NextHikes

Project: Rossman Pharmaceutical Sales Prediction across multiple Stores

SUBMITTED BY: HARSHAD MAURYA

Introduction

- Predicting sales performance is one of the key challenges every business face.
- It is important for firms to predict customer demands to offer the right product at the right time and at the right place.
- The importance of this issue is underlined by the fact that figuratively a bazillion consulting firms are on the market trying to offer sales forecasting services to businesses of all sizes.

Project Need

- Rossmann operates over 3,000 drug stores in 7 European countries.
- Currently, Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance.
- Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality.
- With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied.

Datasets and features

- **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 (this is what you are predicting)
- **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.

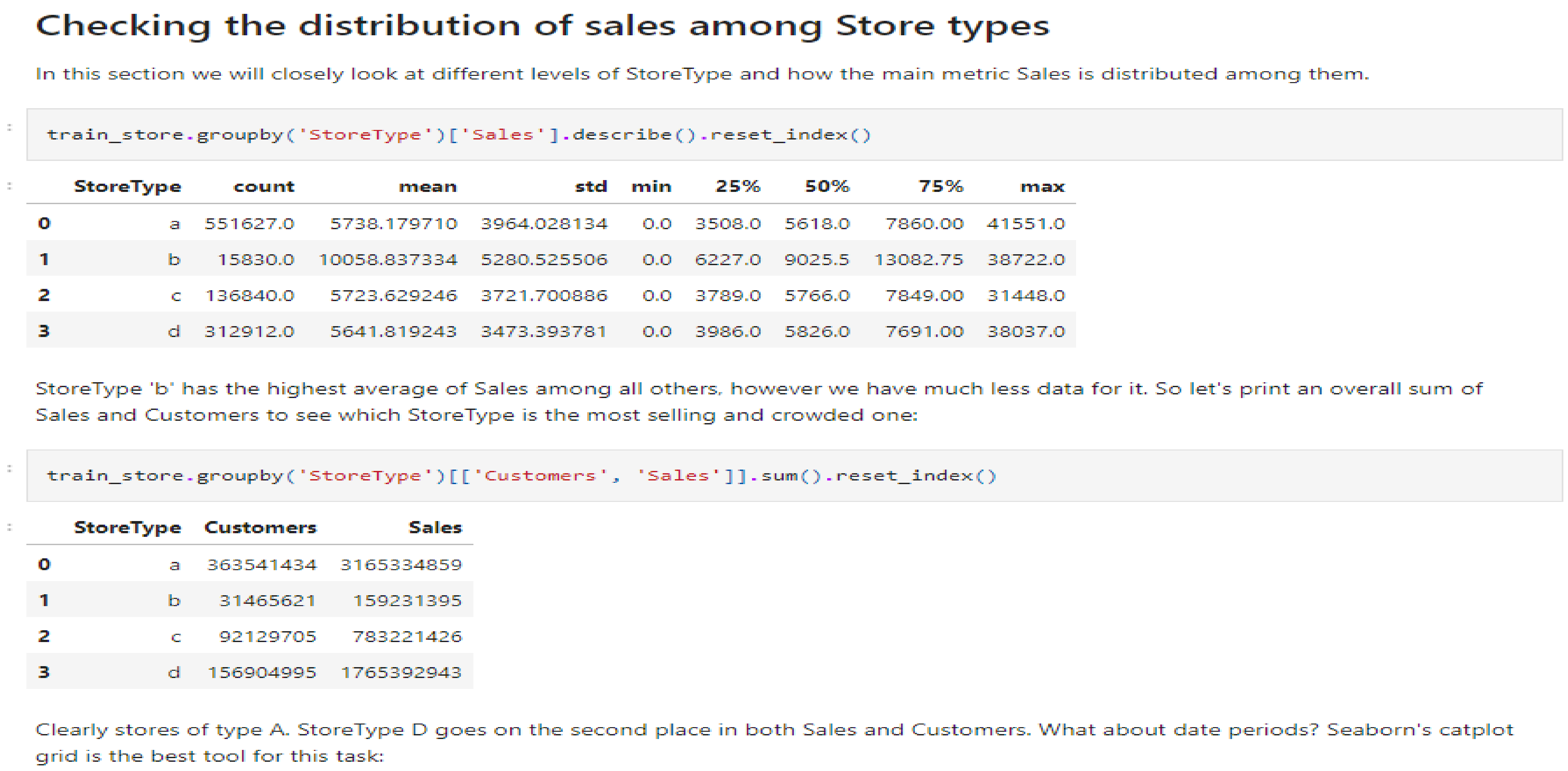
Datasets and features

- **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

Exploration of Customer Purchasing Behaviour=>



Checking the distribution of sales among Store types

In this section we will closely look at different levels of StoreType and how the main metric Sales is distributed among them.

```
train_store.groupby('StoreType')['Sales'].describe().reset_index()
```

	StoreType	count	mean	std	min	25%	50%	75%	max
0	a	551627.0	5738.179710	3964.028134	0.0	3508.0	5618.0	7860.00	41551.0
1	b	15830.0	10058.837334	5280.525506	0.0	6227.0	9025.5	13082.75	38722.0
2	c	136840.0	5723.629246	3721.700886	0.0	3789.0	5766.0	7849.00	31448.0
3	d	312912.0	5641.819243	3473.393781	0.0	3986.0	5826.0	7691.00	38037.0

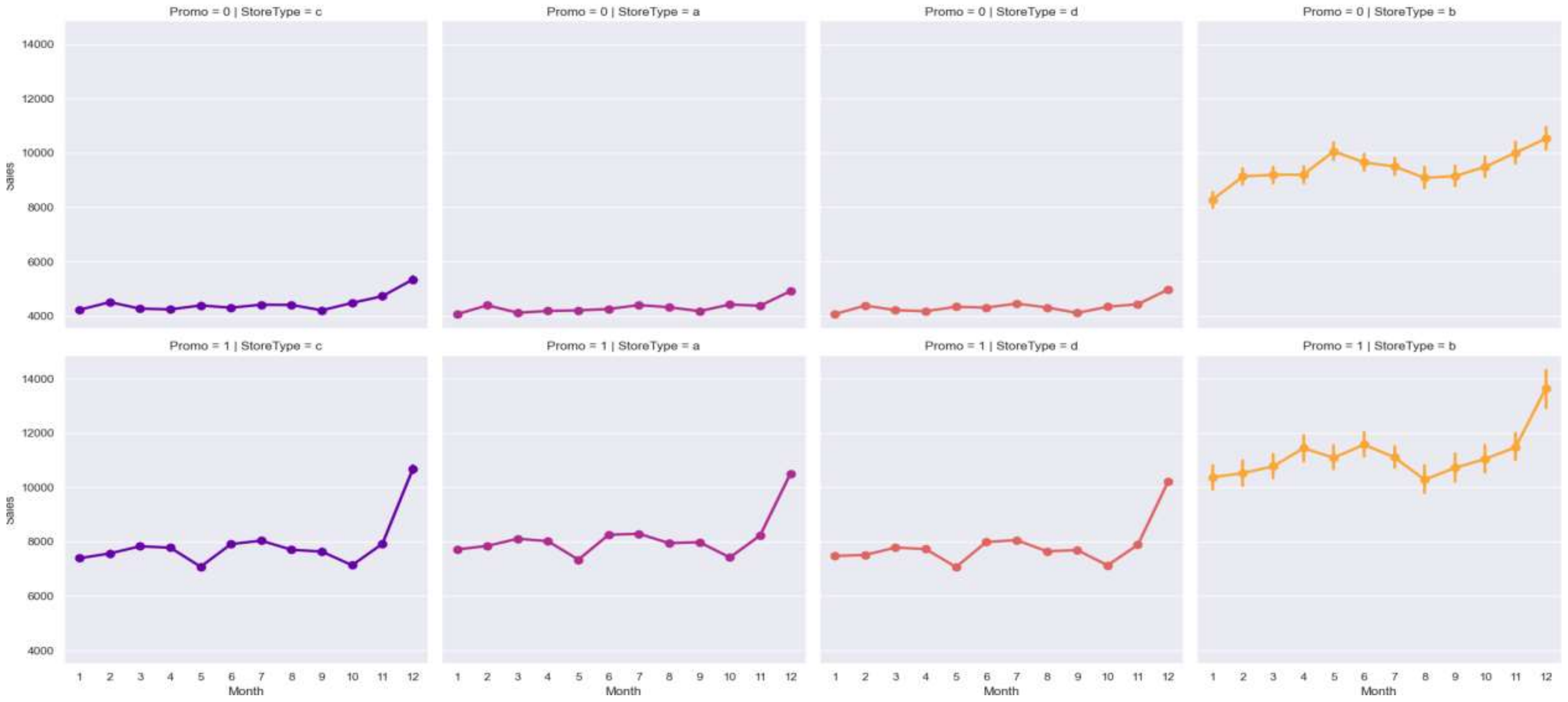
StoreType 'b' has the highest average of Sales among all others, however we have much less data for it. So let's print an overall sum of Sales and Customers to see which StoreType is the most selling and crowded one:

```
train_store.groupby('StoreType')[['Customers', 'Sales']].sum().reset_index()
```

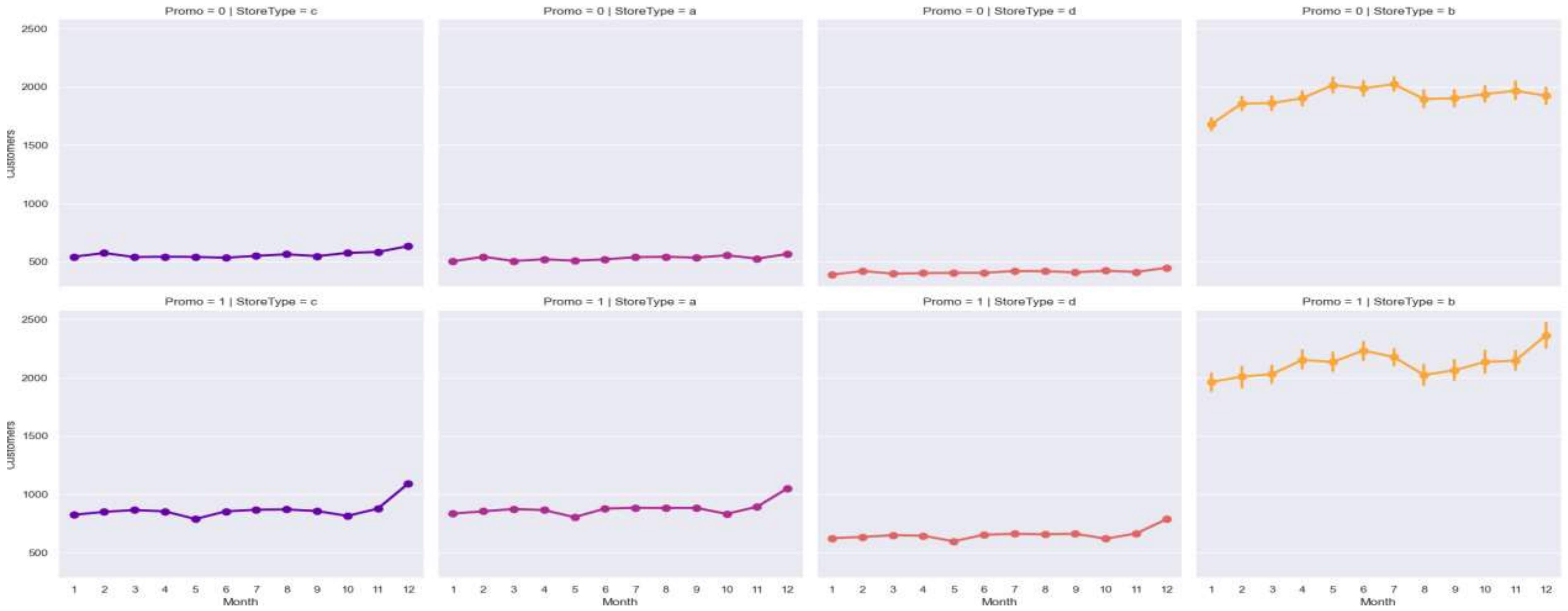
	StoreType	Customers	Sales
0	a	363541434	3165334859
1	b	31465621	159231395
2	c	92129705	783221426
3	d	156904995	1765392943

Clearly stores of type A. StoreType D goes on the second place in both Sales and Customers. What about date periods? Seaborn's catplot grid is the best tool for this task:

Sales Trends



Customer Trends



All store types follow the same trend but at different scales depending on the presence of the (first) promotion Promo and StoreType itself (case for B).

