ROSSMANN SALES PREDICTION

PREDICTING THE SALES OF ROSSMANN COMPANY - ONE OF THE LARGEST DRUG CHAIN STORES IN EUROPE

- RITIK PRAKASH NAYAK

A REGRESSION TASK

ROADMAP TO REGRESSION

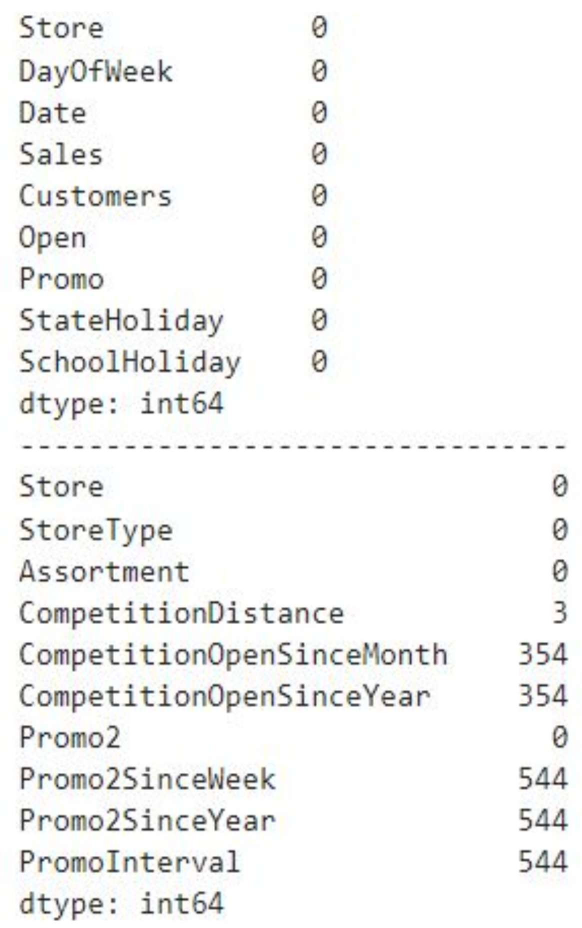
1. A BRIEF INTRODUCTION TO THE DATA AND THE PROBLEM STATEMENT
2. SALES DATA
   a. UNIVARIATE ANALYSIS
   b. BIVARIATE ANALYSIS
3. STORES DATA
   a. UNIVARIATE ANALYSIS
   b. DEALING WITH THE MISSING OBSERVATIONS
   c. FEATURE ENGINEERING
   d. MERGING THE DATASETS TOGETHER
4. MACHINE LEARNING
   a. LINEAR REGRESSION
   b. DECISION TREE REGRESSOR
   c. HYPERPARAMETER TUNING

“This use of the term “regression” is a historical accident; it is only indirectly related to the original meaning of the word”

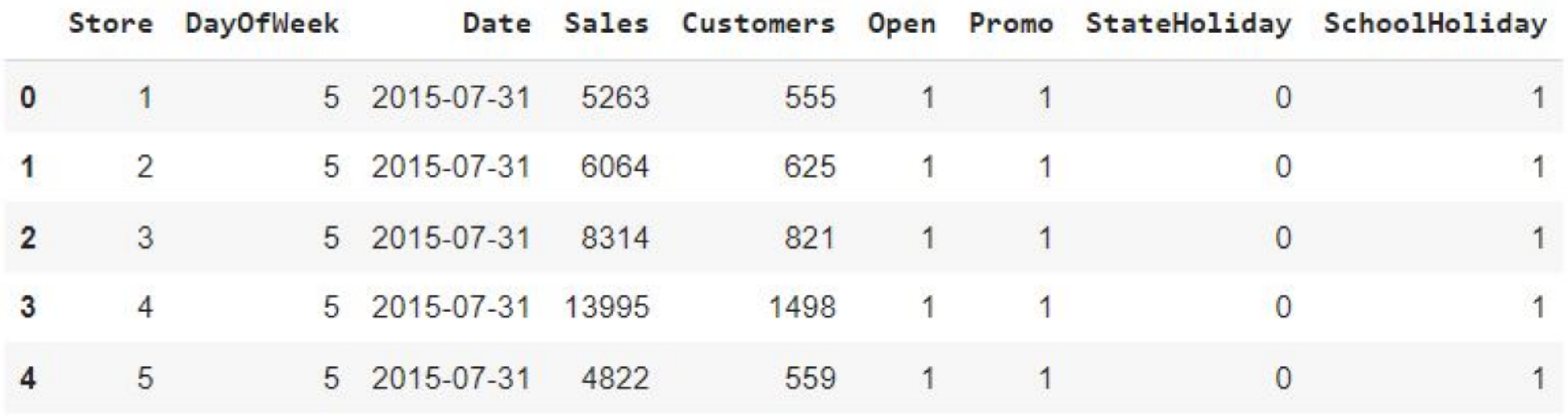
- ALLEN B. DOWNEY (Think Stats, 2nd edition)

A BRIEF INTRODUCTION TO THE DATA AND THE PROBLEM STATEMENT

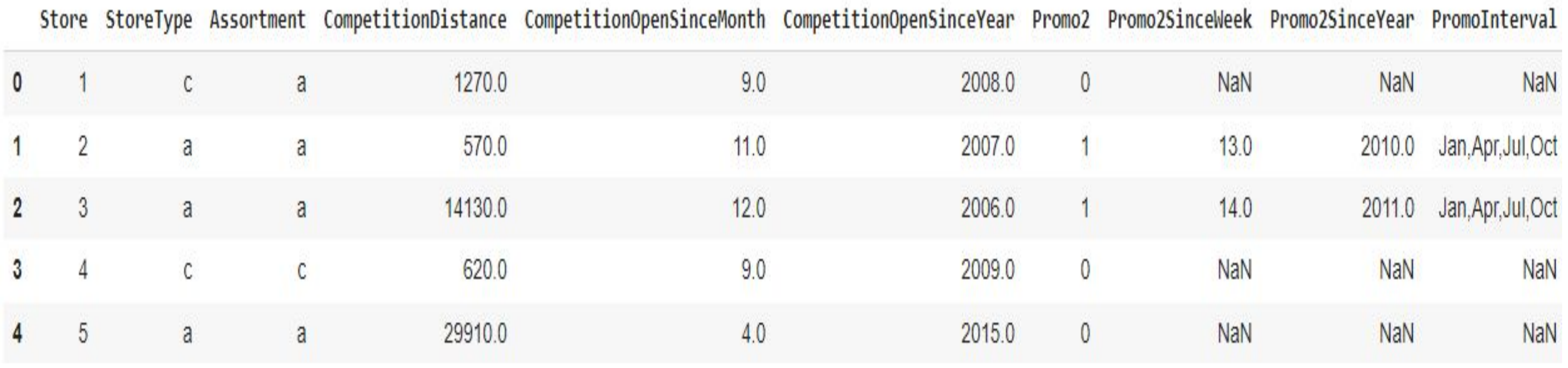
1. Rossmann operates over 3,000 drug stores in 7 European countries.
2. Currently, Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance.
3. We're provided with historical sales data for 1,115 Rossmann stores. The dataset has 9 variables and 1,017,209 records.
4. A supplementary dataset consisting of the 10 variables describing 10 different characteristics for each of the 1,115 distinct store is provided. Naturally, it has 1,115 records.
5. The succeeding slides contain some snapshots of other information of both the datasets



```
Store            0
DayOfWeek        0
Date             0
Sales            0
Customers        0
Open             0
Promo            0
StateHoliday     0
SchoolHoliday    0
dtype: int64
--------------------------------
Store                          0
StoreType                      0
Assortment                     0
CompetitionDistance            3
CompetitionOpenSinceMonth    354
CompetitionOpenSinceYear     354
Promo2                         0
Promo2SinceWeek              544
Promo2SinceYear              544
PromoInterval                544
dtype: int64
```



	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

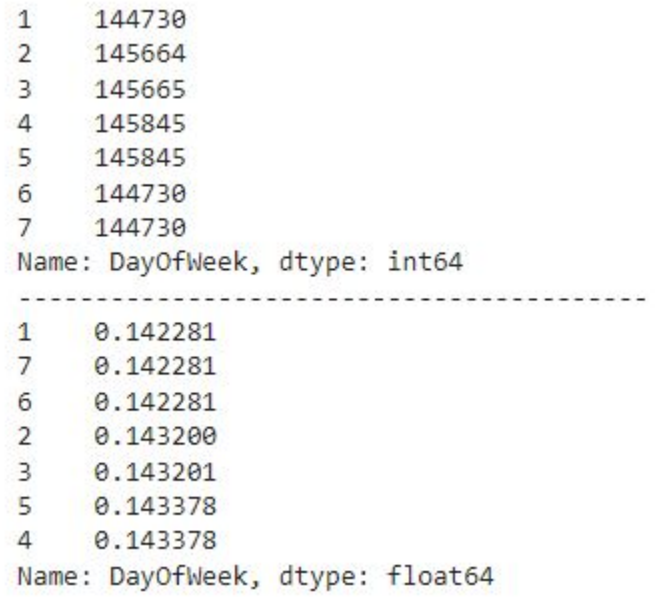


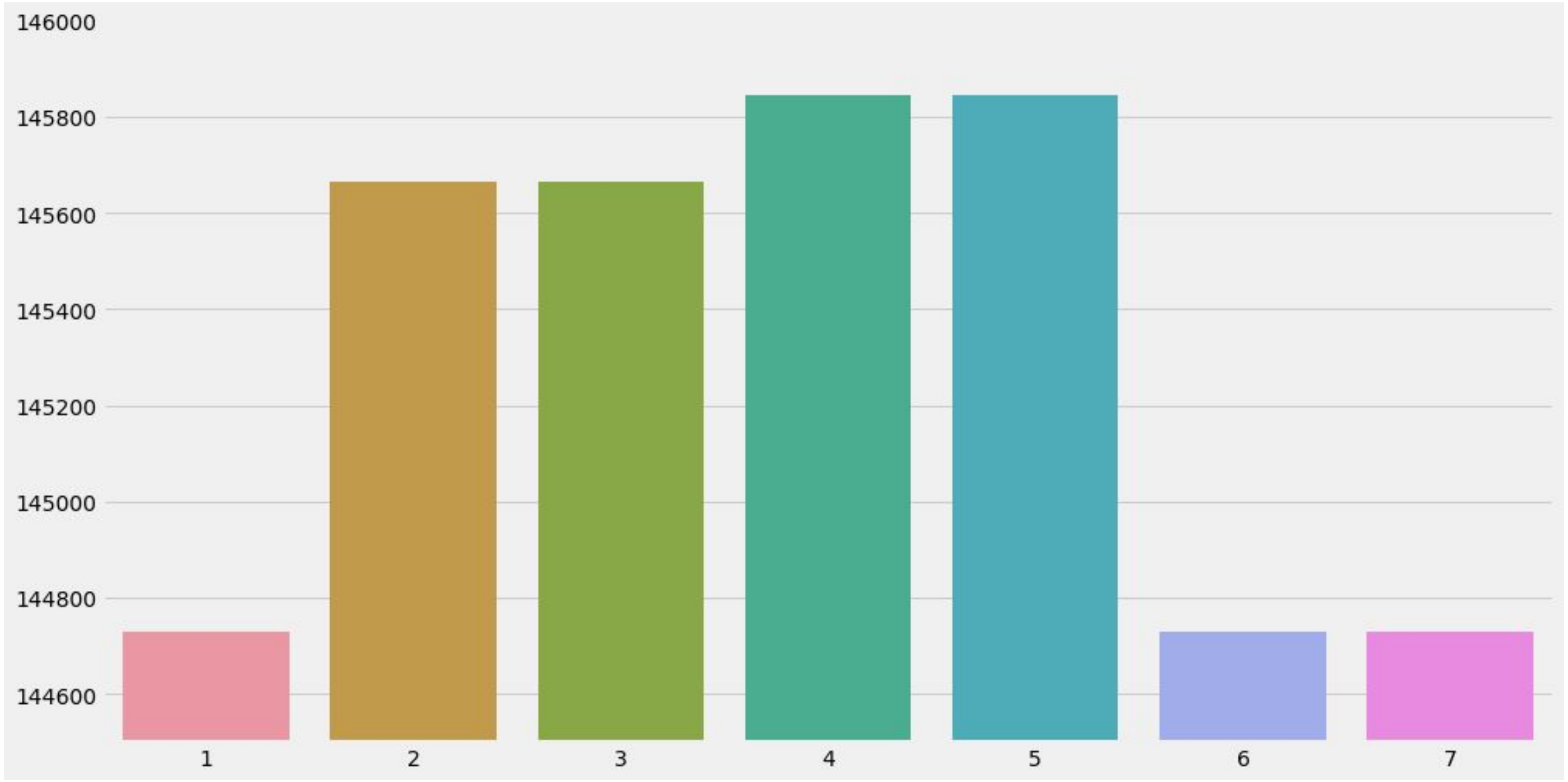
	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
0	1	c	a	1270.0	9.0	2008.0	0	NaN	NaN	NaN
1	2	a	a	570.0	11.0	2007.0	1	13.0	2010.0	Jan,Apr,Jul,Oct
2	3	a	a	14130.0	12.0	2006.0	1	14.0	2011.0	Jan,Apr,Jul,Oct
3	4	c	c	620.0	9.0	2009.0	0	NaN	NaN	NaN
4	5	a	a	29910.0	4.0	2015.0	0	NaN	NaN	NaN

