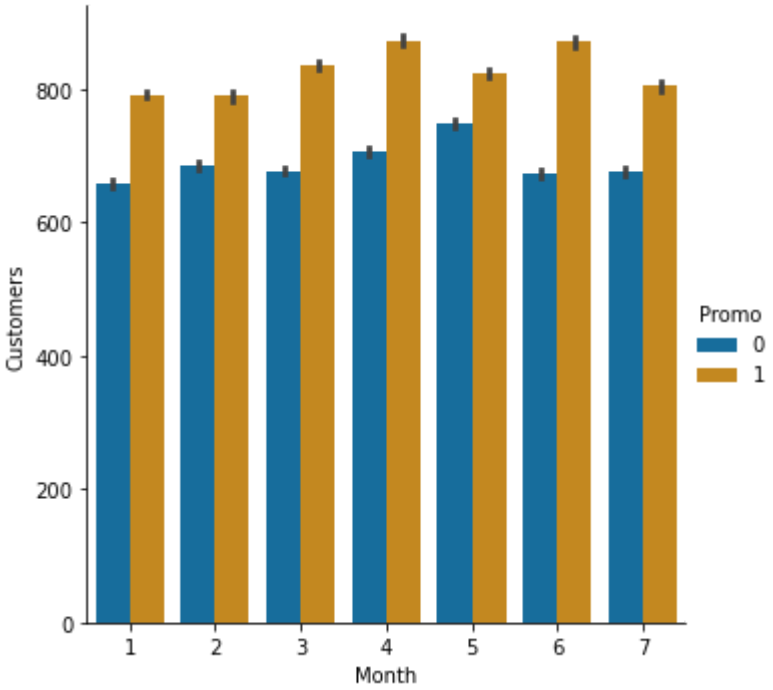


In [55]:

```
#for 2015
sns.catplot(data=store_open15, x='Month', y='Customers', palette='colorblind', hue='Promo',
```

Out[55]:

```
<seaborn.axisgrid.FacetGrid at 0x1f88d4246a0>
```



From the above analysis In 2013 and 2014, customers bought more from stores in December during promotions

In 2015, customers bought more from stores in April and June during promotions

Without promotions, customers bought more in May 2015

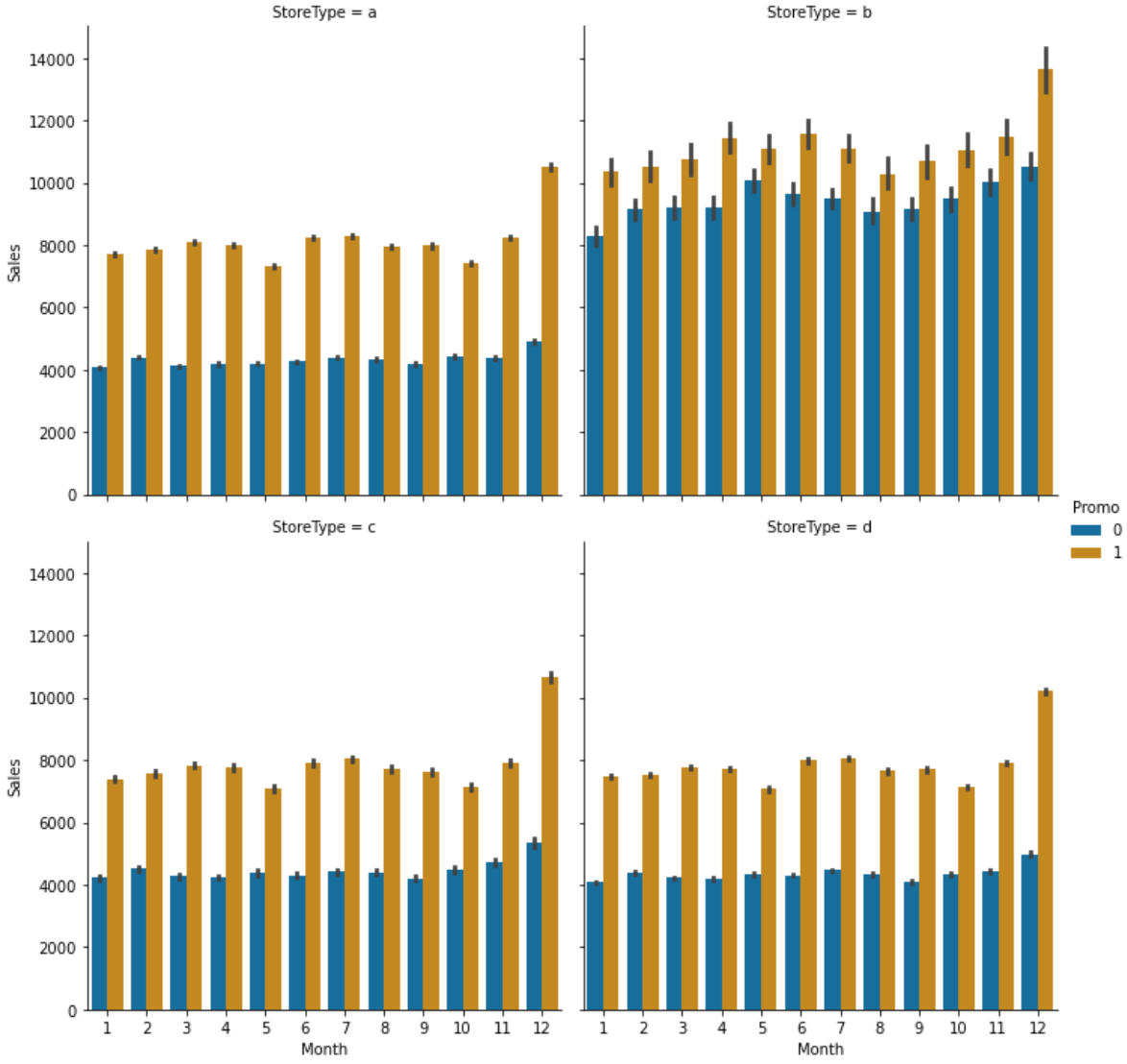
How well does each store do during promotions

In [56]:

```
#since the years have similar trends of sales during promotions, it won't be grouped by yea
sns.catplot(data=train_store, x='Month', y='Sales', col='StoreType', col_order=['a','b','c'
            hue='Promo', palette='colorblind', col_wrap=2, kind='bar')
```

Out[56]:

```
<seaborn.axisgrid.FacetGrid at 0x1f88d2ef4f0>
```



All stores experienced the highest sales with promotions in December

What is the highest amount of Sales made in December by all Store Types?

In [57]:

```
train_store_dec = train_store[train_store.Month == 12]
```

In [58]:

```
train_store_dec.groupby(['StoreType', 'Year', 'Promo']).Sales.sum()
```

Out[58]:

```
StoreType  Year  Promo
a          2013  0        59748149
                 1        63960807
           2014  0        55854415
                 1        53700950
b          2013  0         3627130
                 1         2269047
           2014  0         3669297
                 1         2233576
c          2013  0        15751777
                 1        15859019
           2014  0        15878420
                 1        14231052
d          2013  0        34821543
                 1        35673089
           2014  0        29419222
                 1        27133660
Name: Sales, dtype: int64
```

Store A sold the highest in 2013 in December with a count of 63960807 with promotions Store B sold the least in 2014 in December with a count of 2233576 with promotions.

In [ ]:

```

```