To complete our preliminary data analysis, we can add variables describing the period of time during which competition and promotion were opened:

[38]:
```python
## competition open time (in months)
train_store['CompetitionOpen'] = 12 * (train_store.Year - train_store.CompetitionOpenSinceYear) + \
        (train_store.Month - train_store.CompetitionOpenSinceMonth)

## Promo open time
train_store['PromoOpen'] = 12 * (train_store.Year - train_store.Promo2SinceYear) + \
        (train_store.WeekOfYear - train_store.Promo2SinceWeek) / 4.0

## replace NA's by 0
train_store.fillna(0, inplace = True)

## average PromoOpen time and CompetitionOpen time per store type
train_store.loc[:, ['StoreType', 'Sales', 'Customers',
                    'PromoOpen', 'CompetitionOpen']].groupby('StoreType').mean().reset_index()
```

[38]:

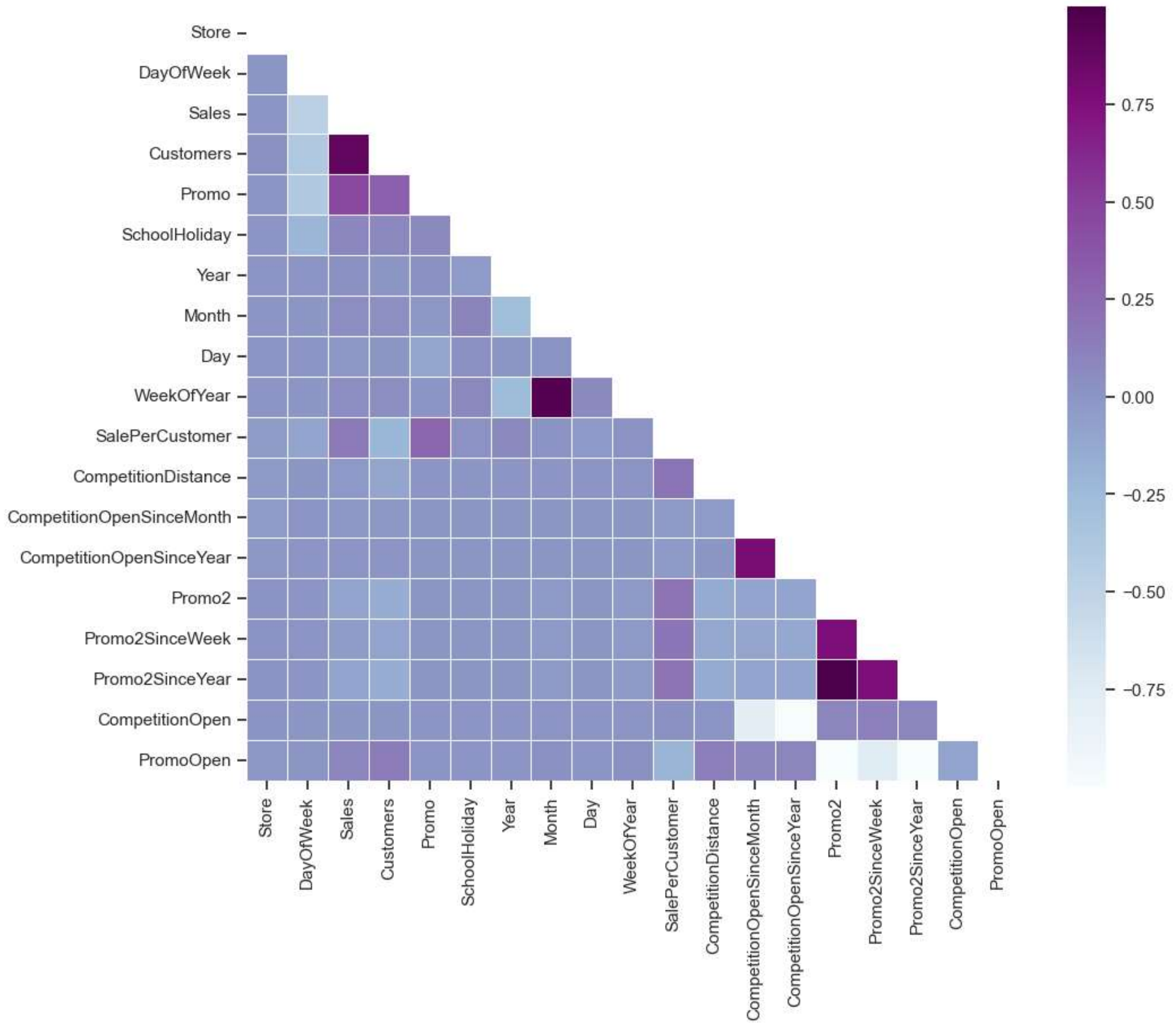
	StoreType	Sales	Customers	PromoOpen	CompetitionOpen
0	a	5738.179710	659.034880	12882.592255	7122.919204
1	b	10058.837334	1987.720846	17264.621605	11264.823310
2	c	5723.629246	673.265894	12128.625157	6737.717159
3	d	5641.819243	501.434892	10397.346995	9038.541369

The most selling and crowded StoreType A doesn't appear to be the one the most exposed to competitors. Instead it's a StoreType B, which also has the longest running period of promotion.

Correlational Analysis

- As mentioned before, we have a strong positive correlation between the amount of Sales and Customers of a store. We can also observe a positive correlation between the fact that the store had a running promotion (Promo equal to 1) and amount of Customers.

- However, as soon as the store continues a consecutive promotion (Promo2 equal to 1) the number of Customers and Sales seems to stay the same or even decrease, which is described by the pale negative correlation on the heatmap. The same negative correlation is observed between the presence of the promotion in the store and the day of a week.

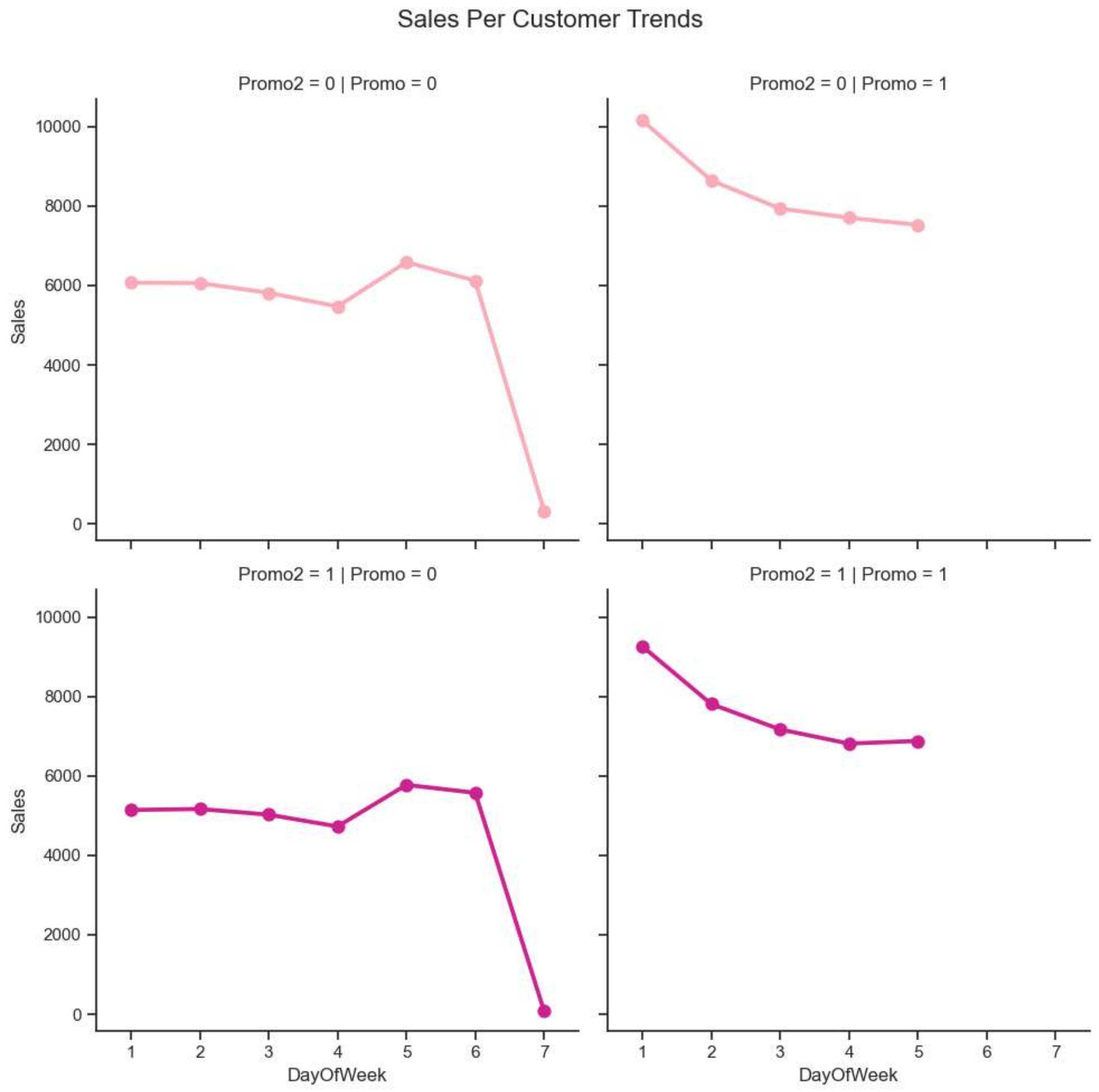


There are several things here:

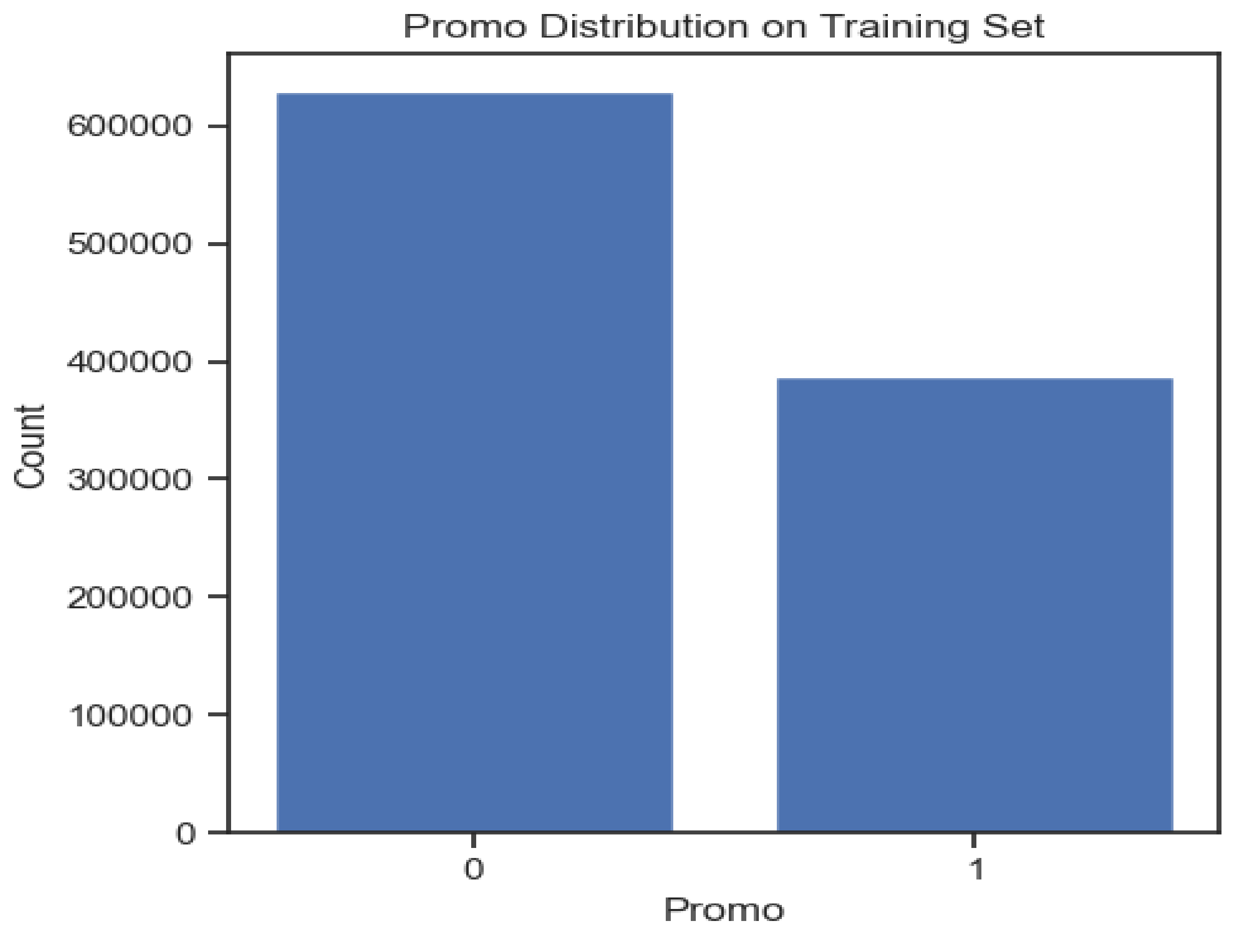
- In case of no promotion, both Promo and Promo2 are equal to 0, Sales tend to peak on Sunday (!). Though we should note that StoreType C doesn't work on Sundays. So it is mainly data from StoreType A, B and D.