SALES DATA

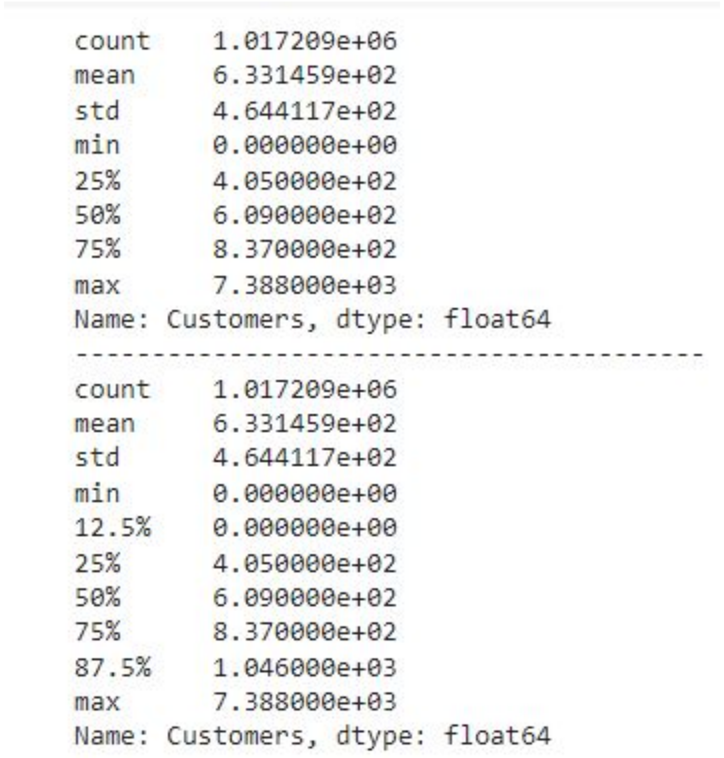
a. Univariate Analysis

1. Day of Week

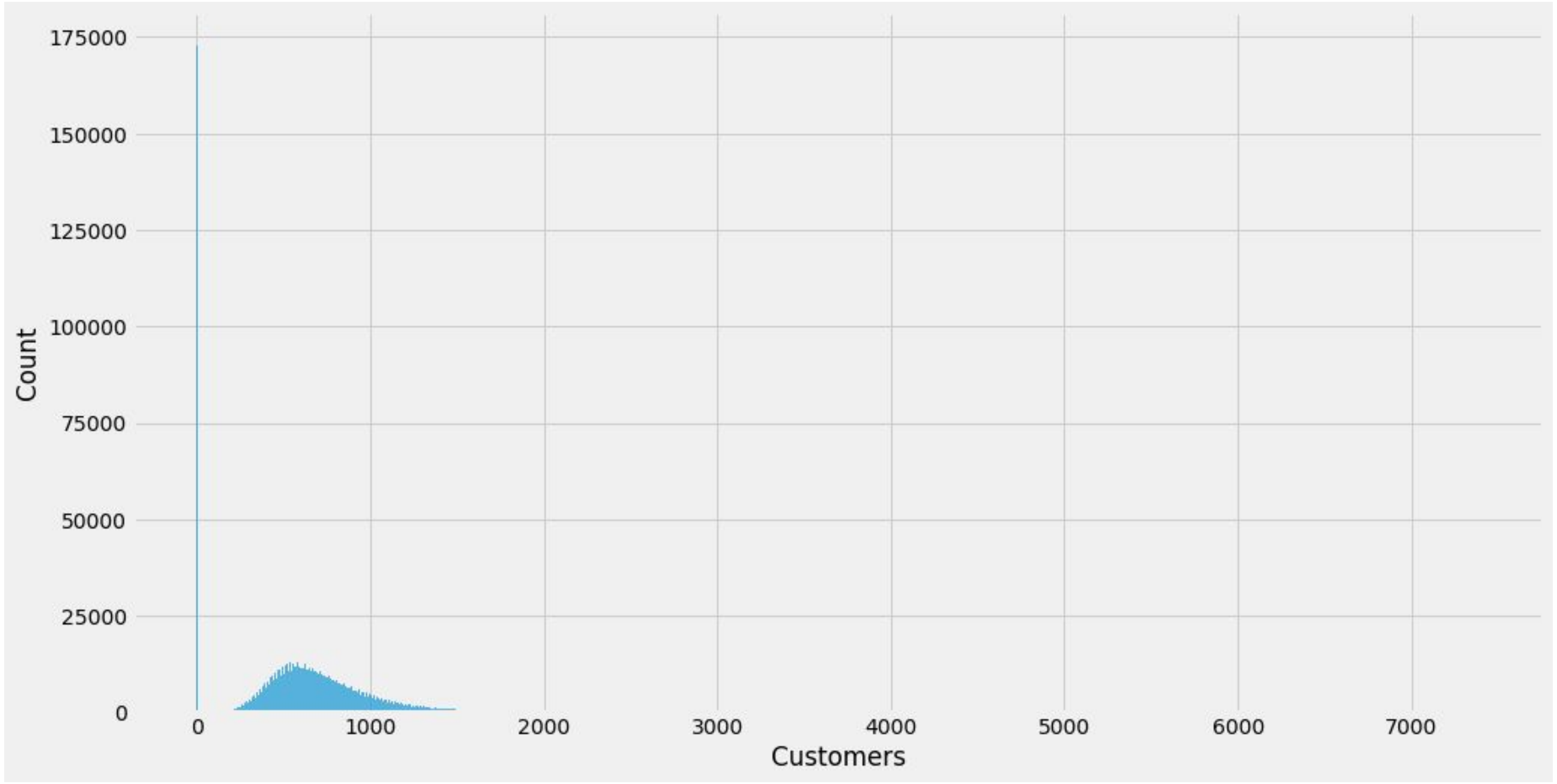

```
1    144730
2    145664
3    145665
4    145845
5    145845
6    144730
7    144730
Name: DayOfWeek, dtype: int64
-----------------------------------------
1    0.142281
7    0.142281
6    0.142281
2    0.143200
3    0.143201
5    0.143378
4    0.143378
Name: DayOfWeek, dtype: float64
```


146000
145800
145600
145400
145200
145000
144800
144600
1
2
3
4
5
6
7

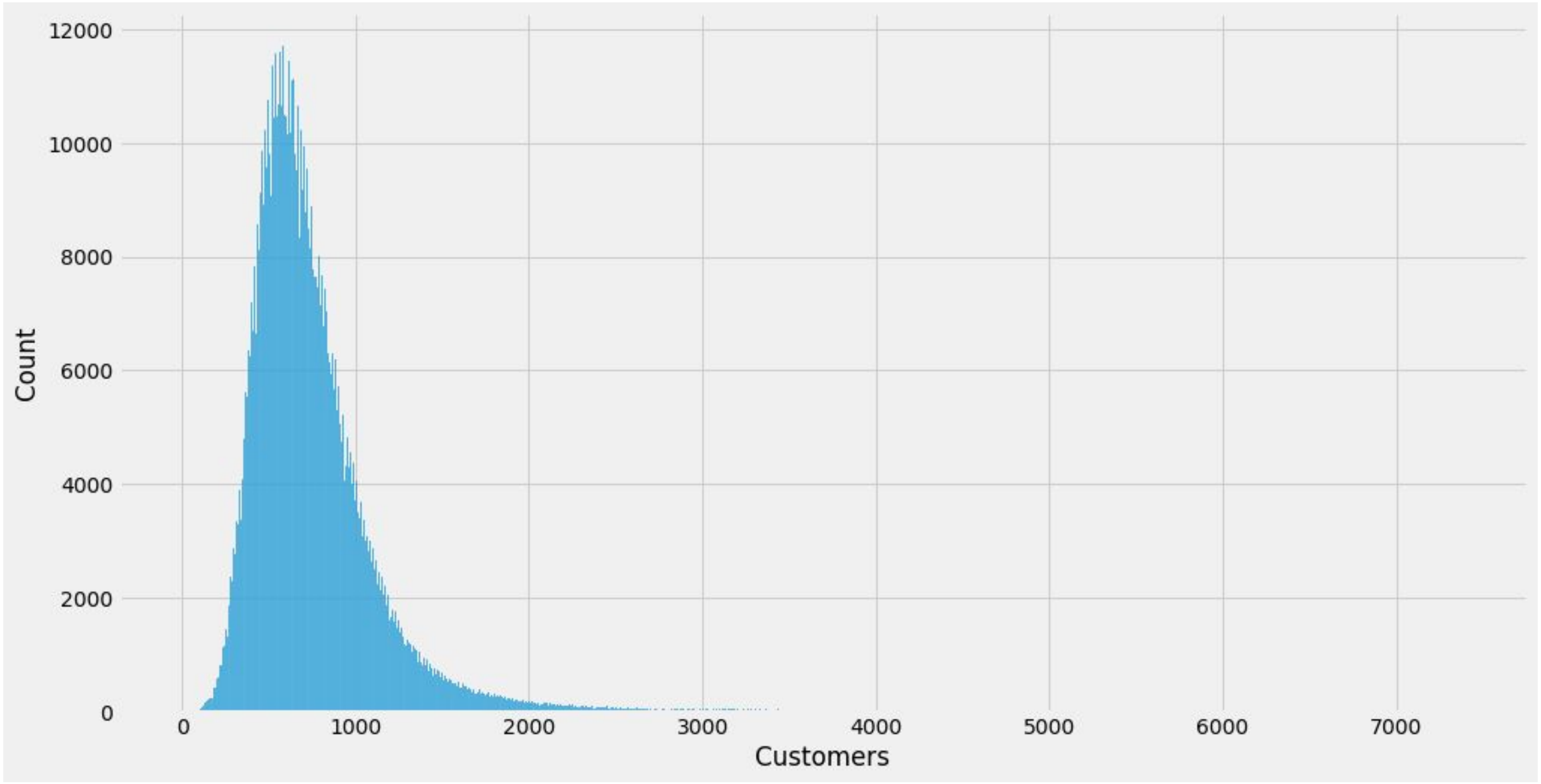
b. Customers



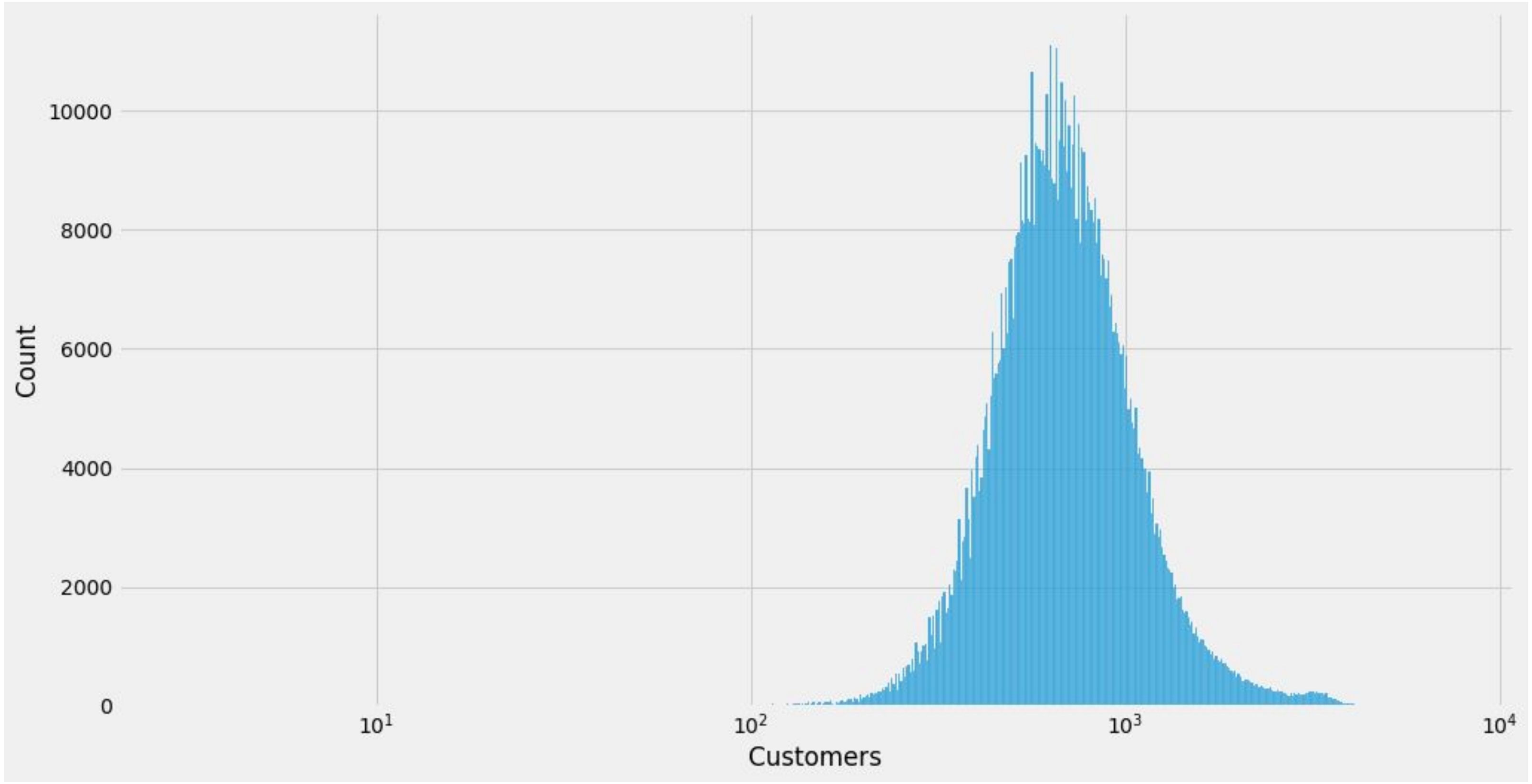
```
count    1.017209e+06
mean     6.331459e+02
std      4.644117e+02
min      0.000000e+00
25%      4.050000e+02
50%      6.090000e+02
75%      8.370000e+02
max      7.388000e+03
Name: Customers, dtype: float64
-----------------------------------------
count    1.017209e+06
mean     6.331459e+02
std      4.644117e+02
min      0.000000e+00
12.5%    0.000000e+00
25%      4.050000e+02
50%      6.090000e+02
75%      8.370000e+02
87.5%    1.046000e+03
max      7.388000e+03
Name: Customers, dtype: float64
```



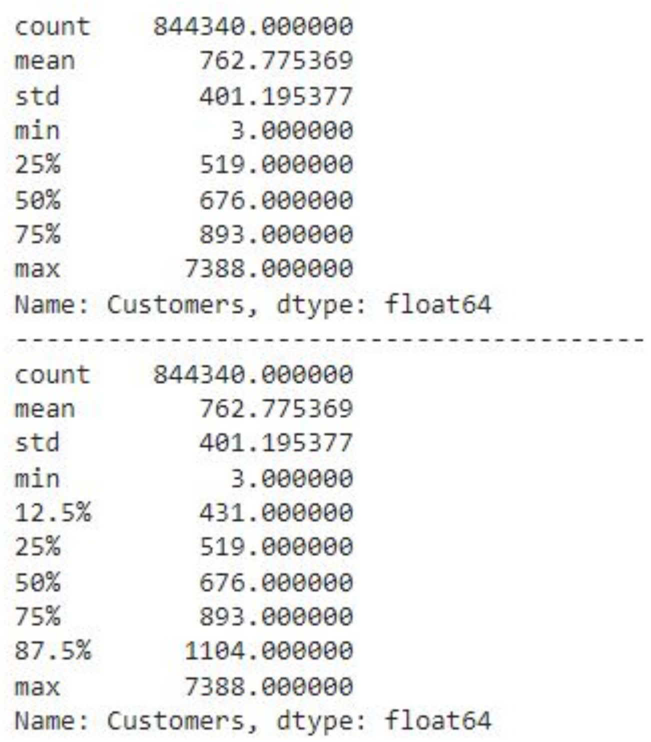
Count
Customers
175000
150000
125000
100000
75000
50000
25000
0
0
1000
2000
3000
4000
5000
6000
7000



Count
Customers
12000
10000
8000
6000
4000
2000
0
0
1000
2000
3000
4000
5000
6000
7000

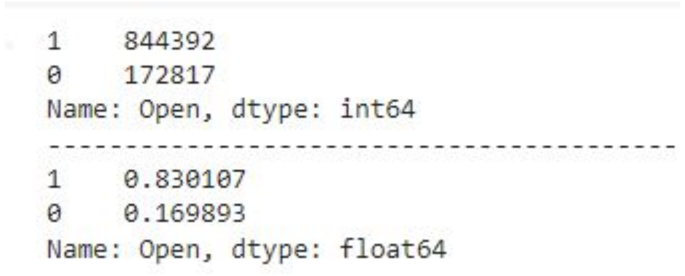

Count
10000
8000
6000
4000
2000
0
10^1
10^2
10^3
10^4
Customers

Description of the variable if we remove the zeros from it



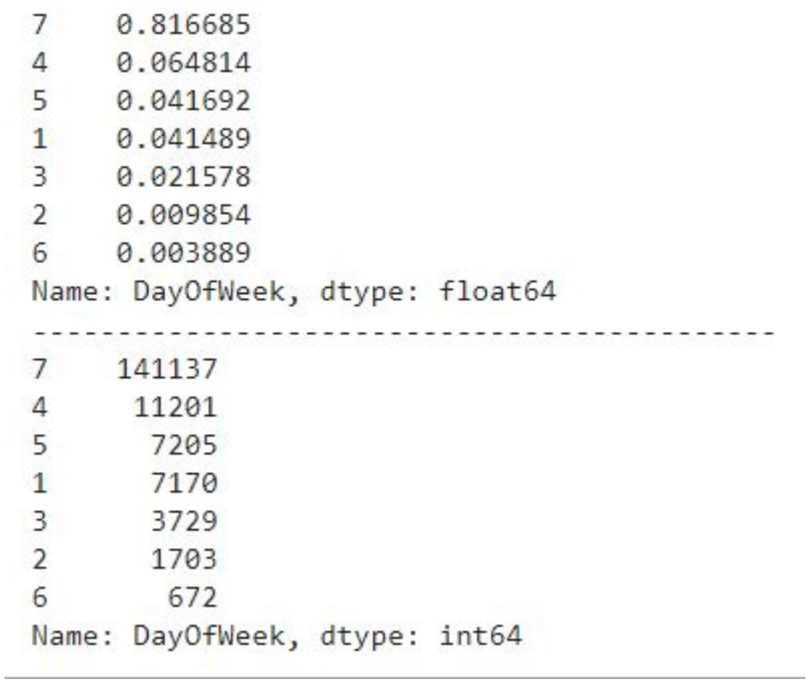
```
count    844340.000000
mean        762.775369
std         401.195377
min           3.000000
25%         519.000000
50%         676.000000
75%         893.000000
max        7388.000000
Name: Customers, dtype: float64
----------------------------------------
count    844340.000000
mean        762.775369
std         401.195377
min           3.000000
12.5%       431.000000
25%         519.000000
50%         676.000000
75%         893.000000
87.5%      1104.000000
max        7388.000000
Name: Customers, dtype: float64
```

3. Open



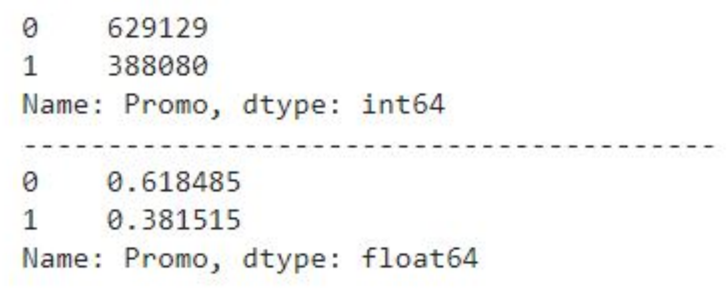
```
1    844392
0    172817
Name: Open, dtype: int64
------------------------------------------
1    0.830107
0    0.169893
Name: Open, dtype: float64
```

The stores are closed most often on Sundays and least often on Tuesday and Saturday



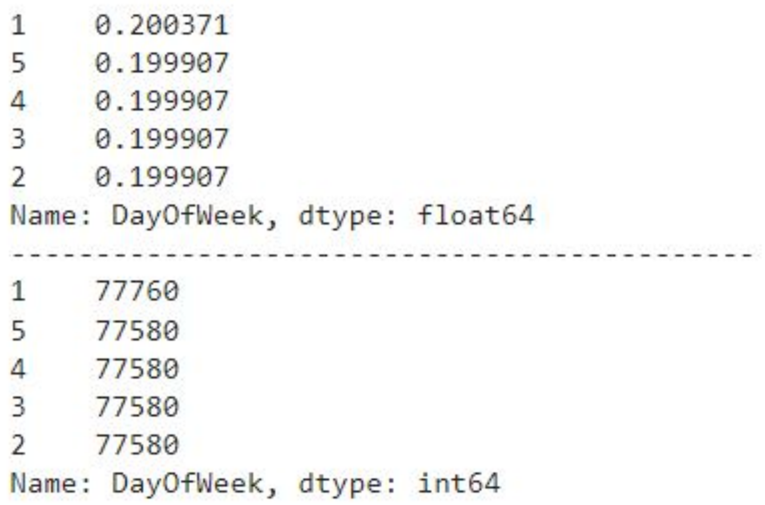
```
7    0.816685
4    0.064814
5    0.041692
1    0.041489
3    0.021578
2    0.009854
6    0.003889
Name: DayOfWeek, dtype: float64
-------------------------------------------
7    141137
4     11201
5      7205
1      7170
3      3729
2      1703
6       672
Name: DayOfWeek, dtype: int64
```

4. Promo



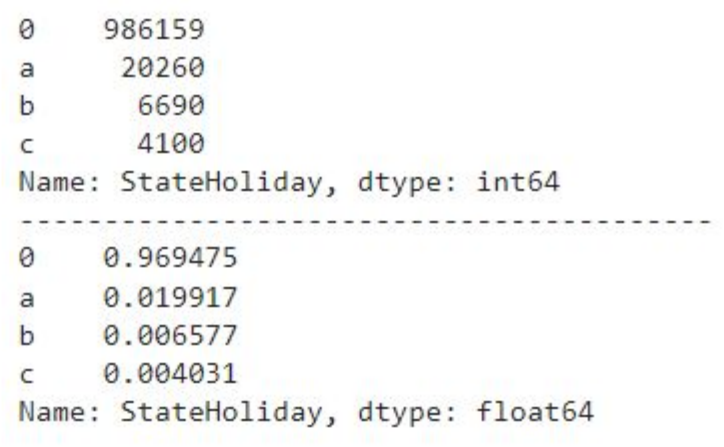
```
0    629129
1    388080
Name: Promo, dtype: int64
-----------------------------------------
0    0.618485
1    0.381515
Name: Promo, dtype: float64
```

Let's compare 'Promo' too with 'DayOfWeek'



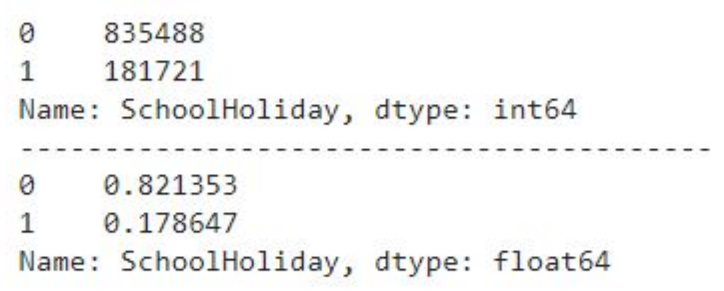
```
1    0.200371
5    0.199907
4    0.199907
3    0.199907
2    0.199907
Name: DayOfWeek, dtype: float64
--------------------------------------------
1    77760
5    77580
4    77580
3    77580
2    77580
Name: DayOfWeek, dtype: int64
```

5. State Holiday



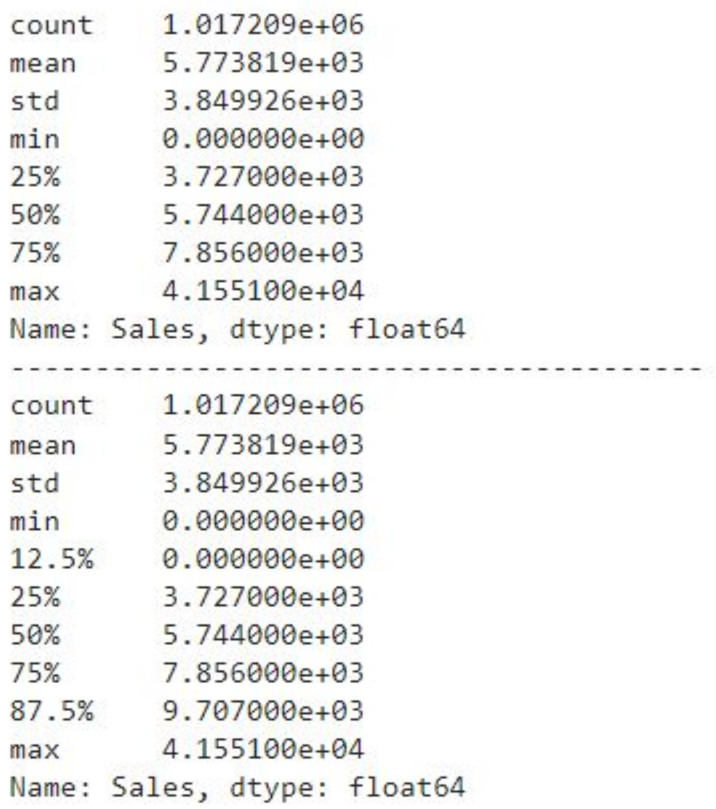
```
0    986159
a     20260
b      6690
c      4100
Name: StateHoliday, dtype: int64
-----------------------------------------
0    0.969475
a    0.019917
b    0.006577
c    0.004031
Name: StateHoliday, dtype: float64
```

6. School Holiday



```
0    835488
1    181721
Name: SchoolHoliday, dtype: int64
----------------------------------------
0    0.821353
1    0.178647
Name: SchoolHoliday, dtype: float64
```

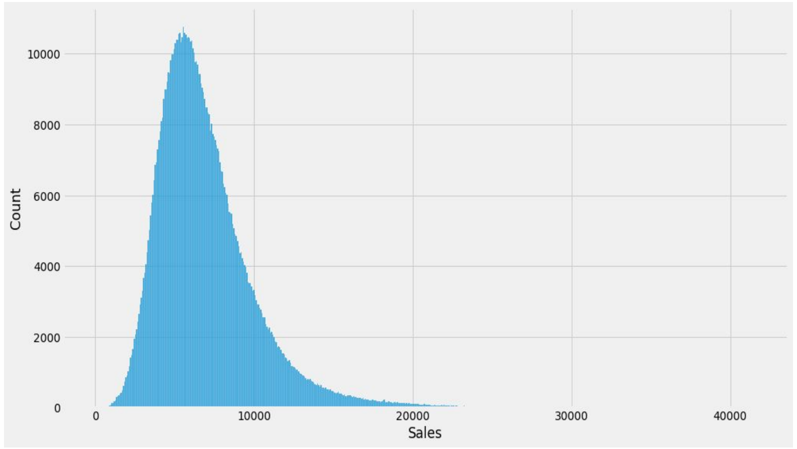
7. Sales



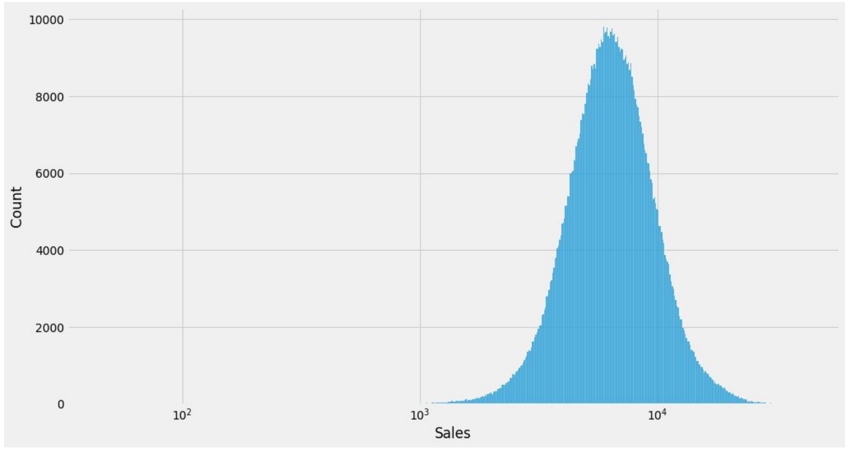
```
count    1.017209e+06
mean     5.773819e+03
std      3.849926e+03
min      0.000000e+00
25%      3.727000e+03
50%      5.744000e+03
75%      7.856000e+03
max      4.155100e+04
Name: Sales, dtype: float64
----------------------------------------
count    1.017209e+06
mean     5.773819e+03
std      3.849926e+03
min      0.000000e+00
12.5%    0.000000e+00
25%      3.727000e+03
50%      5.744000e+03
75%      7.856000e+03
87.5%    9.707000e+03
max      4.155100e+04
Name: Sales, dtype: float64
```