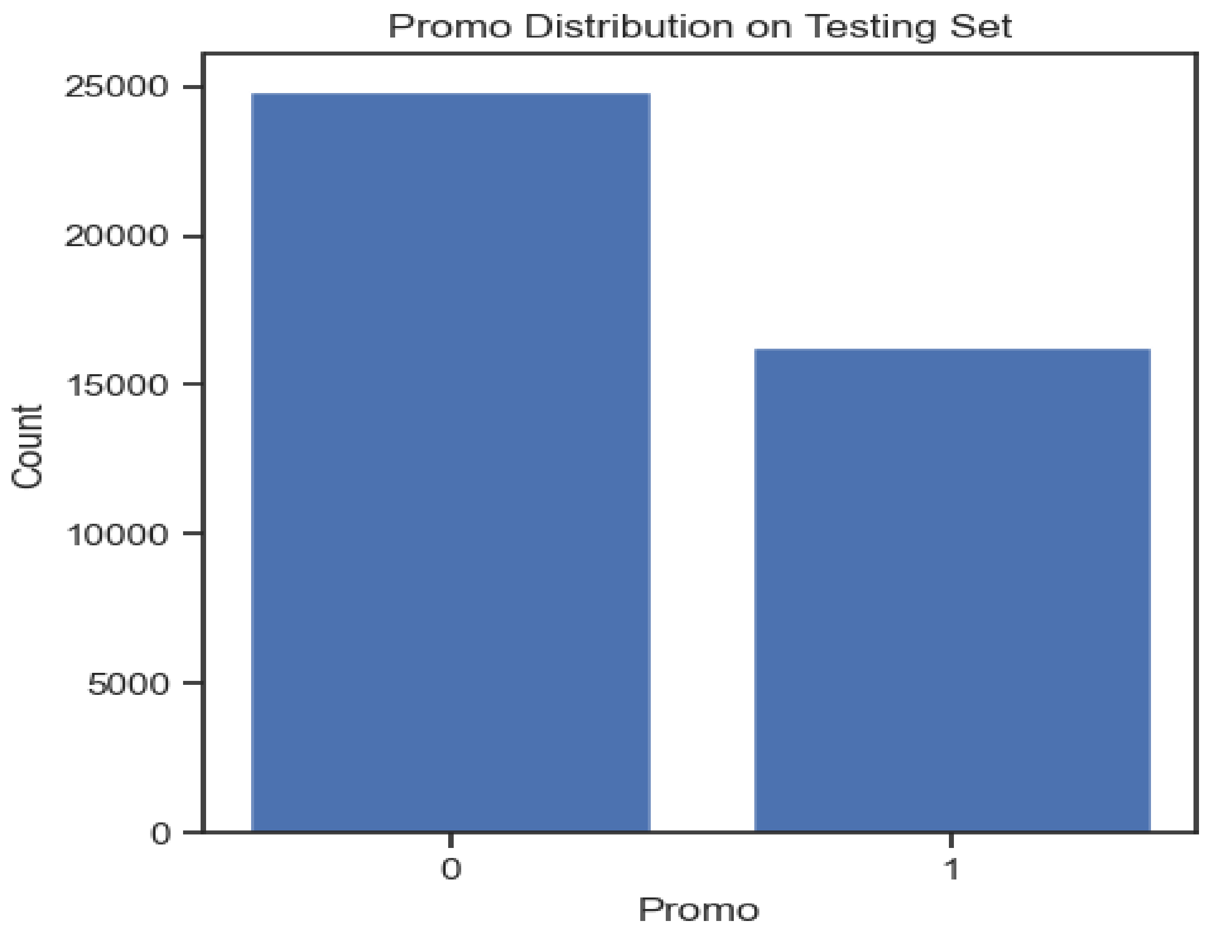
- On the contrary, stores that run the promotion tend to make most of the Sales on Monday. This fact could be a good indicator for Rossmann marketing campaigns. The same trend follow the stores which have both promotion at the same time (Promo and Promo2 are equal to 1).

- Promo2 alone doesn't seem to be correlated to any significant change in the Sales amount. This can be also prooved by the blue pale area on the heatmap above.

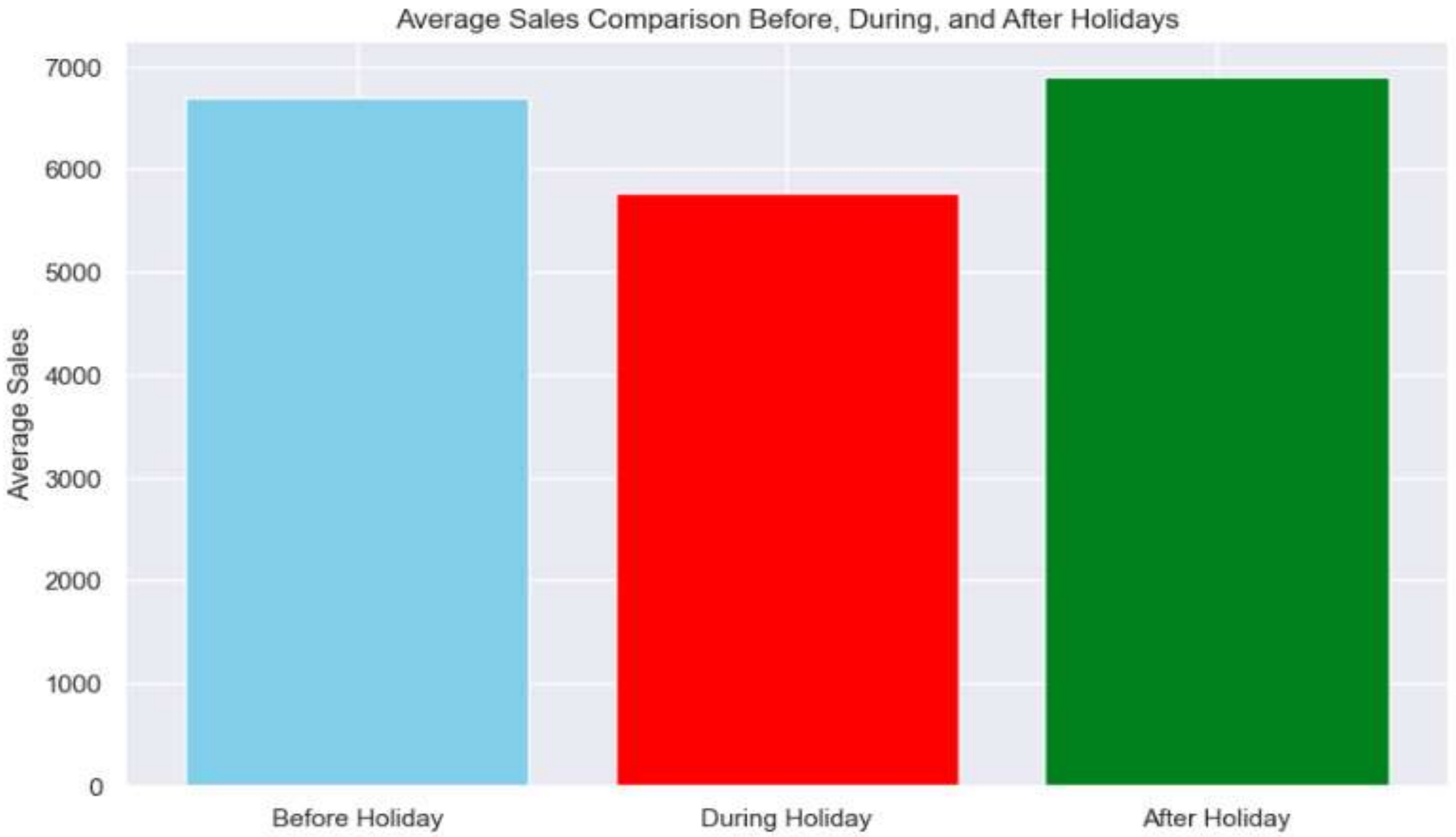


Check for distribution in both training and test sets - are the promotions distributed similarly between these two groups?