Sales Data Having Removed the Zeros

1. Original

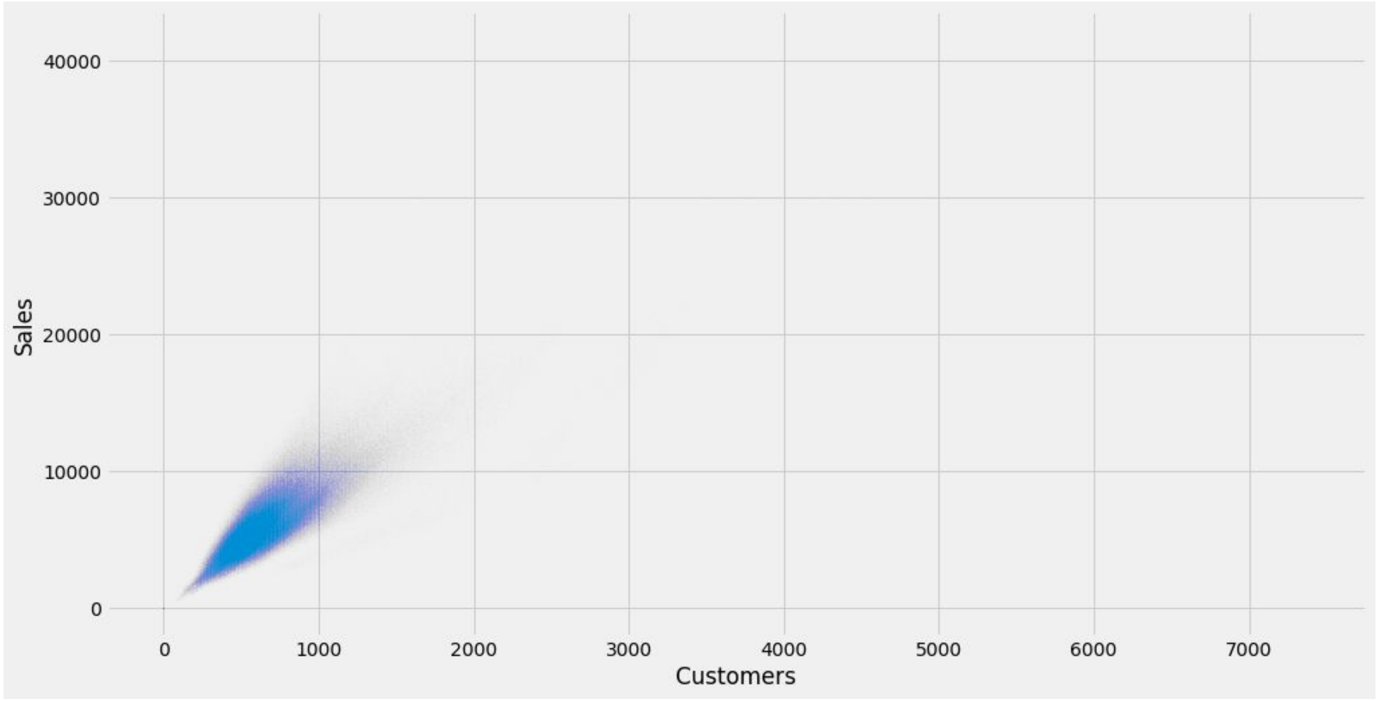


2. Log Transformed

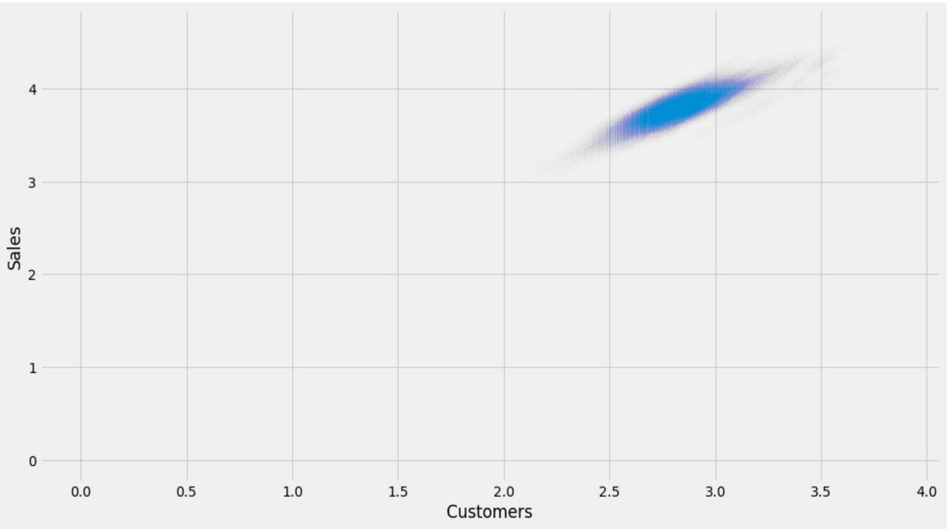


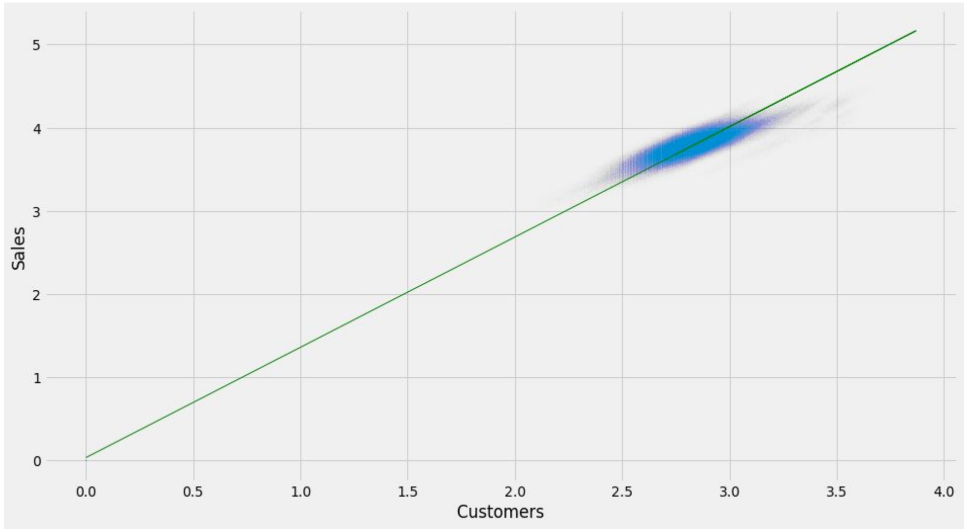
b. Bivariate Analysis

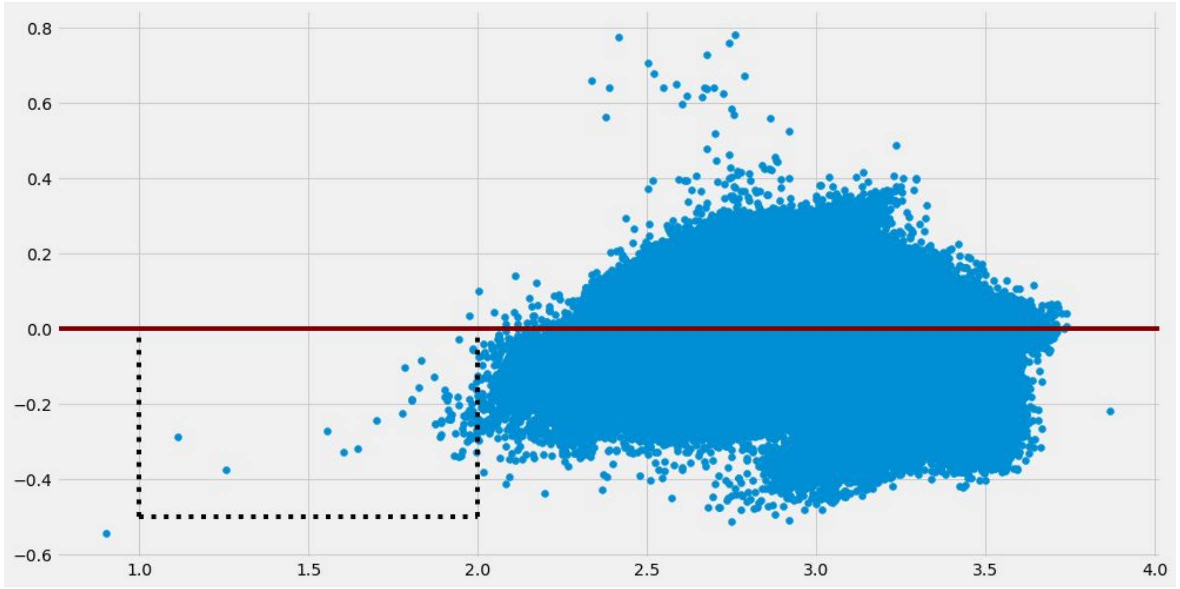
1. Customers vs Sales



Log Transformed customers vs log transformed sales

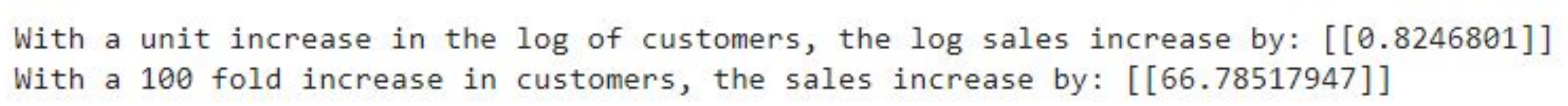




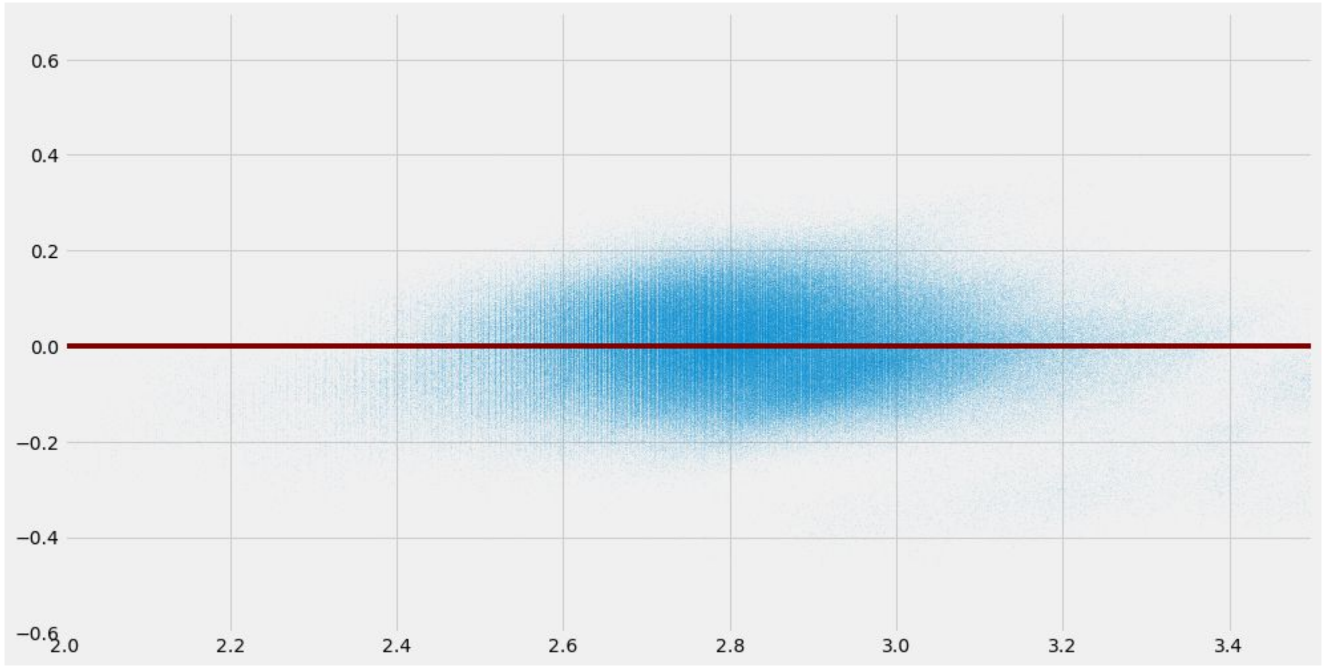


When plotted the log values for the customers against the residuals, it becomes apparent that when the customers are in the range 1 to 2; the residuals are overwhelmingly between -0.1 to -0.5. 1 and 2 on log scale will correspond to 10 to 100 raw customers. And the corresponding log value for the residuals demonstrate that *when the customers are in the range of 10 to 100, the sales are off by around 21% to 69%.*

The rectangular region highlights that proportion.



```
With a unit increase in the log of customers, the log sales increase by: [[0.8246801]]
With a 100 fold increase in customers, the sales increase by: [[66.78517947]]
```

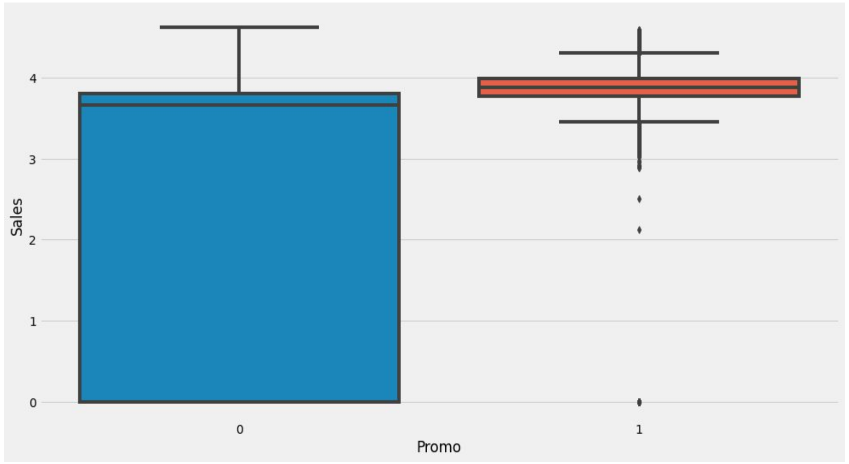


A glance at the middle portion of the plot shows that when the no. of customers are in the range of 150 to 2500, the residuals are evenly distributed on either side of zero (The mean of residuals).

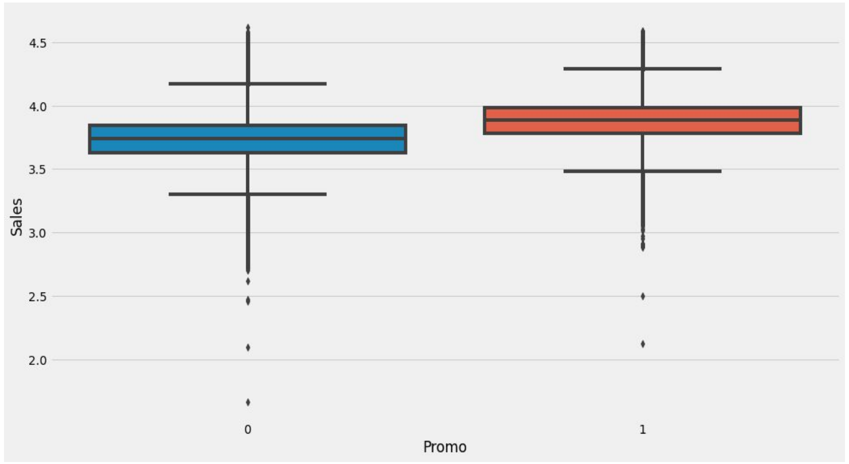
The apparent vertical lines appear because of the discrete nature of the data. Of course, one expects the no. of customers to be count of whole nos. not real nos.

2. Promo vs Sales

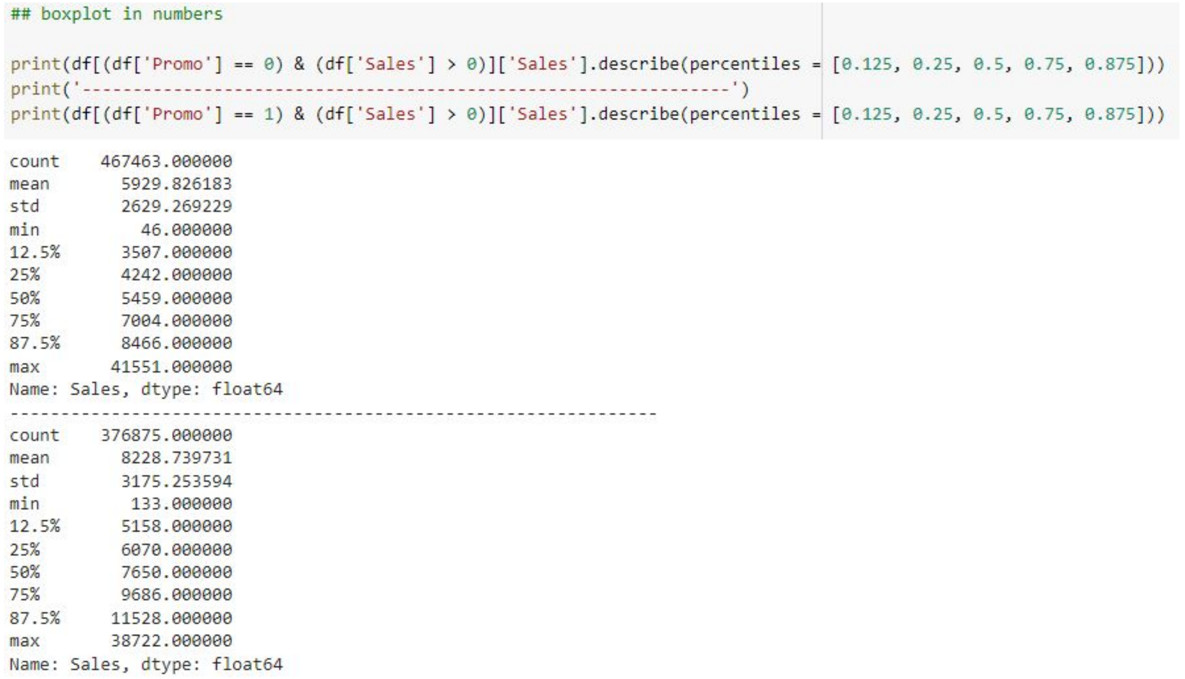
Log transformation with zeros in the sales variable



Log transformation sans zeros in the sales variable



The mean, and median of the sales when there's a promotion in place is greater than the corresponding figures when there's no promotion in place. The range of the values below the lower whisker is more for when there is no promotion than its counterpart. But surprisingly, the highest value for the sales when there is not promotion is greater than the maximum sale when there is a promotion. Perhaps a square root transformation will make the difference more apparent.



```
## boxplot in numbers

print(df[(df['Promo'] == 0) & (df['Sales'] > 0)]['Sales'].describe(percentiles = [0.125, 0.25, 0.5, 0.75, 0.875]))
print('----------------------------------------------------------------')
print(df[(df['Promo'] == 1) & (df['Sales'] > 0)]['Sales'].describe(percentiles = [0.125, 0.25, 0.5, 0.75, 0.875]))
```

```
count    467463.000000
mean       5929.826183
std        2629.269229
min          46.000000
12.5%      3507.000000
25%        4242.000000
50%        5459.000000
75%        7004.000000
87.5%      8466.000000
max       41551.000000
Name: Sales, dtype: float64
----------------------------------------------------------------
count    376875.000000
mean       8228.739731
std        3175.253594
min         133.000000
12.5%      5158.000000
25%        6070.000000
50%        7650.000000
75%        9686.000000
87.5%     11528.000000
max       38722.000000
Name: Sales, dtype: float64
```

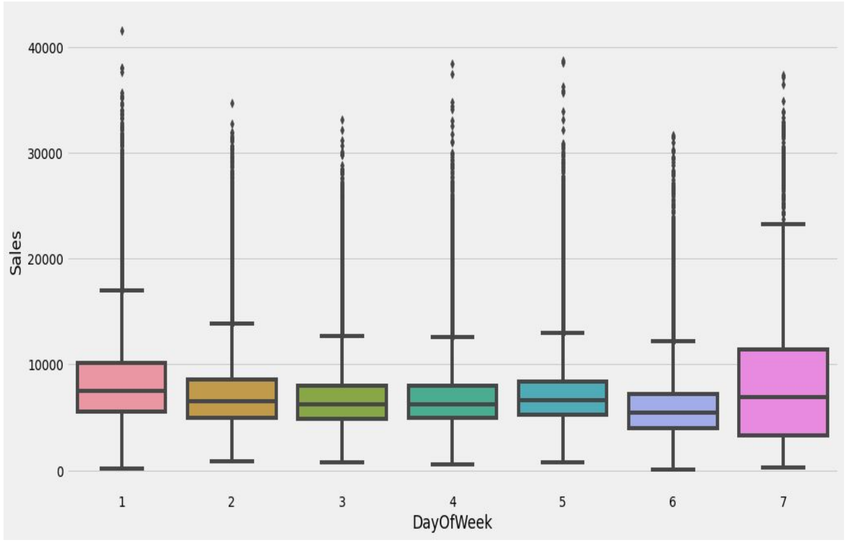
While the other statistics reveal what was expected, one thing that stands out is the values in the top 25 percentile for both the categories. Notice that the range is larger for when there is no promotion than when there's one in place.

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
44393	909	1	2015-06-22	41551	1721	1	0	0	0

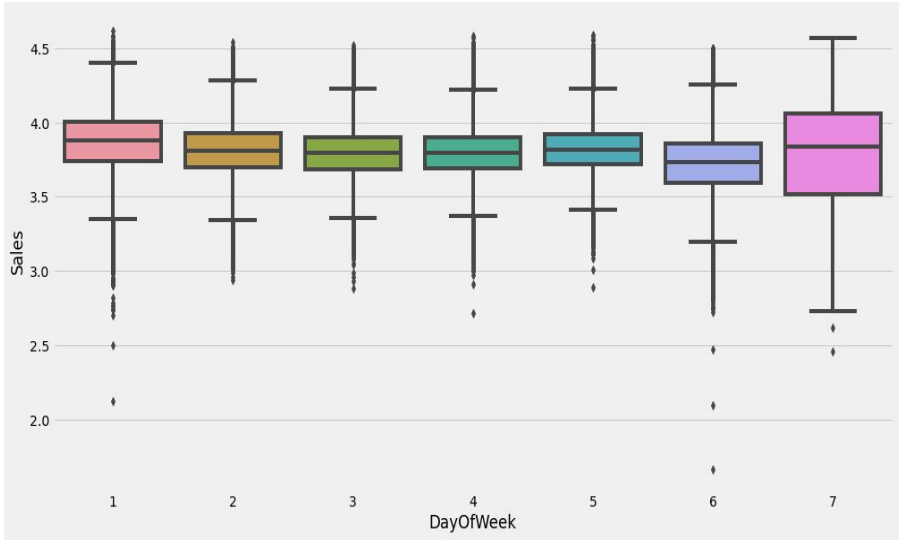
The corresponding observations in the other variables - especially customers - when the no. of sales is 41551 suggest that this might be a potential outlier. As, one may expect - from the previous analysis - an estimated sales of not more than 12K or 13K for 1721 customers. However, not being much sure about this assumption, I keep the record in the dataset.

3. Day of Week vs Sales

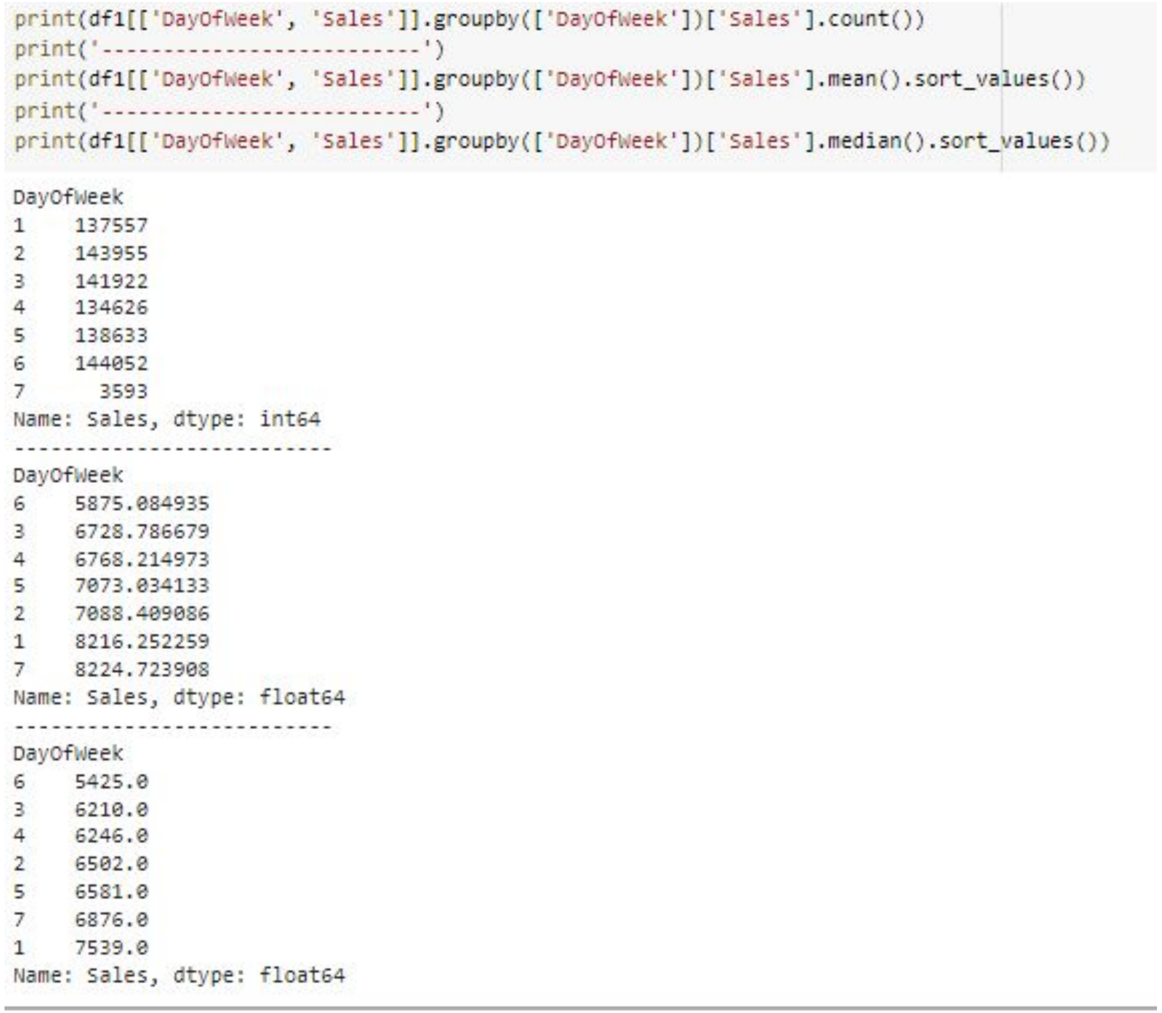
Without log Transformation



With log Transformation

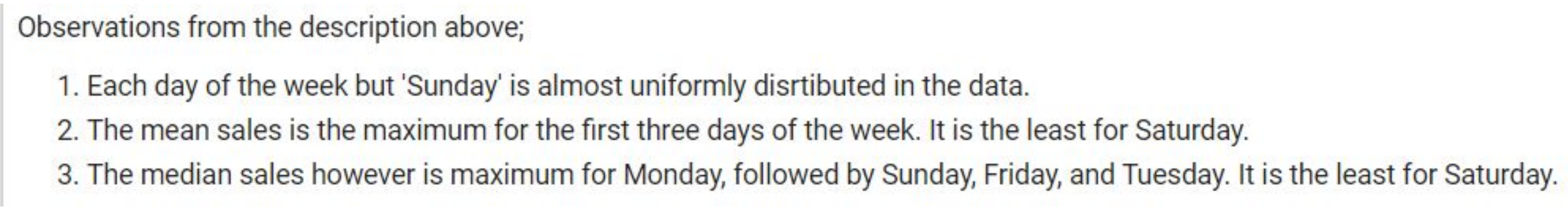


The sales on Mondays appears to outnumber the corresponding figures for Tuesday through Saturday. There's a sudden increase for Friday but the numbers for Saturday is disappointing. The sales figures for Sunday is only second to that of Monday. But Sunday is underrepresented in the data. It is because we have not taken into consideration the records for which the sales is zero. Many of the stores are mostly closed on the Sundays. So, in removing these records, we have removed the data for many sundays too. But in its present form (re-expressed) the Sundays reveal more than they would hide if we took the zeros in consideration.



```
print(df1[['DayOfWeek', 'Sales']].groupby(['DayOfWeek'])['Sales'].count())
print('--------------------------')
print(df1[['DayOfWeek', 'Sales']].groupby(['DayOfWeek'])['Sales'].mean().sort_values())
print('--------------------------')
print(df1[['DayOfWeek', 'Sales']].groupby(['DayOfWeek'])['Sales'].median().sort_values())
```

```
DayOfWeek
1    137557
2    143955
3    141922
4    134626
5    138633
6    144052
7      3593
Name: Sales, dtype: int64
--------------------------
DayOfWeek
6    5875.084935
3    6728.786679
4    6768.214973
5    7073.034133
2    7088.409086
1    8216.252259
7    8224.723908
Name: Sales, dtype: float64
--------------------------
DayOfWeek
6    5425.0
3    6210.0
4    6246.0
2    6502.0
5    6581.0
7    6876.0
1    7539.0
Name: Sales, dtype: float64
```



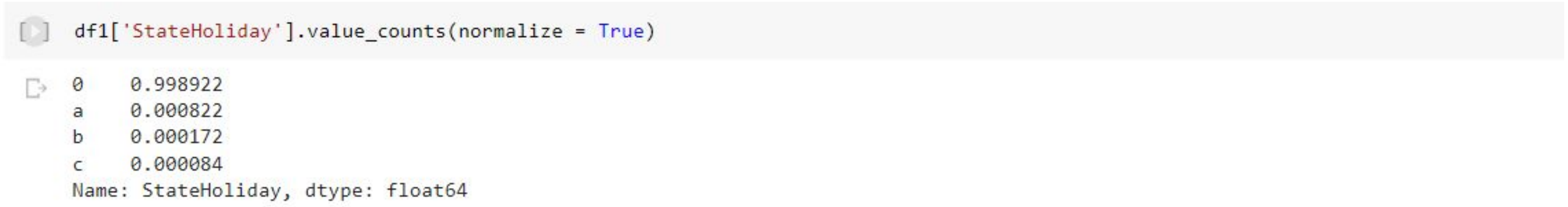
Observations from the description above;

1. Each day of the week but 'Sunday' is almost uniformly disrtibuted in the data.
2. The mean sales is the maximum for the first three days of the week. It is the least for Saturday.
3. The median sales however is maximum for Monday, followed by Sunday, Friday, and Tuesday. It is the least for Saturday.

4. Open vs Sales

Store is always open when the sales is greater than zero. This variable wouldn't matter for making predictions in regression. Nonetheless, it will be useful for the classification model (We'll call attention to it in sometime).

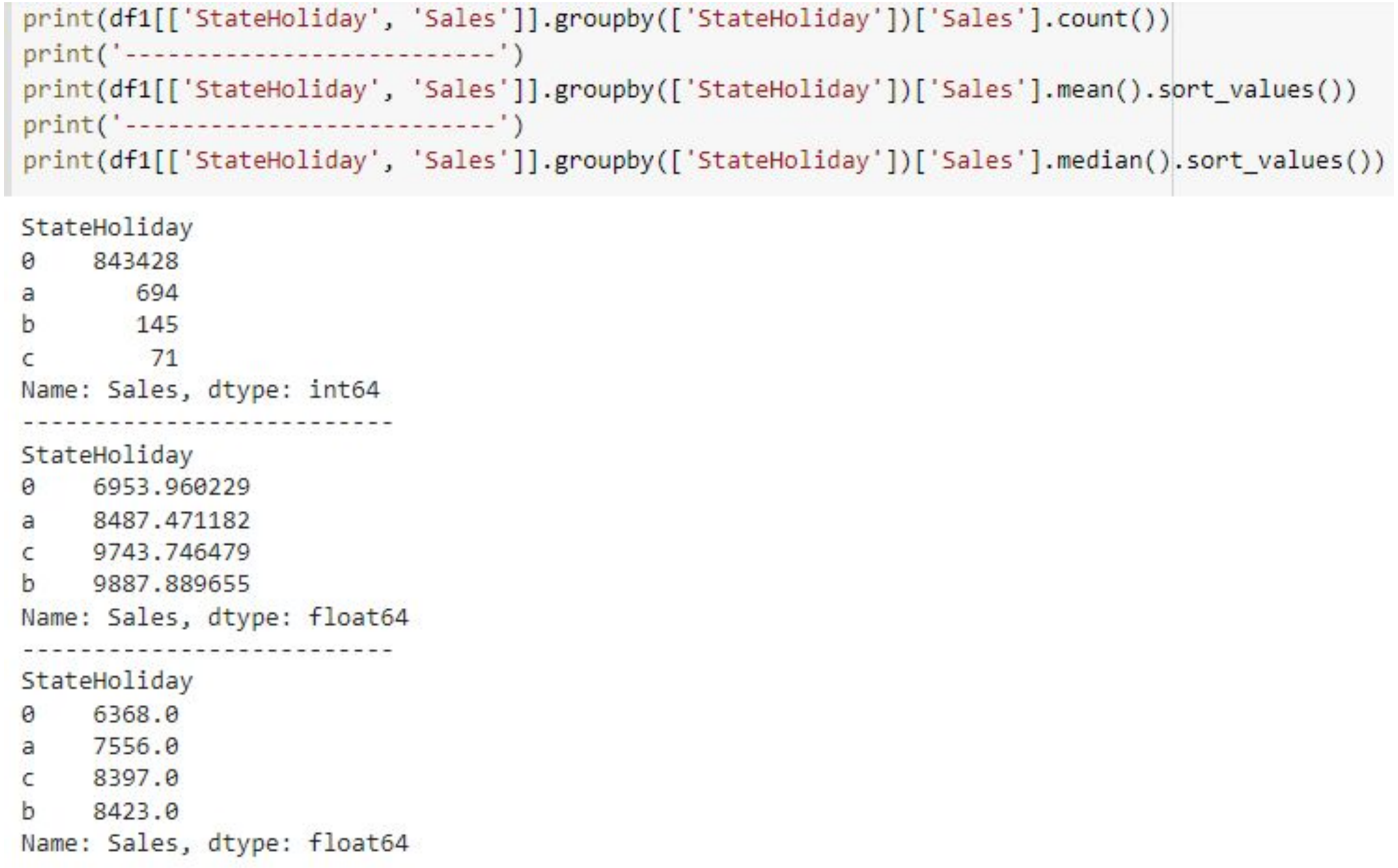
5. State Holiday vs Sales



```
df1['StateHoliday'].value_counts(normalize = True)
```

```
0    0.998922
a    0.000822
b    0.000172
c    0.000084
Name: StateHoliday, dtype: float64
```

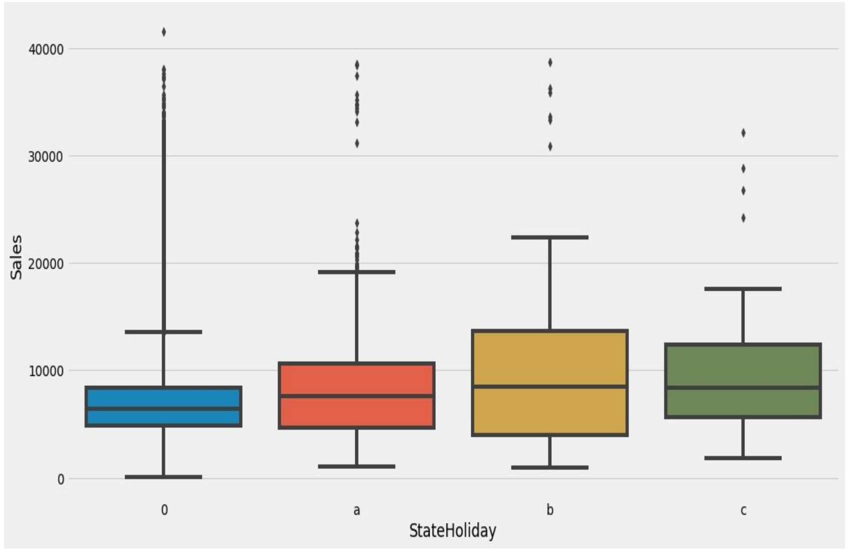
State holidays are severly underrepresented, again this is perhaps when the corresponding sales are zero. As one may expect sales to be zero mostly when there's holiday, like Sunday.



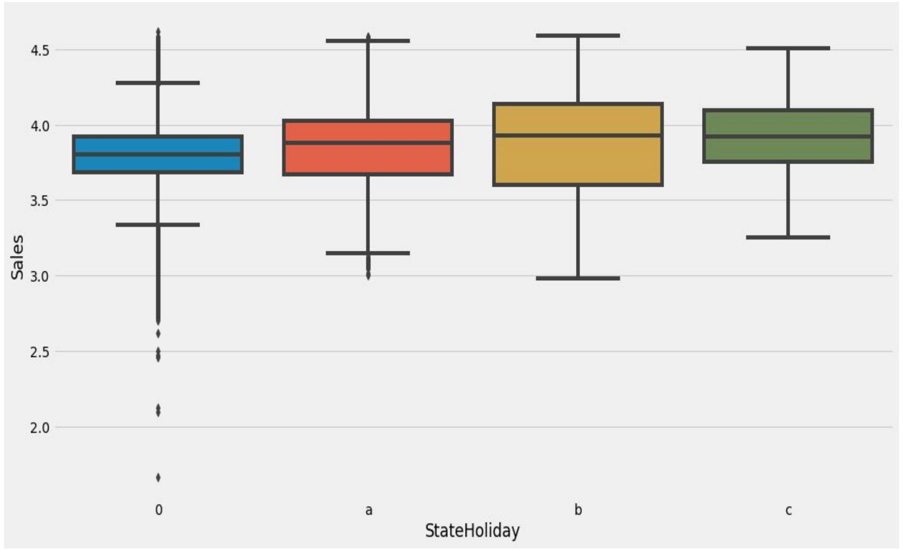
```
print(df1[['StateHoliday', 'Sales']].groupby(['StateHoliday'])['Sales'].count())
print('--------------------------')
print(df1[['StateHoliday', 'Sales']].groupby(['StateHoliday'])['Sales'].mean().sort_values())
print('--------------------------')
print(df1[['StateHoliday', 'Sales']].groupby(['StateHoliday'])['Sales'].median().sort_values())
```

```
StateHoliday
0    843428
a       694
b       145
c        71
Name: Sales, dtype: int64
--------------------------
StateHoliday
0    6953.960229
a    8487.471182
c    9743.746479
b    9887.889655
Name: Sales, dtype: float64
--------------------------
StateHoliday
0    6368.0
a    7556.0
c    8397.0
b    8423.0
Name: Sales, dtype: float64
```