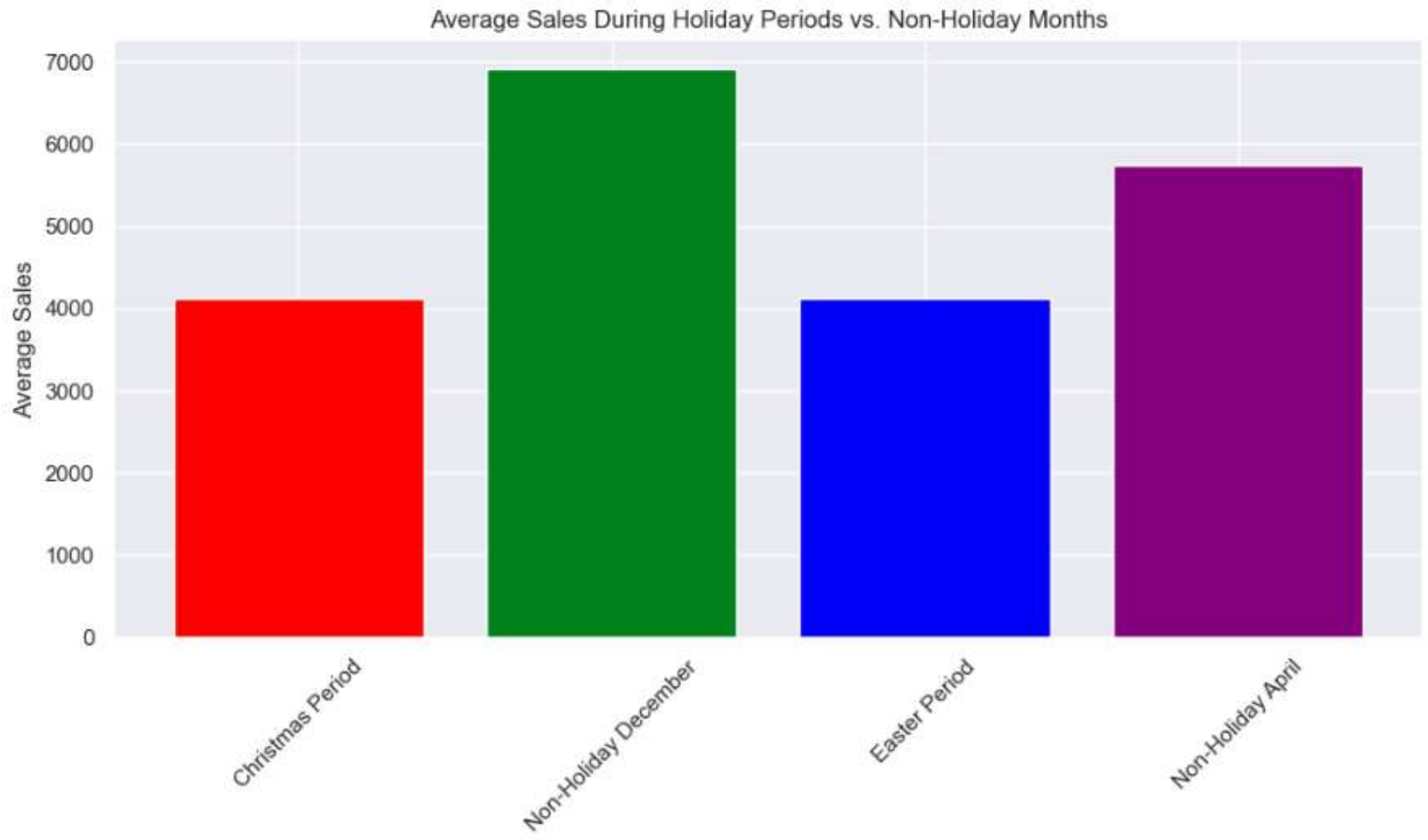




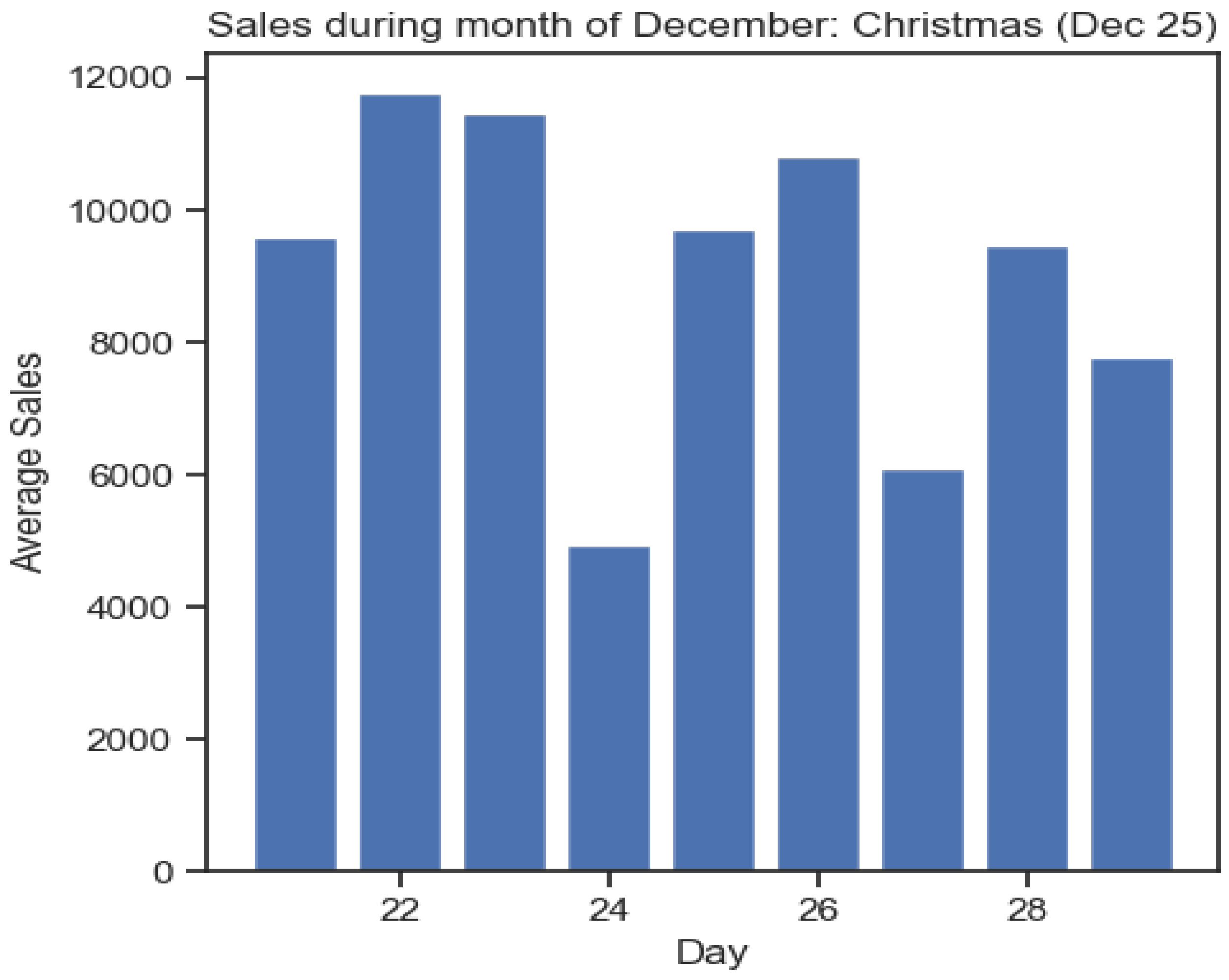
Observation: The distributiuon of promo over on training set is greater than the promo distribution over test set.

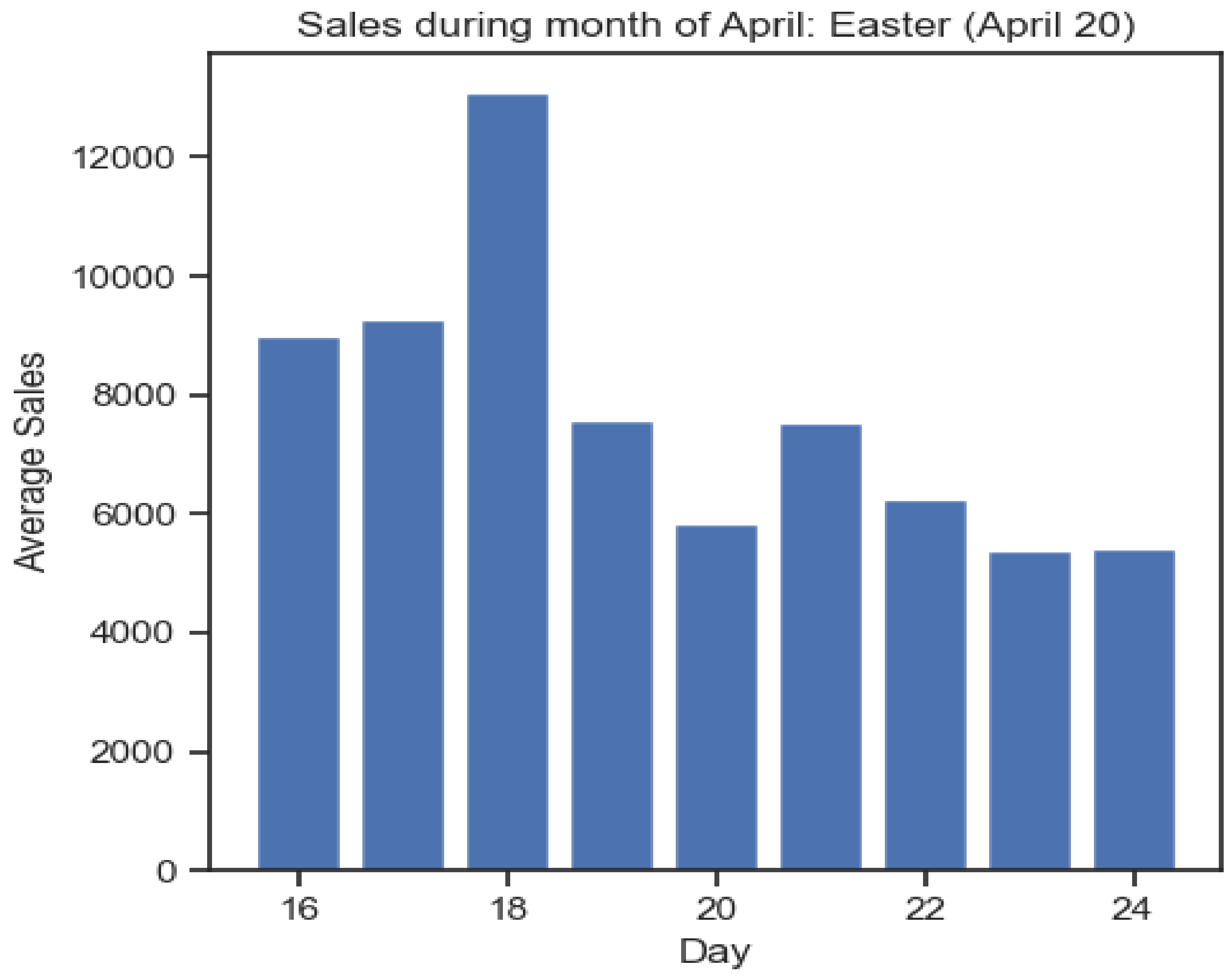


Observation: there are more sales before and after the Holidays when it compared to holidays sales


Average Sales During Holiday Periods vs. Non-Holiday Months
Average Sales
7000
6000
5000
4000
3000
2000
1000
0
Christmas Period
Non-Holiday December
Easter Period
Non-Holiday April

Check & compare sales behavior before, during and after holidays

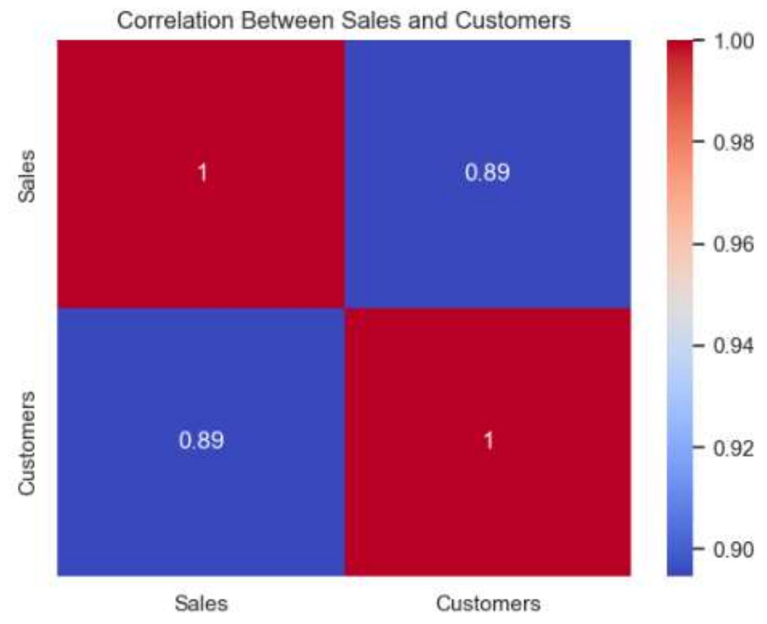




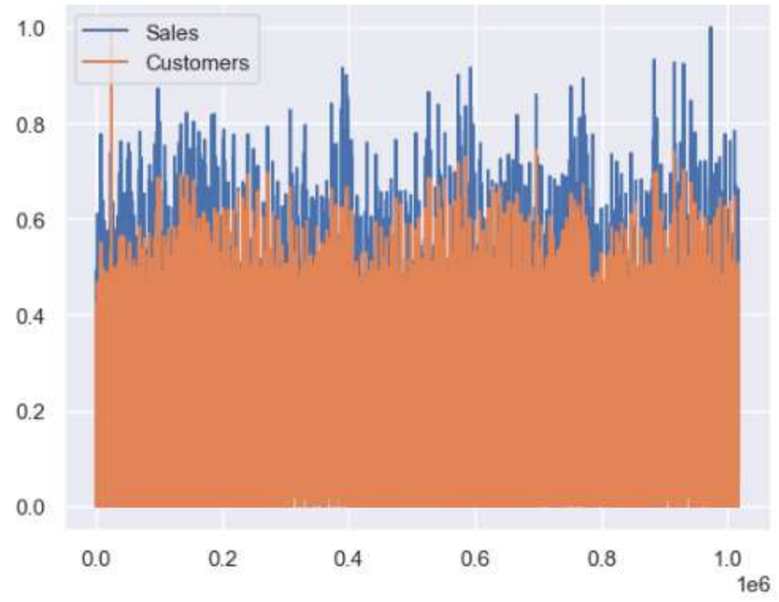
Observation: There are more sales before and after the Holidays when it compared to holidays sales

- What can you say about the correlation between sales and number of customers?

Observation: the more customers the more sales. because there is a high correlation between Sales and Customers.