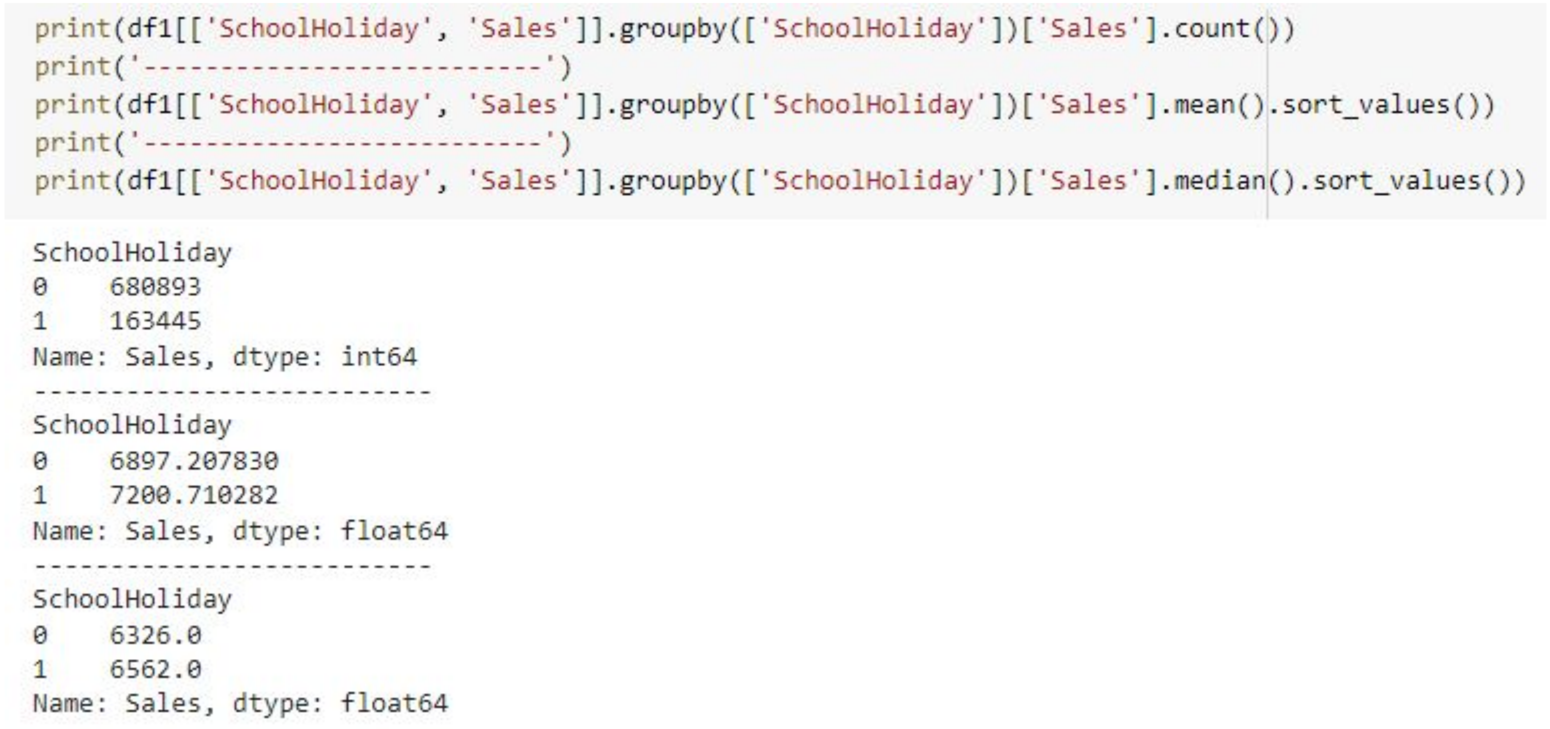
With log Transformation



With log Transformation



6. State Holiday vs Sales

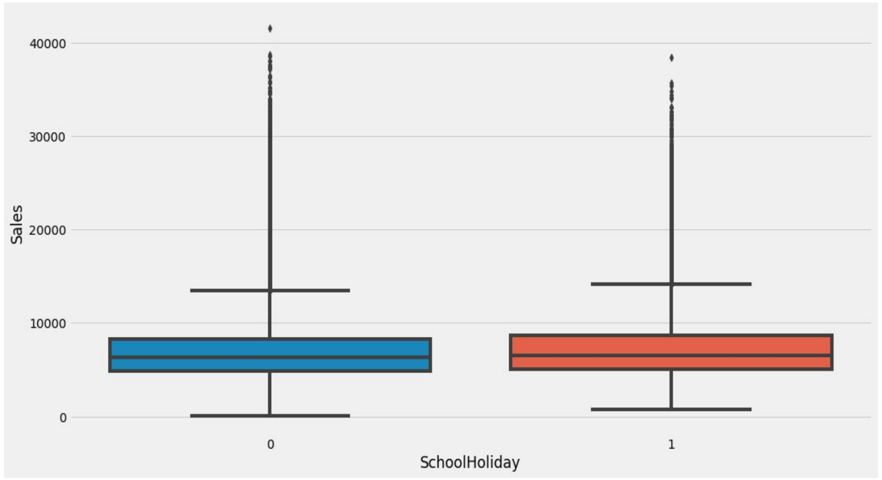


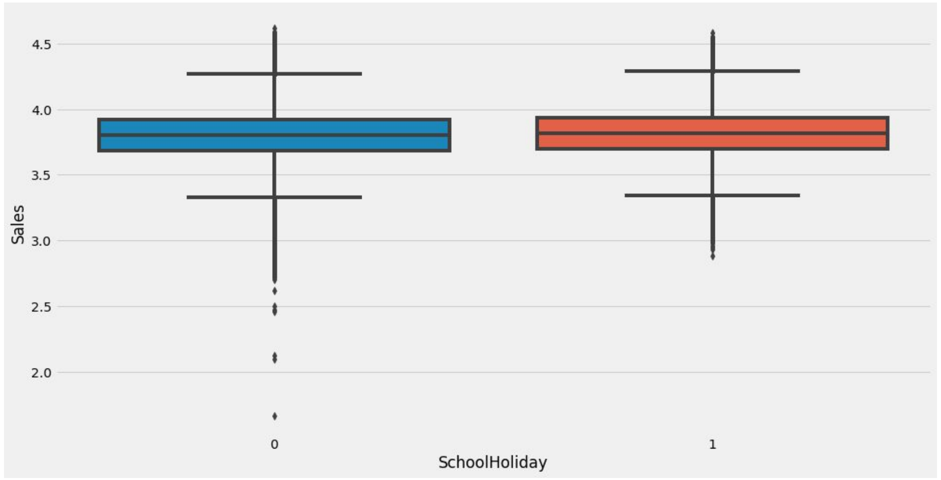
```python
print(df1[['SchoolHoliday', 'Sales']].groupby(['SchoolHoliday'])['Sales'].count())
print('--------------------------')
print(df1[['SchoolHoliday', 'Sales']].groupby(['SchoolHoliday'])['Sales'].mean().sort_values())
print('--------------------------')
print(df1[['SchoolHoliday', 'Sales']].groupby(['SchoolHoliday'])['Sales'].median().sort_values())
```

```
SchoolHoliday
0    680893
1    163445
Name: Sales, dtype: int64
--------------------------
SchoolHoliday
0    6897.207830
1    7200.710282
Name: Sales, dtype: float64
--------------------------
SchoolHoliday
0    6326.0
1    6562.0
Name: Sales, dtype: float64
```

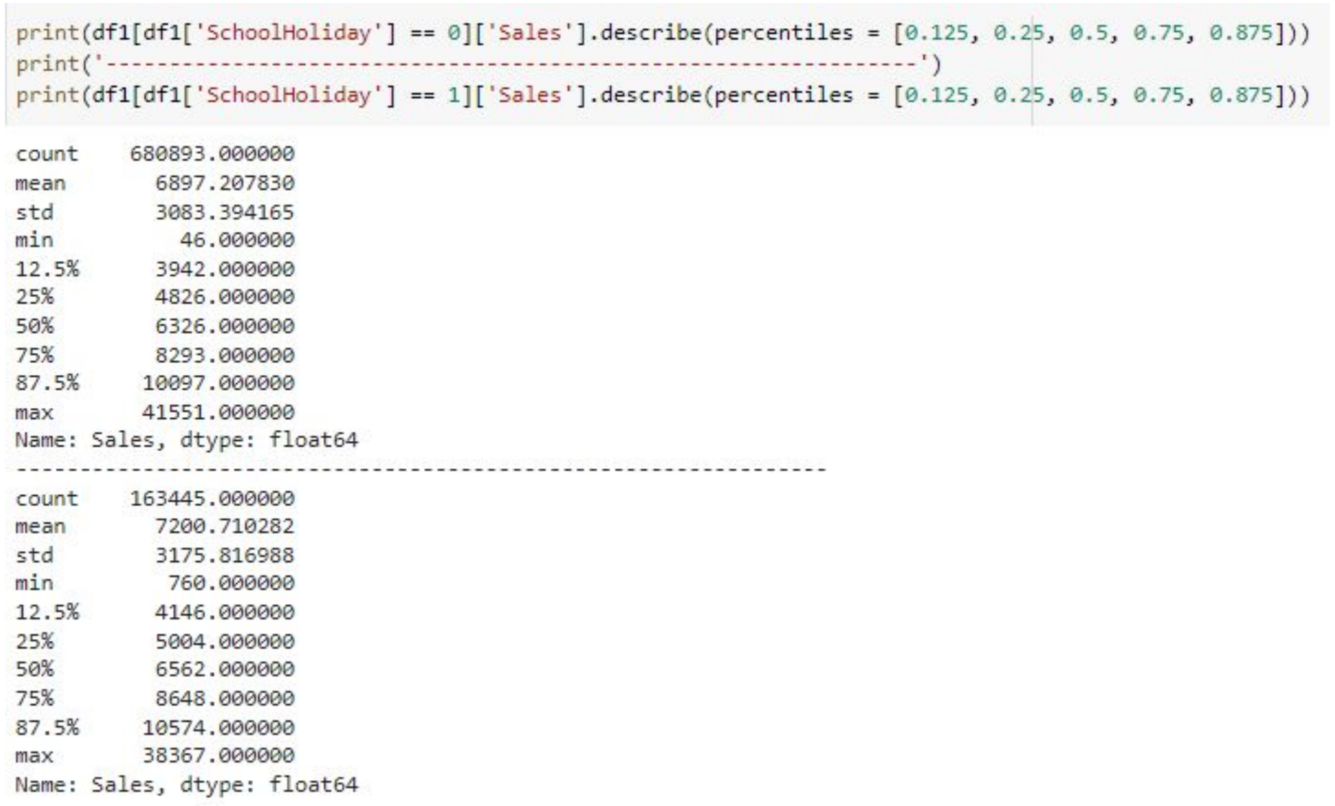
Without log Transformation

With log Transformation





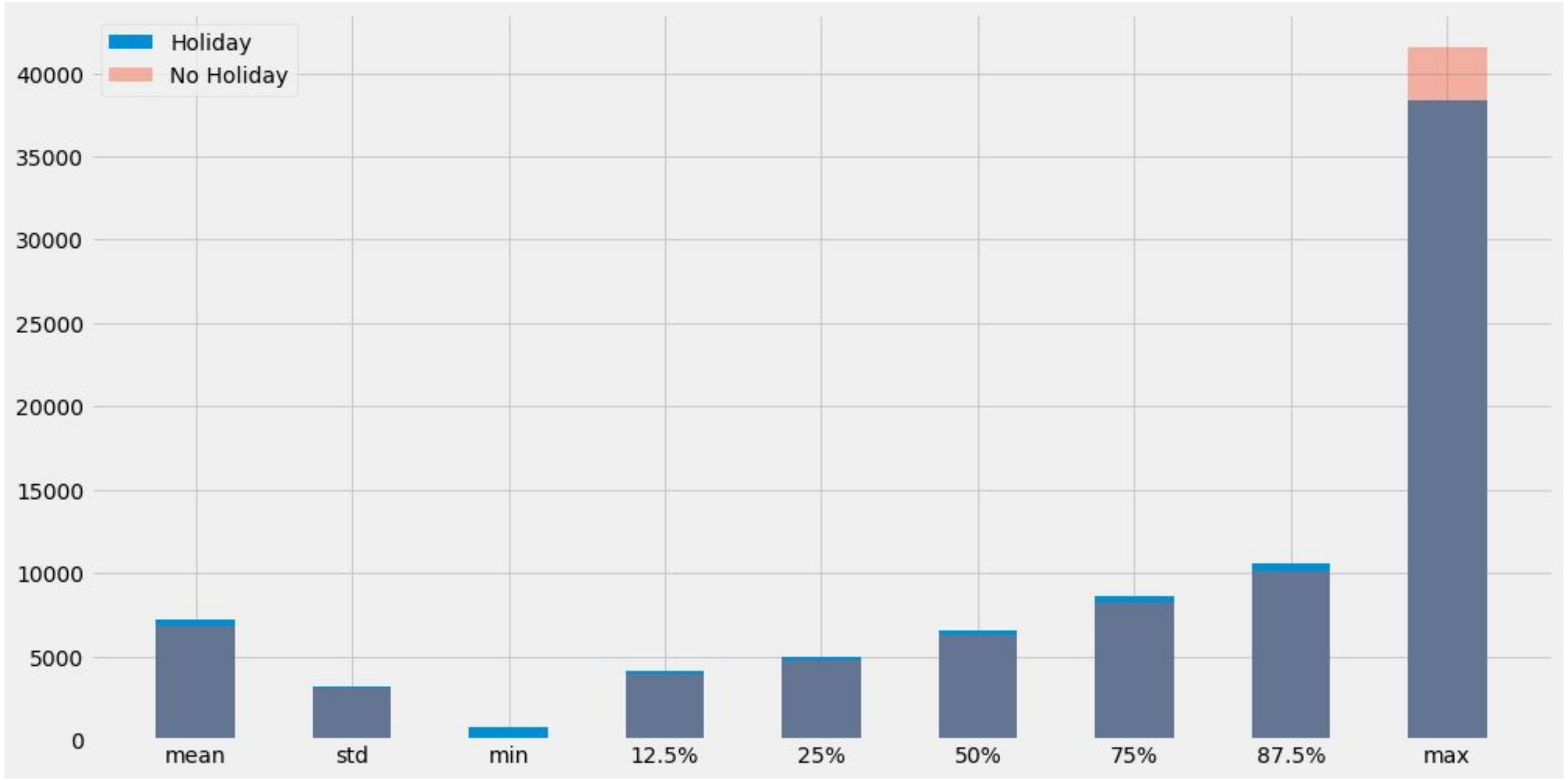
I would like to observe the data more closely



```
print(df1[df1['SchoolHoliday'] == 0]['Sales'].describe(percentiles = [0.125, 0.25, 0.5, 0.75, 0.875]))
print('----------------------------------------------------------------')
print(df1[df1['SchoolHoliday'] == 1]['Sales'].describe(percentiles = [0.125, 0.25, 0.5, 0.75, 0.875]))
```

```
count    680893.000000
mean       6897.207830
std        3083.394165
min          46.000000
12.5%      3942.000000
25%        4826.000000
50%        6326.000000
75%        8293.000000
87.5%     10097.000000
max       41551.000000
Name: Sales, dtype: float64
----------------------------------------------------------------
count    163445.000000
mean       7200.710282
std        3175.816988
min         760.000000
12.5%      4146.000000
25%        5004.000000
50%        6562.000000
75%        8648.000000
87.5%     10574.000000
max       38367.000000
Name: Sales, dtype: float64
```

The 50% of data is centered within 4800 and 8200 for when there's no school holiday as compared to 5000 and 8650 for when there's a school holiday. However the sales in the range of 75% of values in both the categories are almost same. The potential outlier appeared when there was a school holiday.

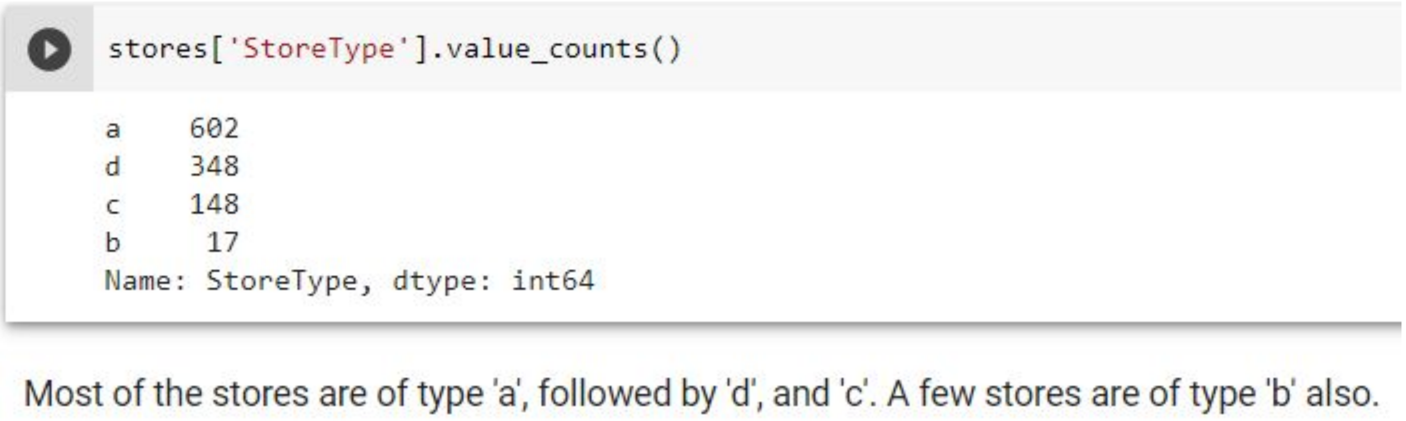


Each statistic is greater for when there's a school holiday as compared to when there's not except for the maximum value. It is greater for the latter category.

STORES DATA

a. Univariate Analysis

1. Store Type

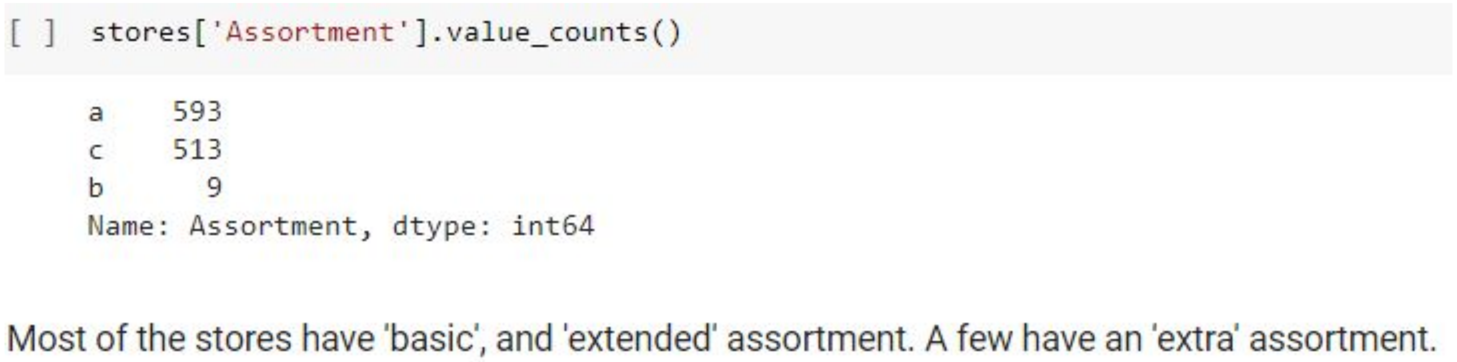


```
stores['StoreType'].value_counts()
```

```
a    602
d    348
c    148
b     17
Name: StoreType, dtype: int64
```

Most of the stores are of type 'a', followed by 'd', and 'c'. A few stores are of type 'b' also.

2. Assortment

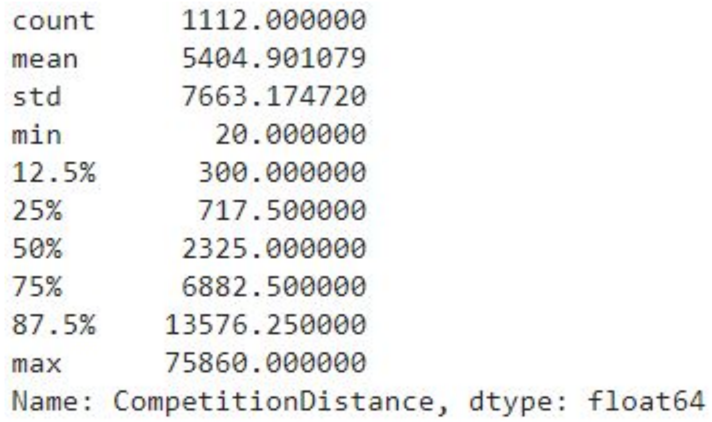


```
[ ]  stores['Assortment'].value_counts()
```

```
a    593
c    513
b      9
Name: Assortment, dtype: int64
```

Most of the stores have 'basic', and 'extended' assortment. A few have an 'extra' assortment.

3. Competition Distance

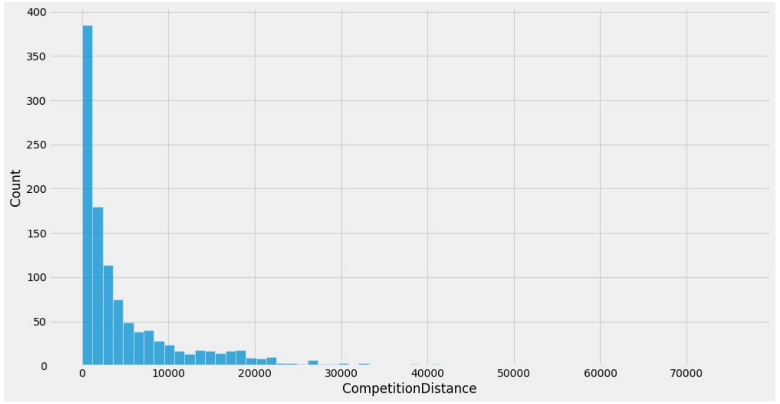


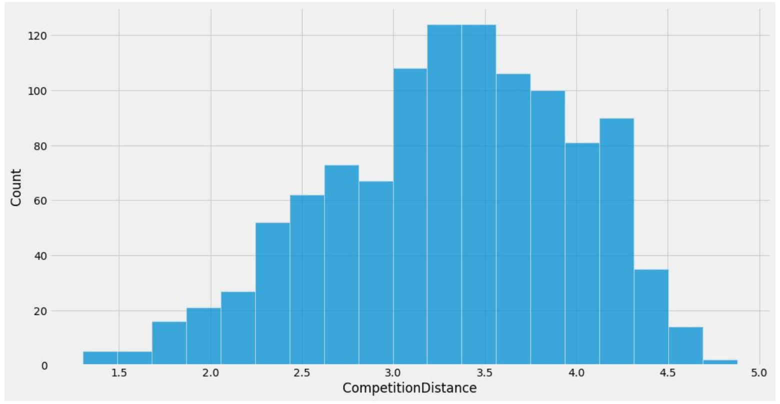
```
count      1112.000000
mean       5404.901079
std        7663.174720
min          20.000000
12.5%       300.000000
25%         717.500000
50%        2325.000000
75%        6882.500000
87.5%     13576.250000
max       75860.000000
Name: CompetitionDistance, dtype: float64
```

3 records in this variable are missing. We'll deal with those but first let's dwell on the other statistics. 50% of the competition distances is between 700 and 6880. The mean is greater than the median demonstrating a right skew in its distribution. The maximum distance until which a competitor has been identified is 75.8 kms.

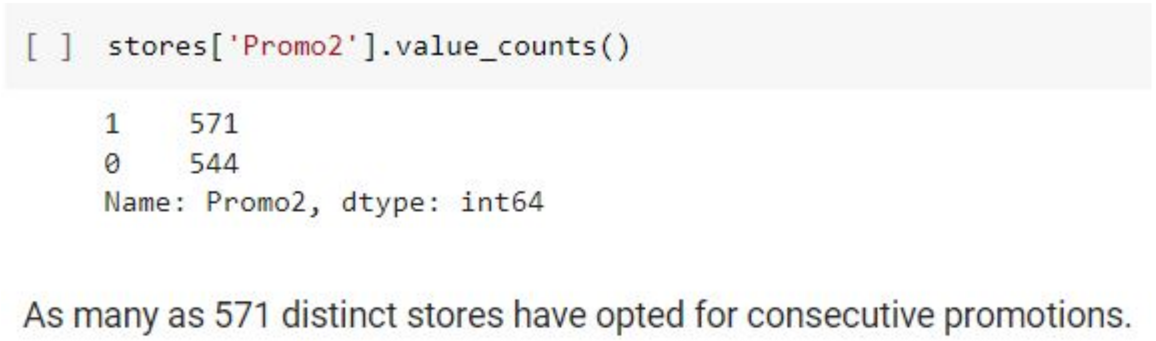
Without log transformation

With log transformation





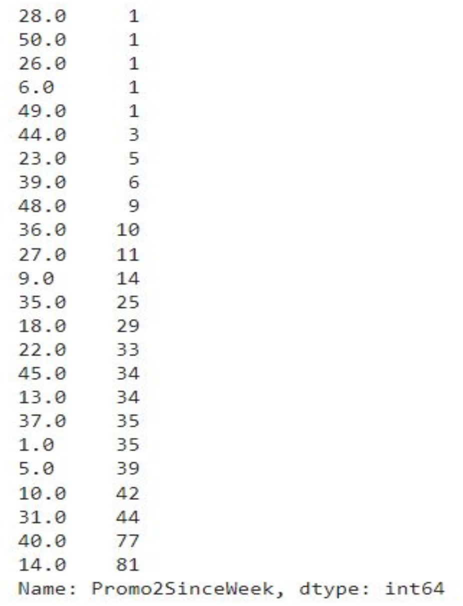
4. Promo2



```
[ ]  stores['Promo2'].value_counts()

     1    571
     0    544
     Name: Promo2, dtype: int64
```

As many as 571 distinct stores have opted for consecutive promotions.

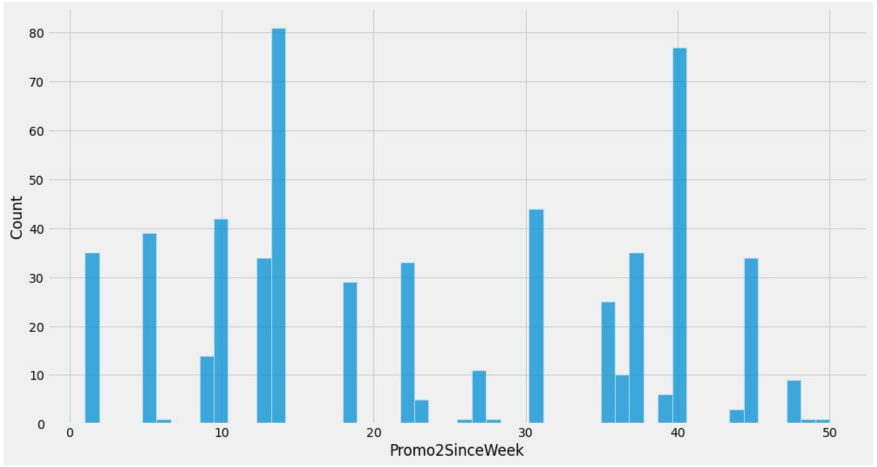
5. Promo2 Since Week



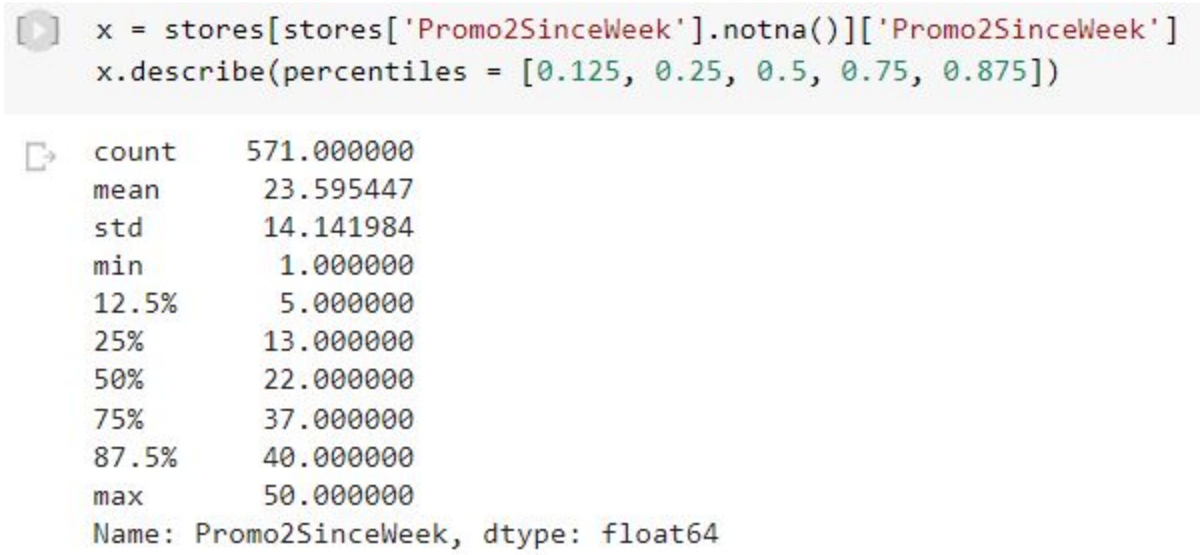
```
28.0     1
50.0     1
26.0     1
6.0      1
49.0     1
44.0     3
23.0     5
39.0     6
48.0     9
36.0    10
27.0    11
9.0     14
35.0    25
18.0    29
22.0    33
45.0    34
13.0    34
37.0    35
1.0     35
5.0     39
10.0    42
31.0    44
40.0    77
14.0    81
Name: Promo2SinceWeek, dtype: int64
```

For 544 stores, the data is not available for these stores are not participating in the consecutive promotions.

Most of these promotions began in the 14th week of the year. That is, towards the end of March or the beginning of April. Followed by the 40th week, that is in October, perhaps. Together, these two weeks account for more than 25% of these promotions.



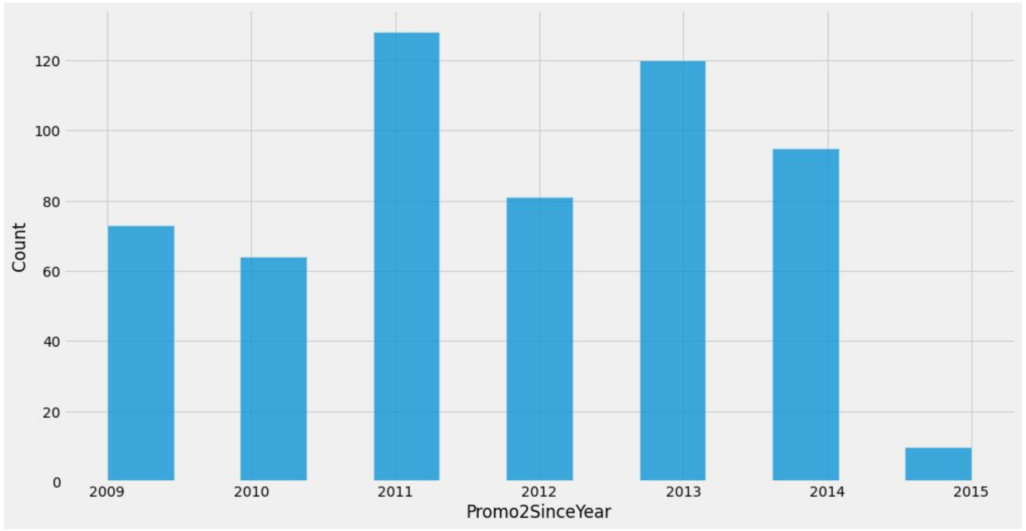
Apparently most of the 2nd promos begin either in the beginning of each months or towards its end. September seems to be an exception to this trend.


```
x = stores[stores['Promo2SinceWeek'].notna()]['Promo2SinceWeek']
x.describe(percentiles = [0.125, 0.25, 0.5, 0.75, 0.875])
```

```
count    571.000000
mean      23.595447
std       14.141984
min        1.000000
12.5%      5.000000
25%       13.000000
50%       22.000000
75%       37.000000
87.5%     40.000000
max       50.000000
Name: Promo2SinceWeek, dtype: float64
```

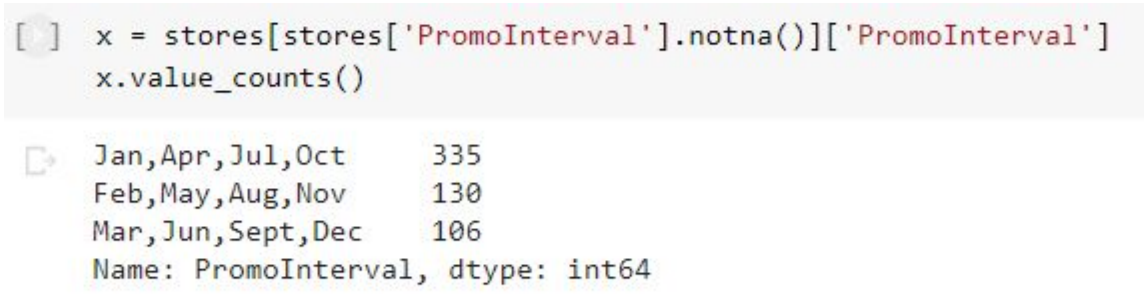
50% of the 2nd promotions began between the 13th and the 40th weeks.