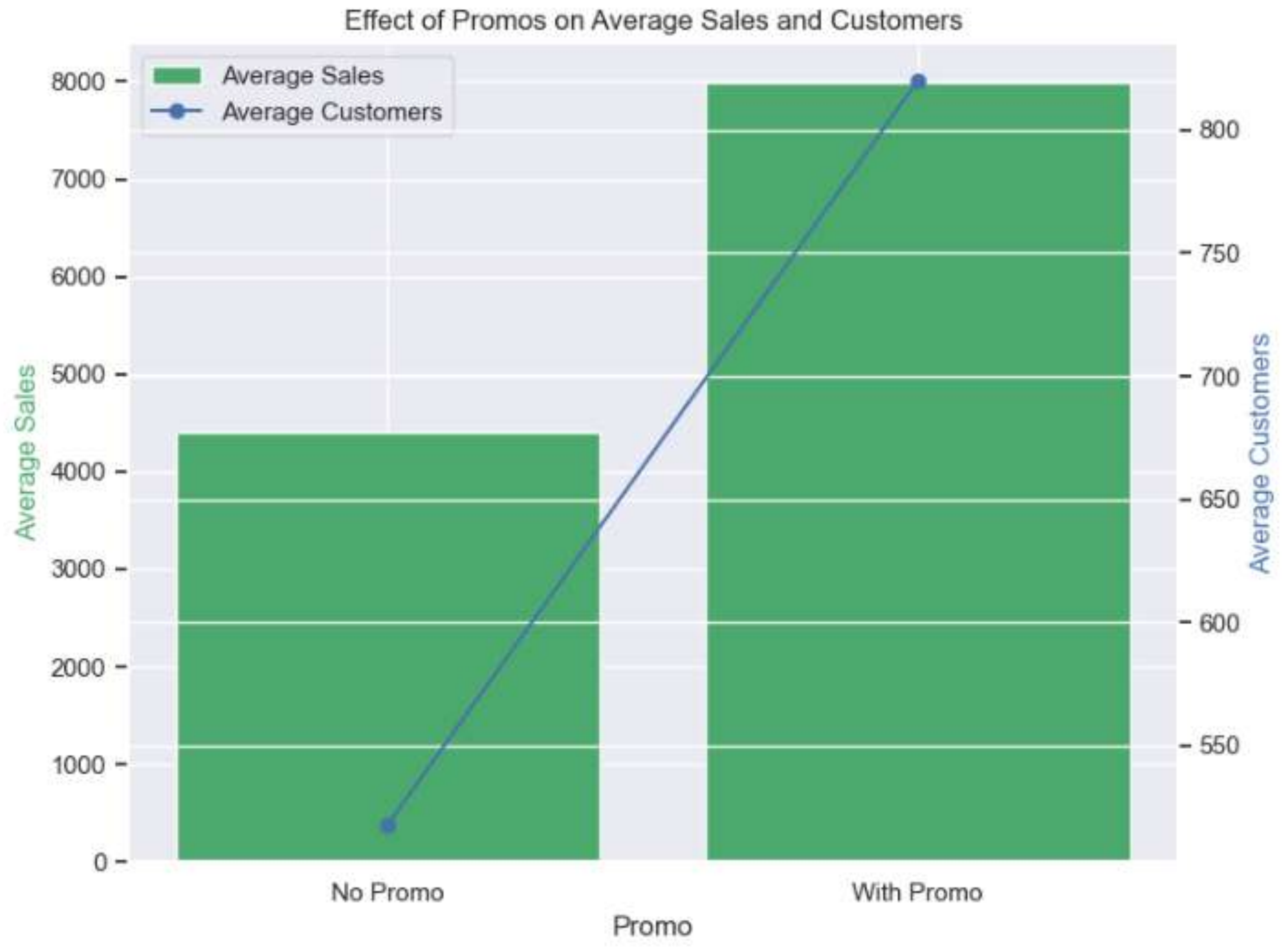


```
<Axes: >
```



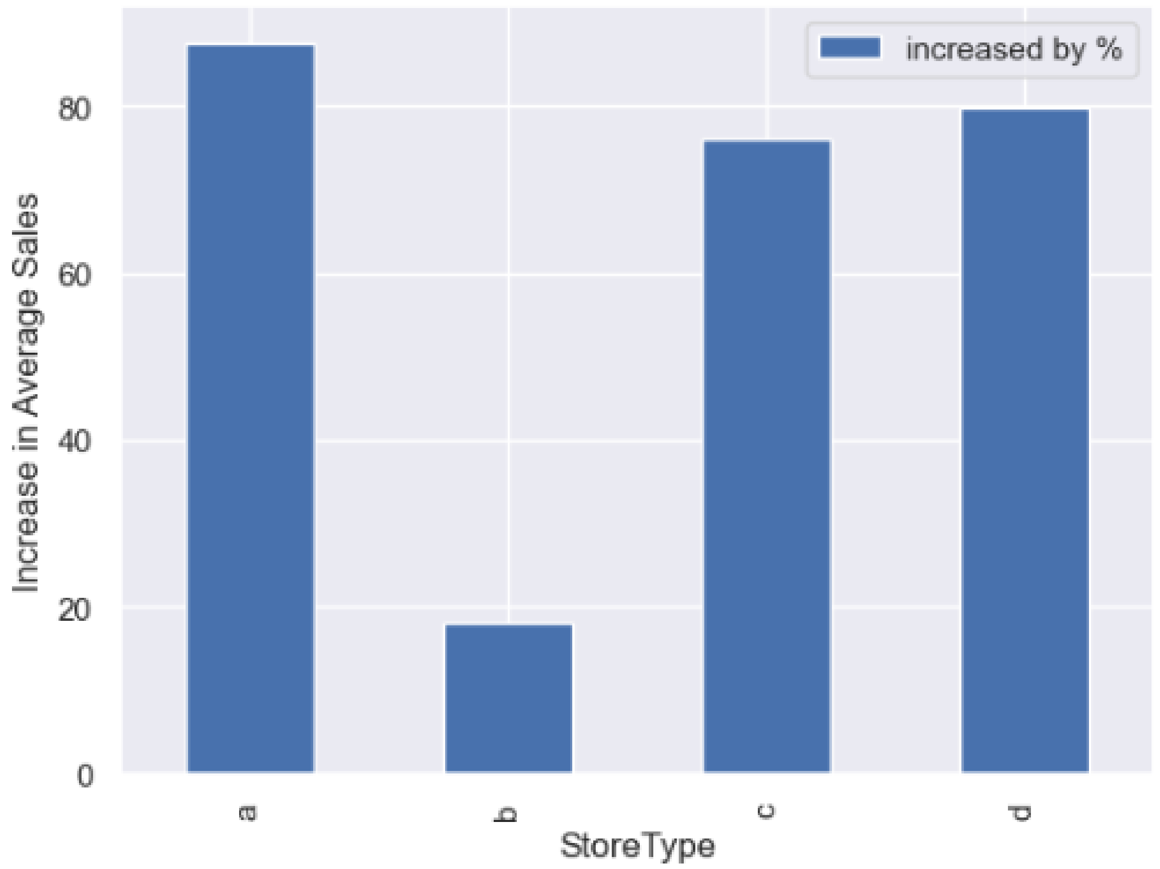
Observation: Sales and Number of Customers are directly proportional as seen in the figure above

How does promo affect sales? Are the promos attracting more customers? How does it affect already existing customers?



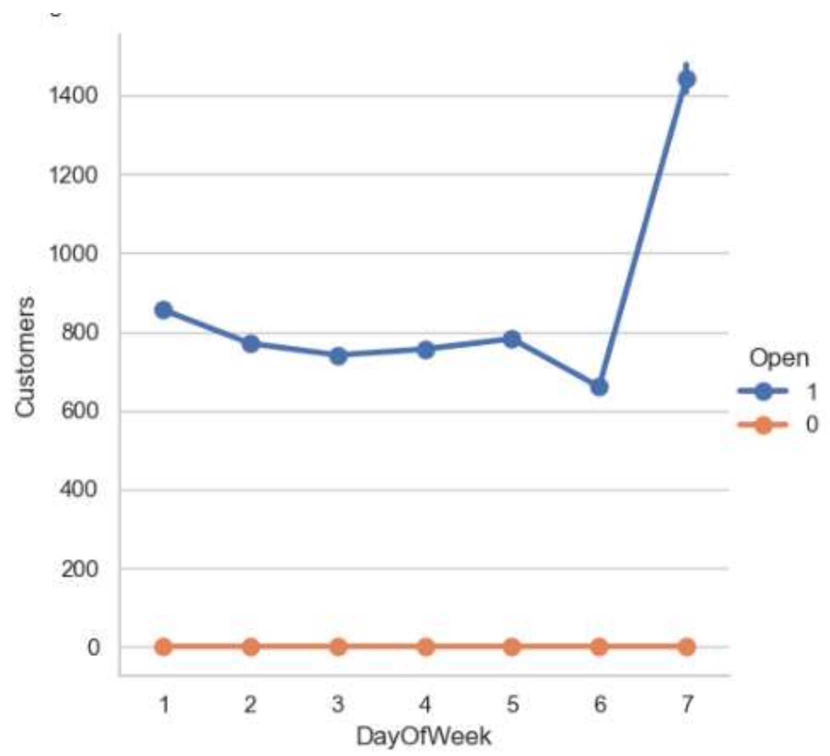
Observation: Working on promotion increase the customers base and Increase Sales.

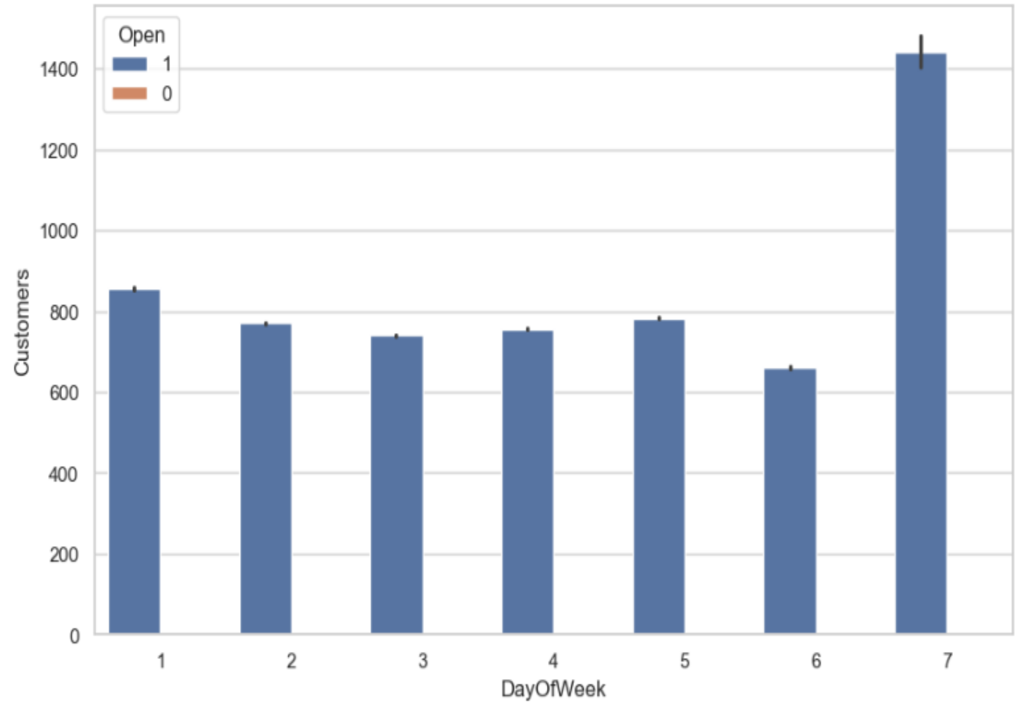
Could the promos be deployed in more effective ways? Which stores should promos be deployed in?



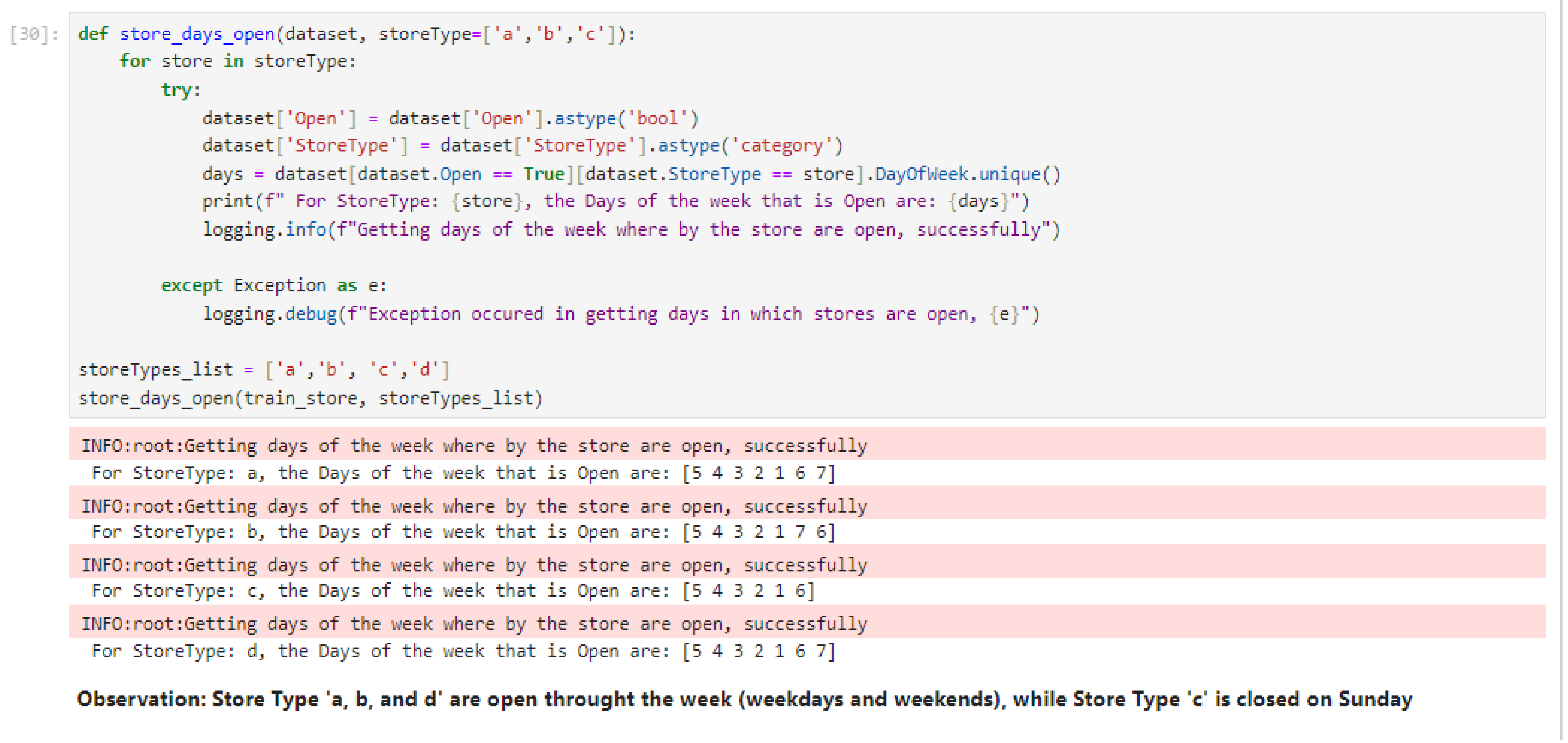
Observation: We have a more similar trends regarding the three stores namely 'a', 'd', and 'c' However Promotion seems to have a bigger impact in StoreType 'a'. We can see there is more than 80% growth in sales due to promotion.

Trends of customer behavior during store open and closing times





Which stores are opened on all weekdays? How does that affect their sales on weekends?



[30]:

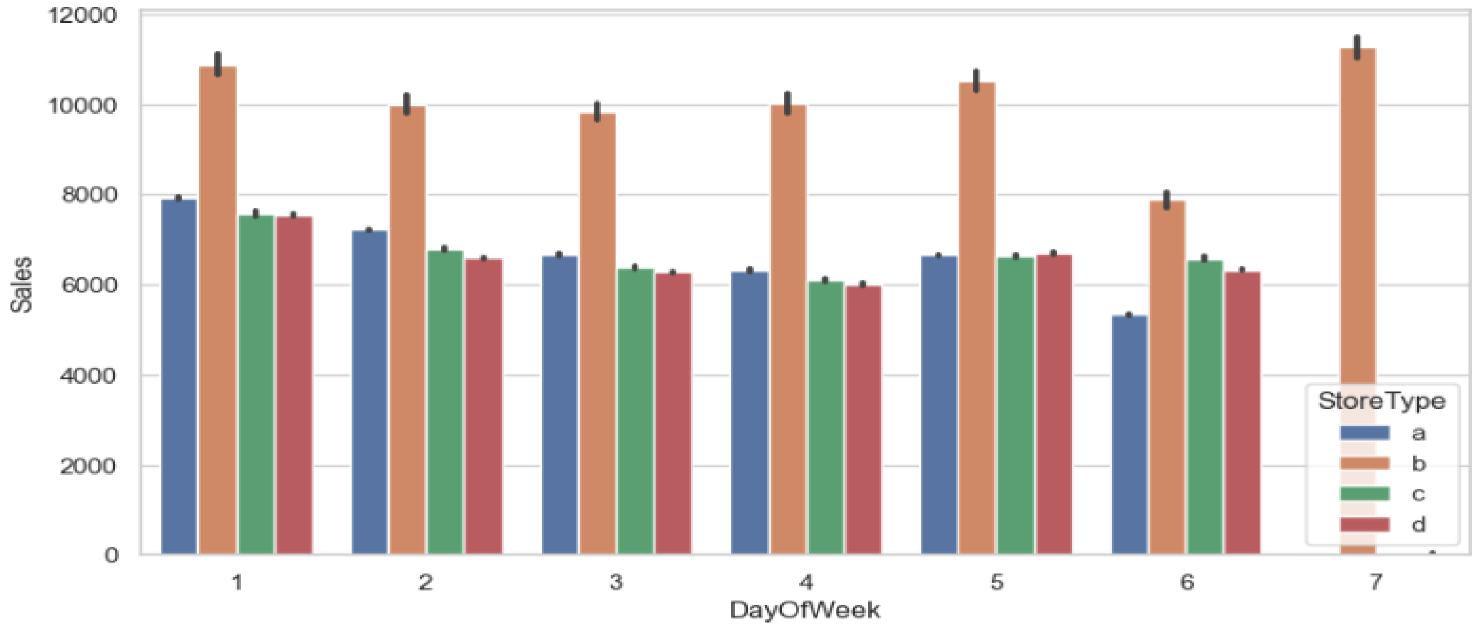
```python
def store_days_open(dataset, storeType=['a','b','c']):
    for store in storeType:
        try:
            dataset['Open'] = dataset['Open'].astype('bool')
            dataset['StoreType'] = dataset['StoreType'].astype('category')
            days = dataset[dataset.Open == True][dataset.StoreType == store].DayOfWeek.unique()
            print(f" For StoreType: {store}, the Days of the week that is Open are: {days}")
            logging.info(f"Getting days of the week where by the store are open, successfully")

        except Exception as e:
            logging.debug(f"Exception occured in getting days in which stores are open, {e}")

storeTypes_list = ['a','b', 'c','d']
store_days_open(train_store, storeTypes_list)
```

```
INFO:root:Getting days of the week where by the store are open, successfully
 For StoreType: a, the Days of the week that is Open are: [5 4 3 2 1 6 7]
INFO:root:Getting days of the week where by the store are open, successfully
 For StoreType: b, the Days of the week that is Open are: [5 4 3 2 1 7 6]
INFO:root:Getting days of the week where by the store are open, successfully
 For StoreType: c, the Days of the week that is Open are: [5 4 3 2 1 6]
INFO:root:Getting days of the week where by the store are open, successfully
 For StoreType: d, the Days of the week that is Open are: [5 4 3 2 1 6 7]
```

Observation: Store Type 'a, b, and d' are open throught the week (weekdays and weekends), while Store Type 'c' is closed on Sunday

Store Type Sales with respect to Days of the week

```
[39]: plt.figure(figsize=(10,5))
      sns.set(style="whitegrid")
      sns.barplot(x='DayOfWeek',y='Sales',hue='StoreType',data=train_store)
      plt.show()
```



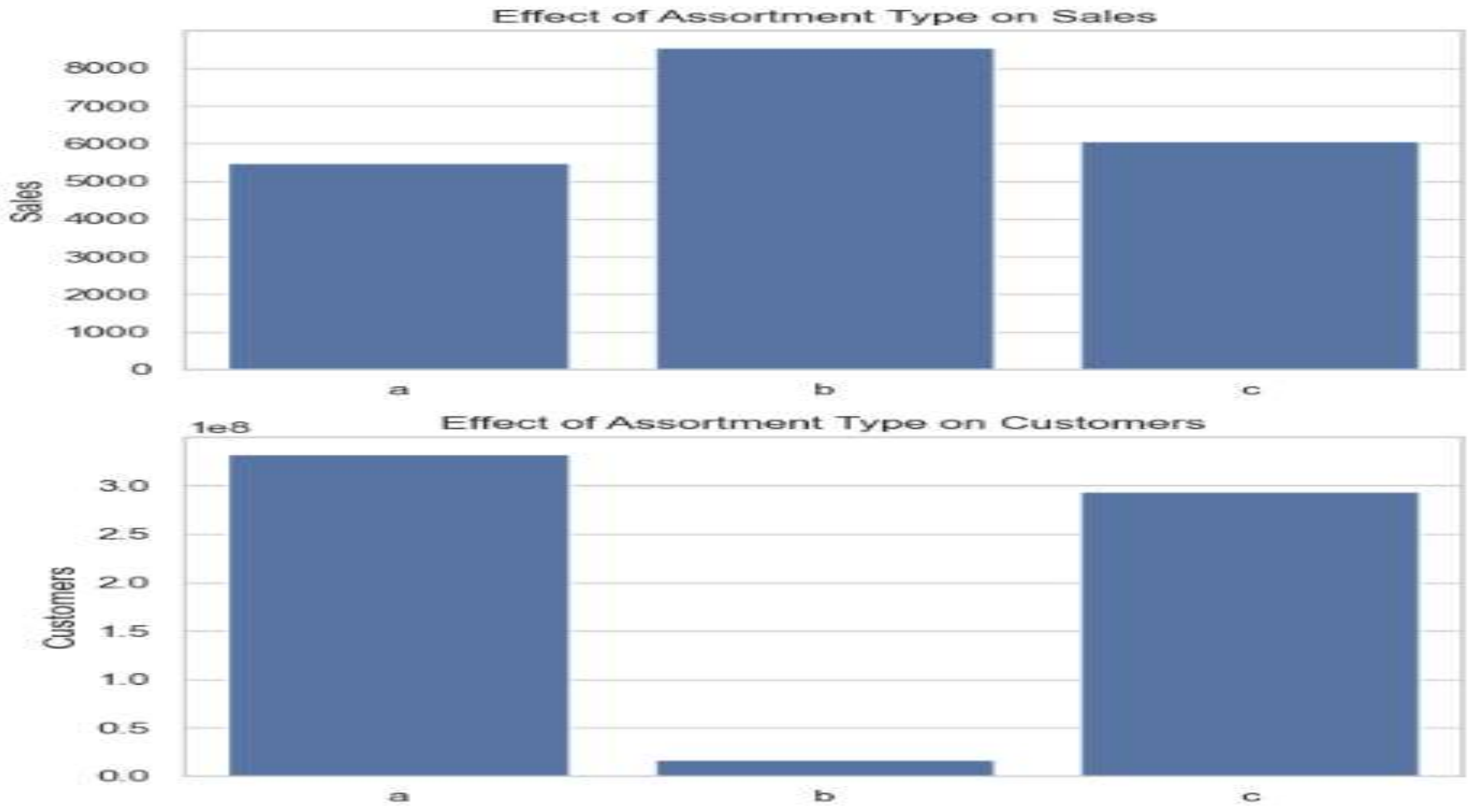
Store Types 'a' is Open on all days of the week, and has least sales on Saturday, and almost no sales on Sunday

Store Type 'b' is open on all days of the week, and has almost constant sales during weekdays and less sales on saturday, and highest sales on Sunday.

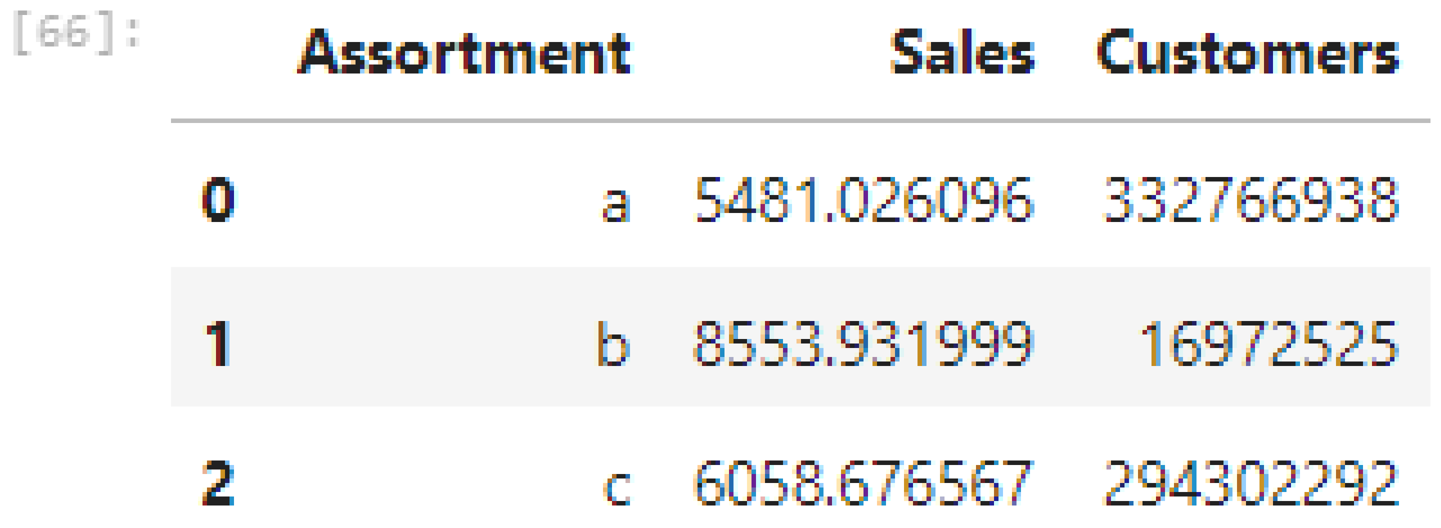
Store Type 'c' is open from Monday to Saturday, with sales almost constant and on Sunday is closed.

Store Type 'd' is open open on all days of the week, and has almost no sales on sunday.

Check how the assortment type affects sales



Check how the assortment type affects sales

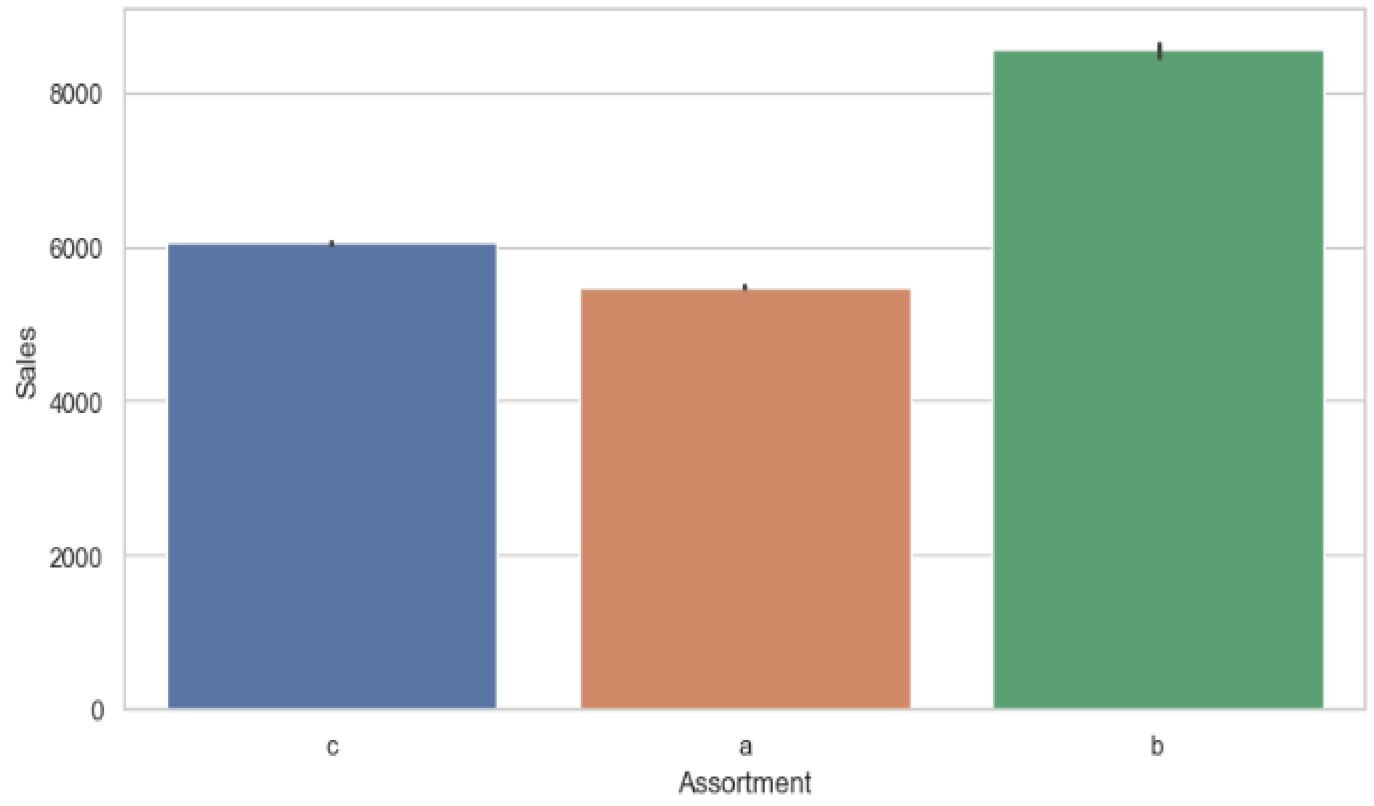


[66]:

	Assortment	Sales	Customers
0	a	5481.026096	332766938
1	b	8553.931999	16972525
2	c	6058.676567	294302292

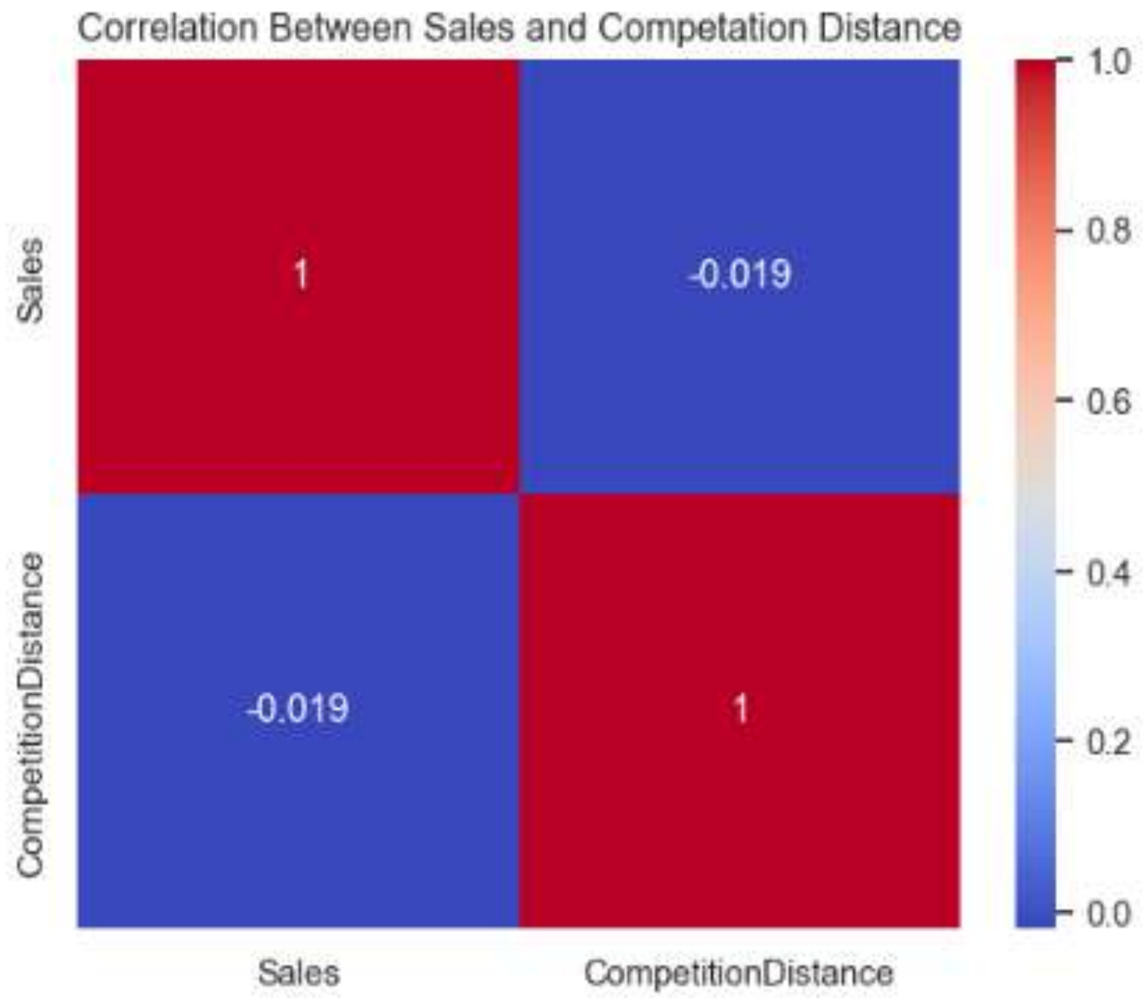
Observation:The stores with extra assortment type have high mean sales and lowest customer numbers.

```
Unique Assortment Types: ['c' 'a' 'b']
```



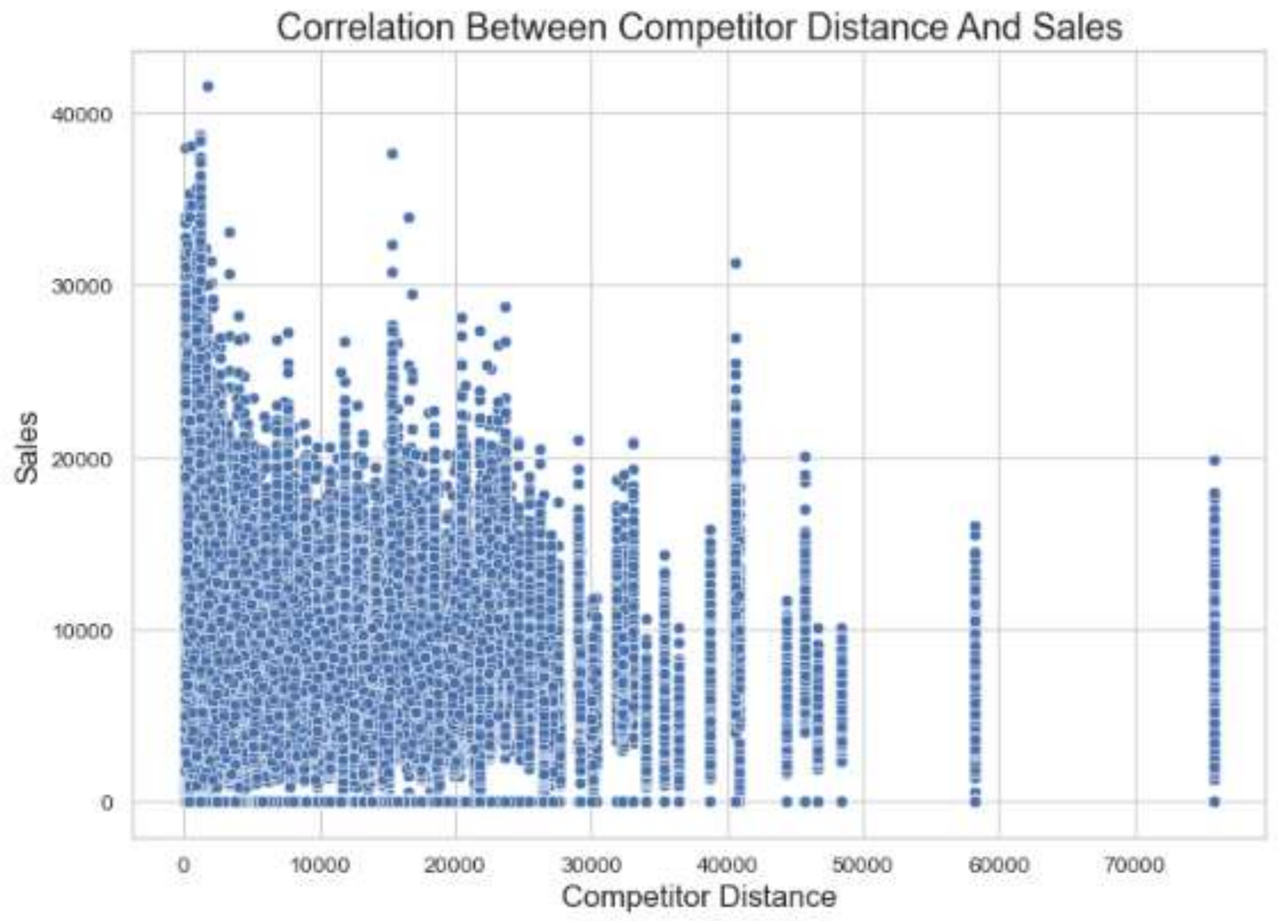
Observation: Sales is Highest in Extra assortment type followed by Extended type.

How does the distance to the next competitor affect sales? What if the store and its competitors all happen to be in city centers, does the distance matter in that case?



Observation: The more the closer the stores the higher the sales, and the more the distant the lower the sales.

How does the opening or reopening of new competitors affect stores? Check for stores with NA as competitor distance but later on has values for competitor distance?



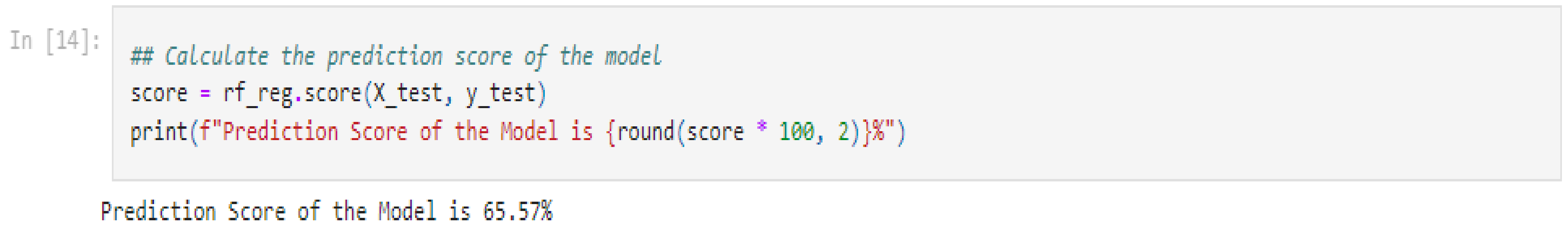
Conclusion of EDA

- The most selling and crowded StoreType is A.
- The best "Sale per Customer" StoreType D indicates to the higher Buyer Cart. To benefit from this fact, Rossmann can consider proposing bigger variety of its products.
- Low SalePerCustomer amount for StoreType B indicates to the possible fact that people shop there essentially for "small" things. Even though this StoreType generated the least amount of sales and customers over the whole period, it shows a great potential.
- Customers tends to buy more on Mondays when there's one promotion (Promo) and on Sundays when there's no promotion at all (both Promo and Promo1 are equal to 0).
- Promo2 alone doesn't seem to be correlated to any significant change in the Sales amount.

Prediction of store sales

- For PreProcessing here using LabelEncoder, MinMaxScaler, and RandomForestRegressor model.

- Prediction score of the model is 65..57%



```
In [14]:
## Calculate the prediction score of the model
score = rf_reg.score(X_test, y_test)
print(f"Prediction Score of the Model is {round(score * 100, 2)}%")
```

```
Prediction Score of the Model is 65.57%
```

Building models with sklearn pipelines

In [20]:

```
rfr_score = rfr.score(X_test, y_test)
lr_score = lr.score(X_test, y_test)
dt_score = dt.score(X_test, y_test)

print(f"Prediction Score of the RandomForestRegressiion is {round(rfr_score * 100, 2)}%")
print(f"Prediction Score of the LinearRegression is {round(lr_score * 100, 2)}%")
print(f"Prediction Score of the DecisionTree is {round(dt_score * 100, 2)}%")

# print(rfr_score)
```

```
Prediction Score of the RandomForestRegressiion is 64.64%
Prediction Score of the LinearRegression is 61.37%
Prediction Score of the DecisionTree is 53.97%
```

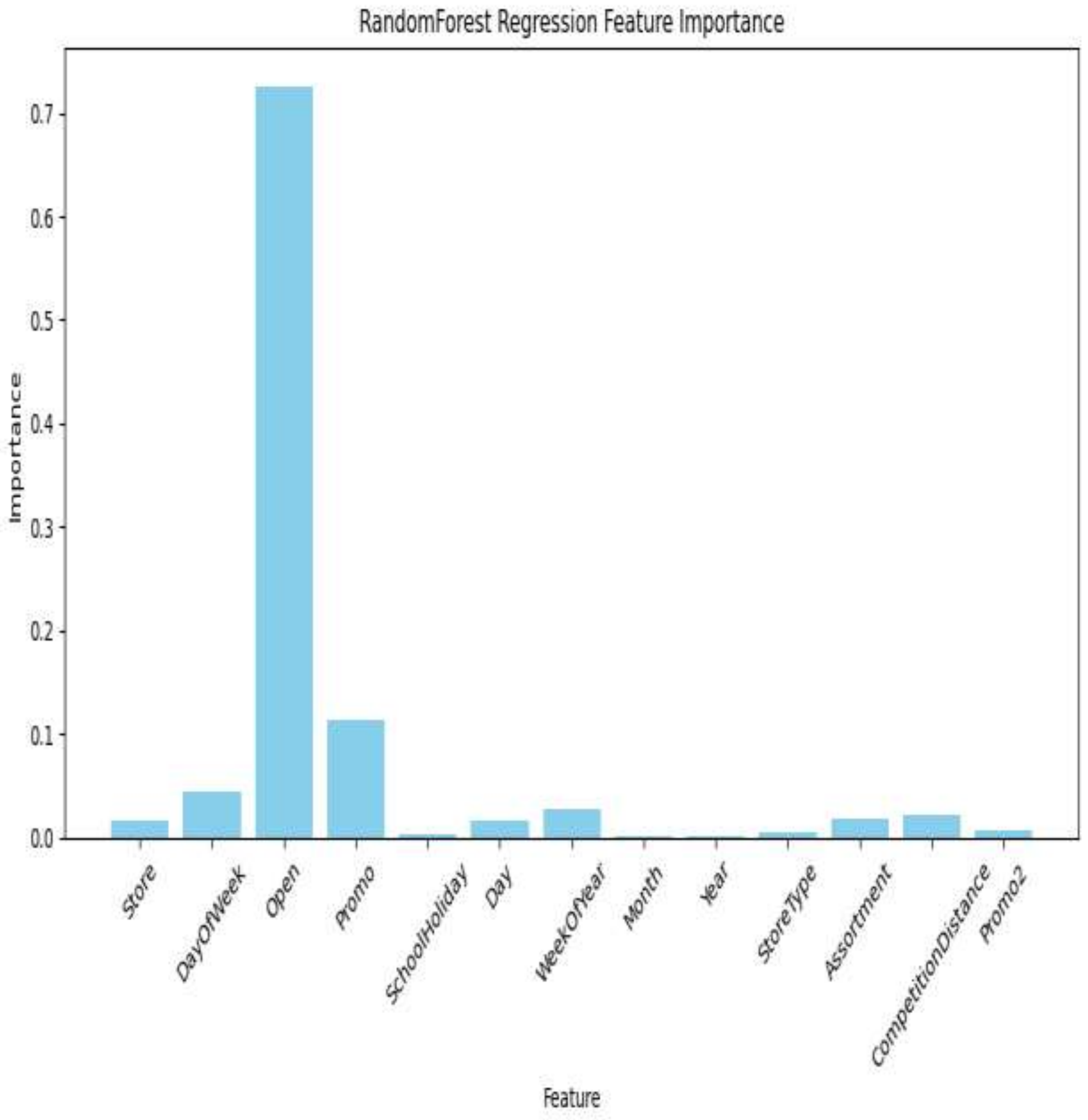
Choose a loss function

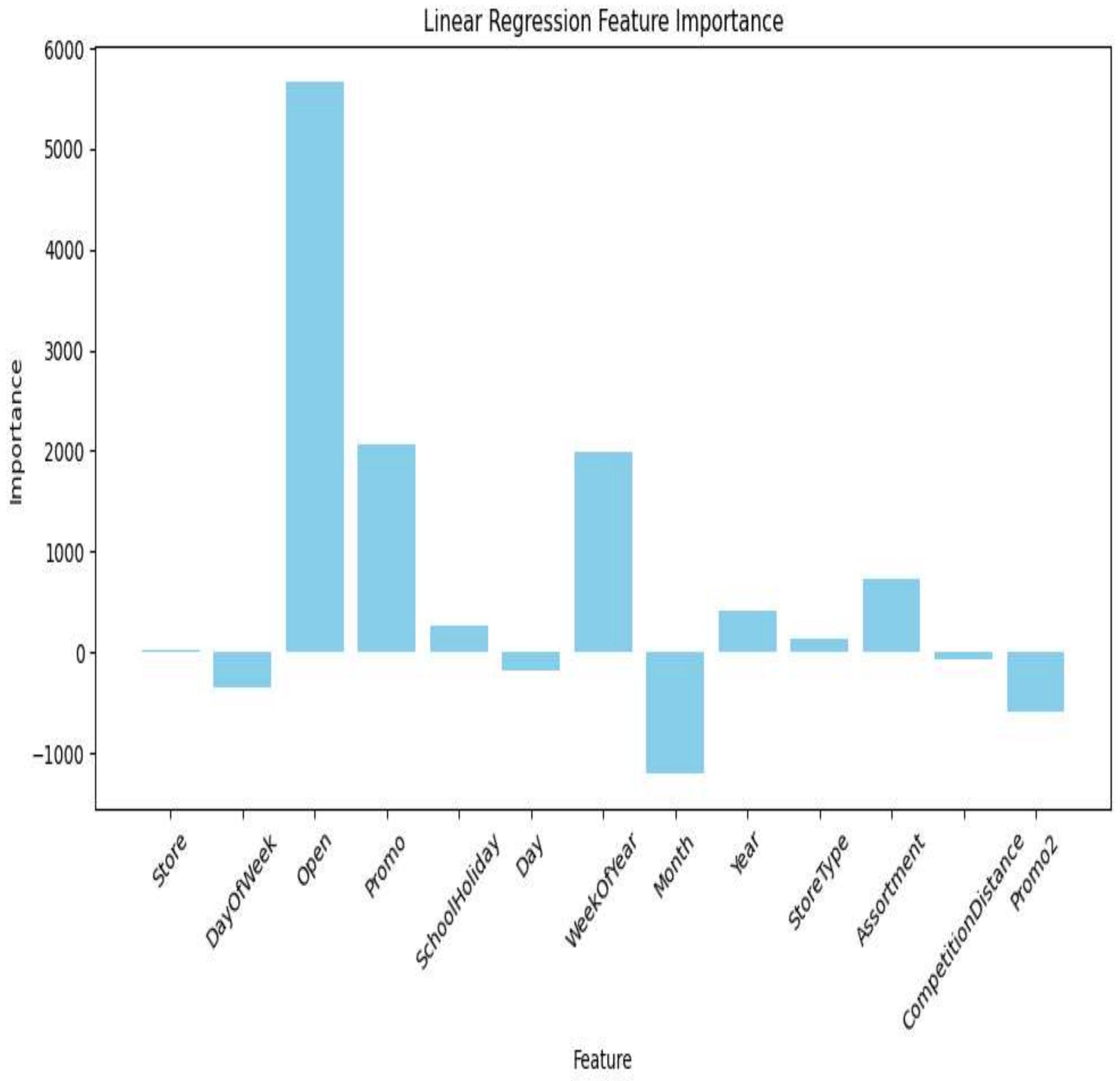
In [21]:

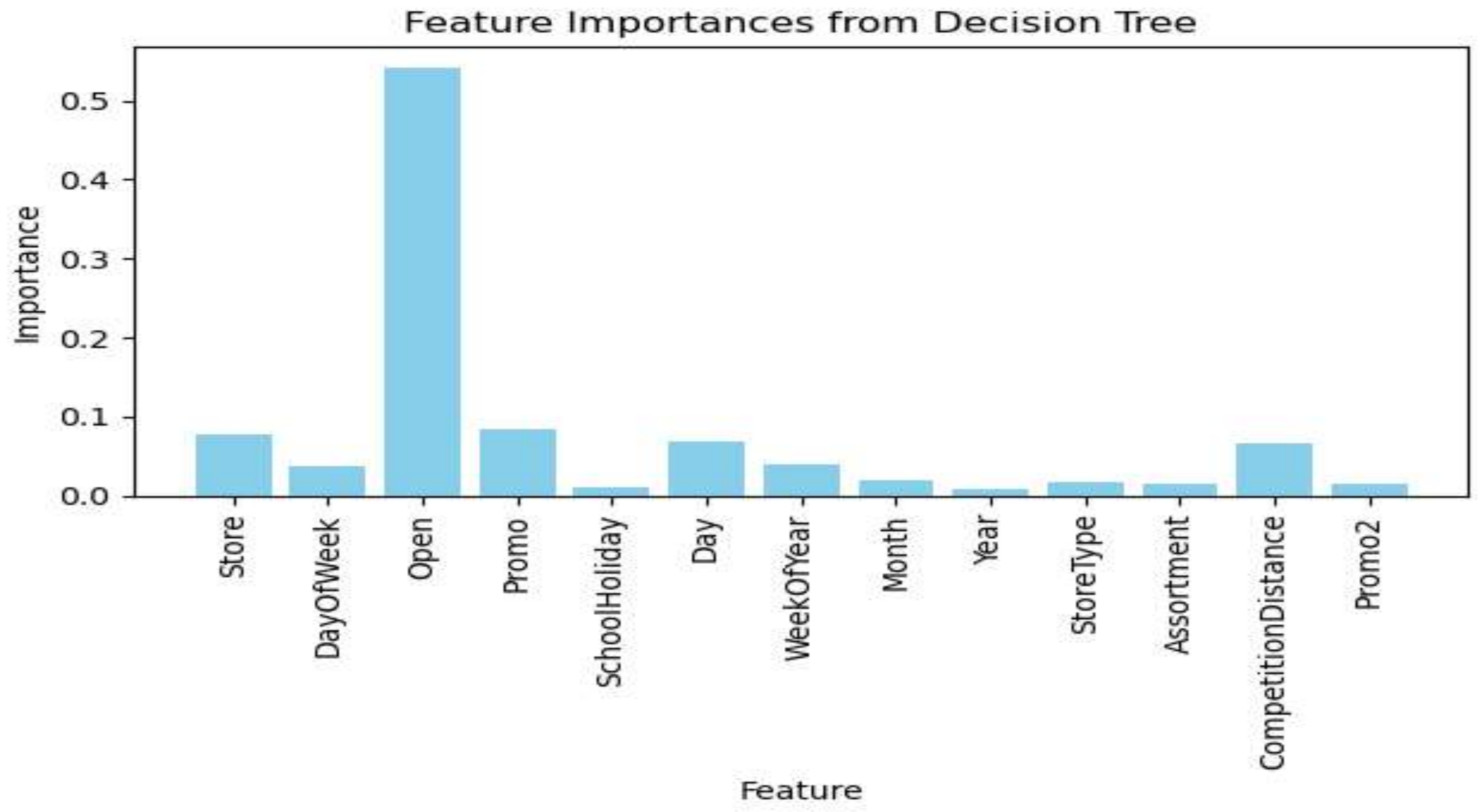
```
rfr_prediction = rfr.predict(X_test)
lr_prediction = lr.predict(X_test)
dt_prediction = dt.predict(X_test)

rfr_rmse = mean_squared_error(y_test, rfr_prediction, squared=False)
lr_rmse = mean_squared_error(y_test, lr_prediction, squared=False)
dt_rmse = mean_squared_error(y_test, dt_prediction, squared=False)

print(f"RandomForest Root Mean Square Loss : {rfr_rmse}")
print(f"Linear Root Mean Square Loss : {lr_rmse}")
print(f"DecisionTree Root Mean Square Loss : {dt_rmse}")
```

```
RandomForest Root Mean Square Loss : 2064.027930567851
Linear Root Mean Square Loss : 2157.445101928802
DecisionTree Root Mean Square Loss : 2355.054476015364
```

Post Prediction Analysis : Feature Importance



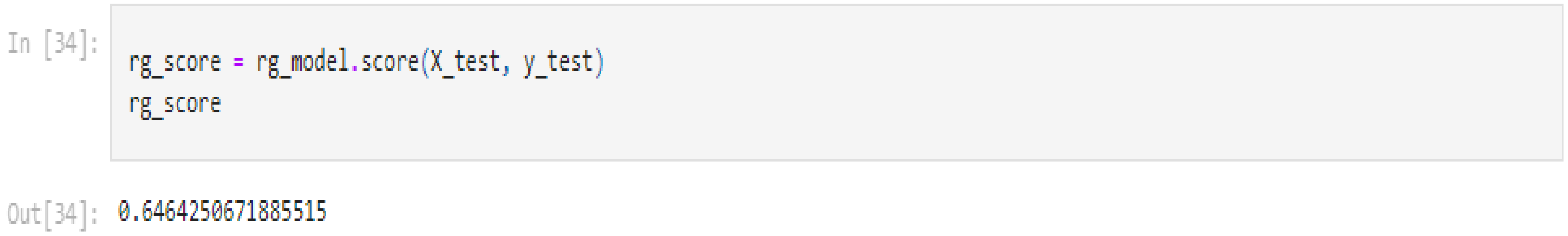



Feature Importances from Decision Tree
Importance
0.5
0.4
0.3
0.2
0.1
0.0
Store
DayOfWeek
Open
Promo
SchoolHoliday
Day
WeekOfYear
Month
Year
StoreType
Assortment
CompetitionDistance
Promo2
Feature

Serialize models

Read and save model -- serialization using pickle to dump mode

Regression Score:



```
In [34]:  rg_score = rg_model.score(X_test, y_test)
          rg_score

Out[34]:  0.6464250671885515
```

Thank you

Guided By : Desmond Onam Sir