6. Promo2 Since Year



50% of the 2nd promos have begun between 2011 and 2013, whereas 75% have started between 2009 and 2014.

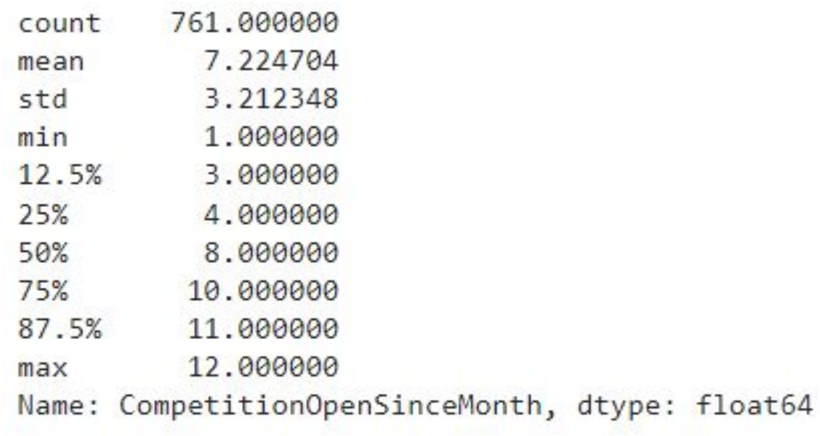
7. Promo Interval



```
x = stores[stores['PromoInterval'].notna()]['PromoInterval']
x.value_counts()
```

```
Jan,Apr,Jul,Oct     335
Feb,May,Aug,Nov     130
Mar,Jun,Sept,Dec    106
Name: PromoInterval, dtype: int64
```

8. Competition Open Since Month

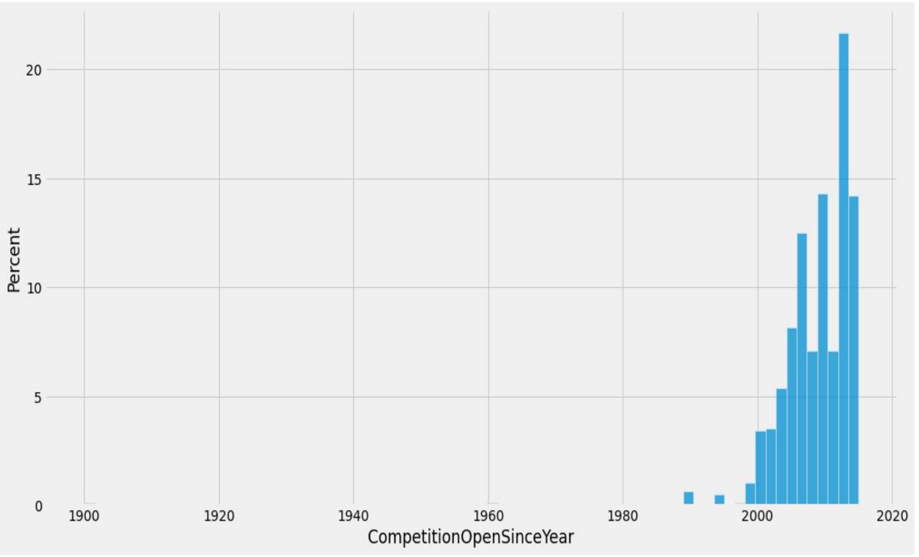


```
count     761.000000
mean        7.224704
std         3.212348
min         1.000000
12.5%       3.000000
25%         4.000000
50%         8.000000
75%        10.000000
87.5%      11.000000
max        12.000000
Name: CompetitionOpenSinceMonth, dtype: float64
```

On an average, the company may expect the competitors to have been established in the month of July or August.

50% of the competitors came about between April and October.

9. Competition Open Since Year



```
count      761.000000
mean      2008.668857
std          6.195983
min       1900.000000
12.5%     2004.000000
25%       2006.000000
50%       2010.000000
75%       2013.000000
87.5%     2014.000000
max       2015.000000
Name: CompetitionOpenSinceYear, dtype: float64
```

On an average, the company may expect the competitor to have been established between 2008 and 10. Whereas, 50% of the competitors have come about between 2006 and 14.

b. Dealing with Missing Observations

Variable with missing observation:

Competition Distance

Way of dealing

For competitor distance variable, it might have been the case that for the distance might not have been recorded when the competitor was more than 76 km away - not within this radius that it might affect the store's sales. The corresponding observations for the competition open month and year too are empty. This further emphasises our claim.

Therefore, it makes perfect sense to replace the missing values with 76K kms.

Variable with missing observation

Competition Open Since Month

Way of dealing

For as many as 354 stores, the approximate month in which the competitor was established is not known.

I don't think that any variable in this dataset could possibly explain the missingness in the variable in question. Perhaps, the stores, when being established, did not take into account the data of their competitors. I conclude - on my own - that this is the case of *Missing not at random.* And that - sans an educated method for imputation - this variable should better be removed.

Variable with missing observation

Competition Open Since Year

Way of dealing

Looks like the data for competition open since year is missing for all the records where the competition open since month too is missing.

It makes sense to drop this variable too.

Variable with missing observation

Promo2 since week/year and promo interval

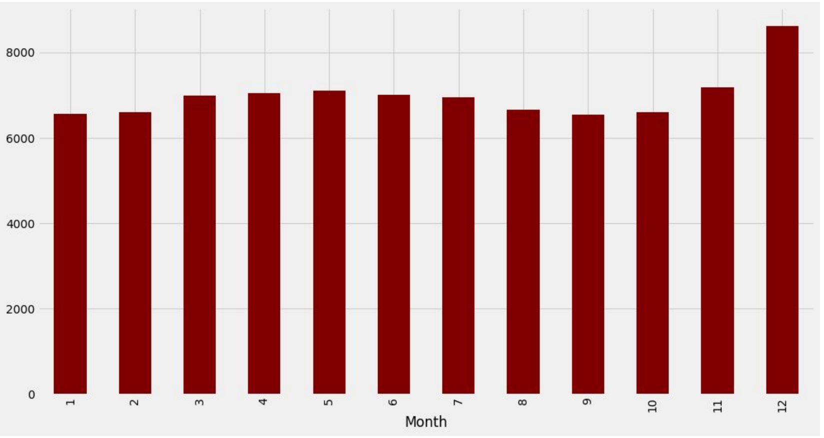
Way of dealing

For these 3 variables, all the variables corresponding to when the promo2 is 0 must be empty. For these variables do not make sense when the corresponding stores are not participating in the consecutive promos in the first place.

For the former to variables (promo since week / year), I'm substituting the nulls with 0s. However for the latter (promo interval), creating a separate category would make more sense.

c. Feature Engineering

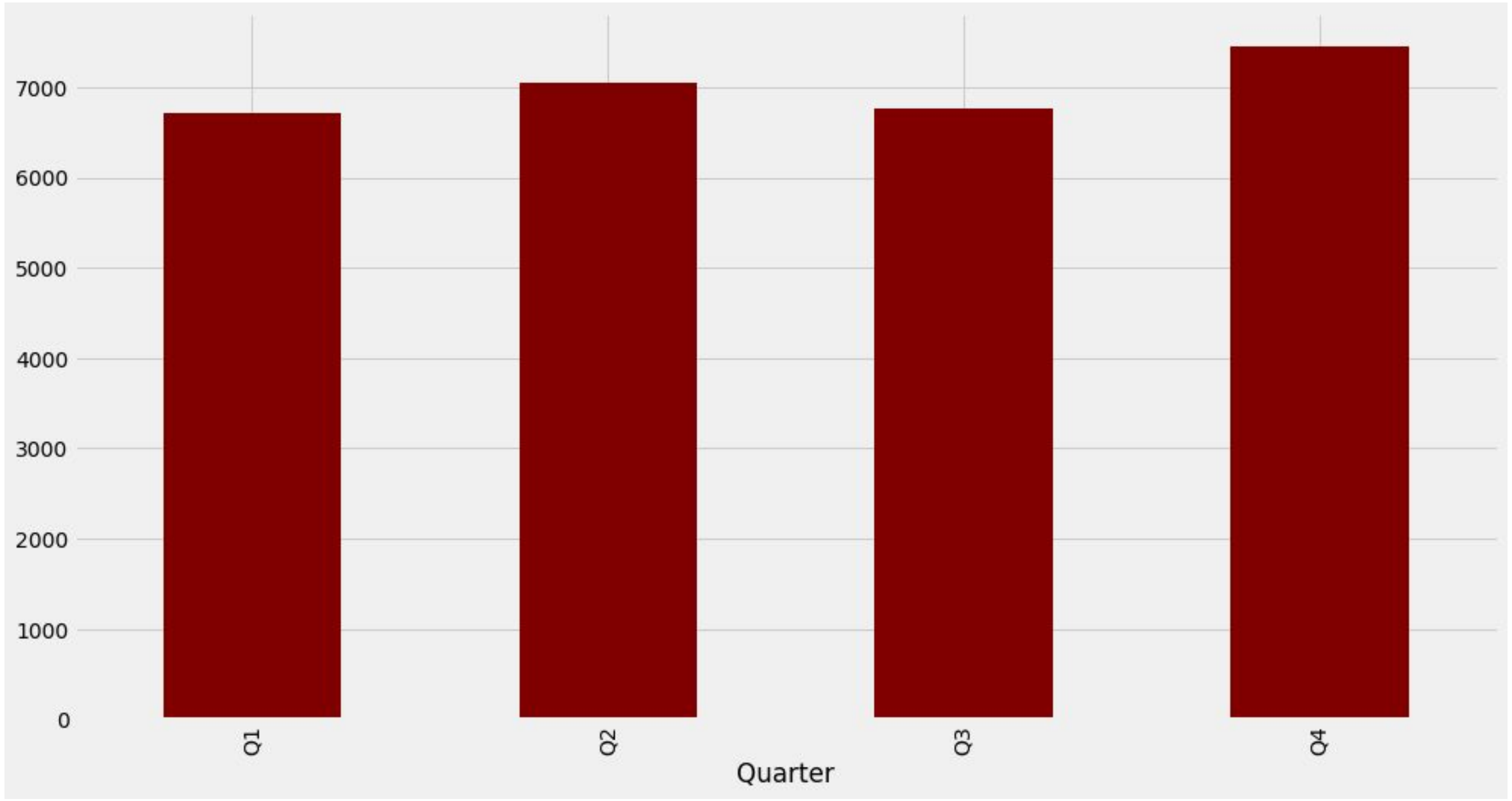
Months variable



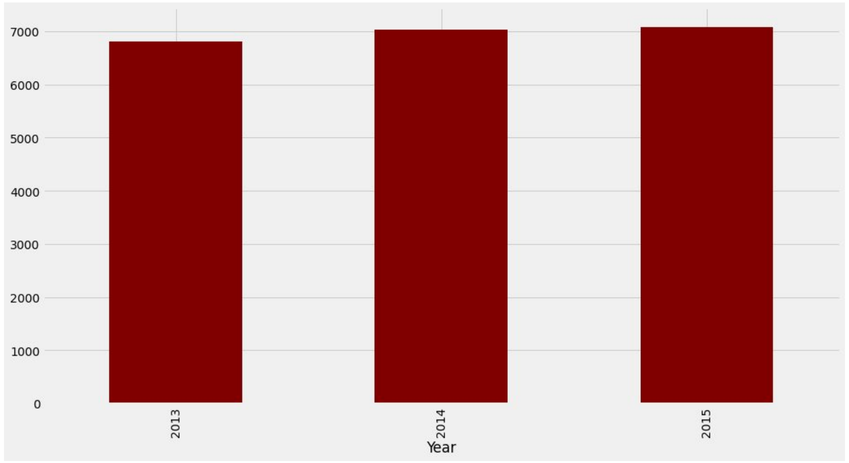
Apparently, the sales increase gradually from the start of the month until May from when it starts decreasing at the same pace making an even steep at the month of May. The sales decrease until October when it is almost at the same level as in January and from then increases at a higher rate until the end of the year when the mean sales cross the 8000 mark.

Fortunately, the data for each month has been given.

“Months” variable after binning



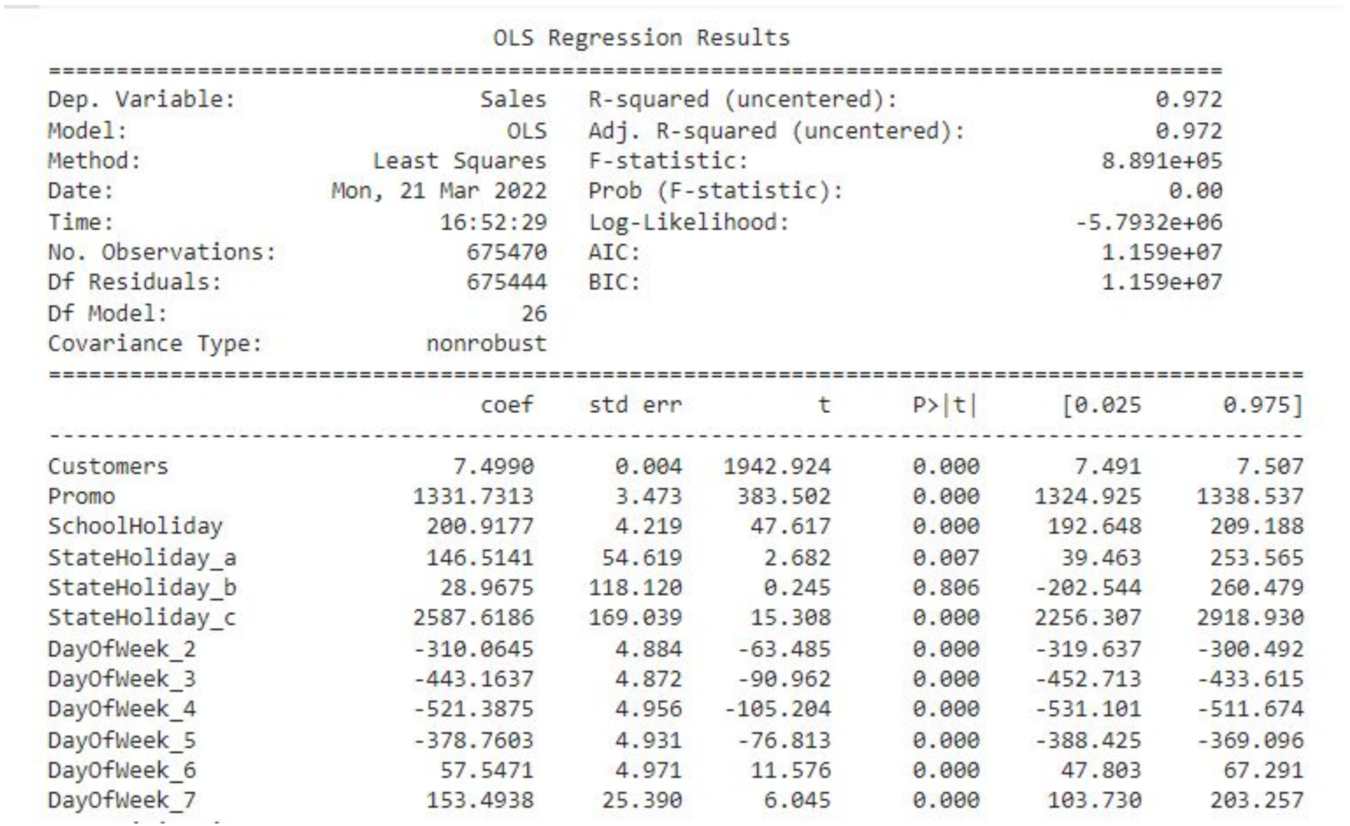
Year variable



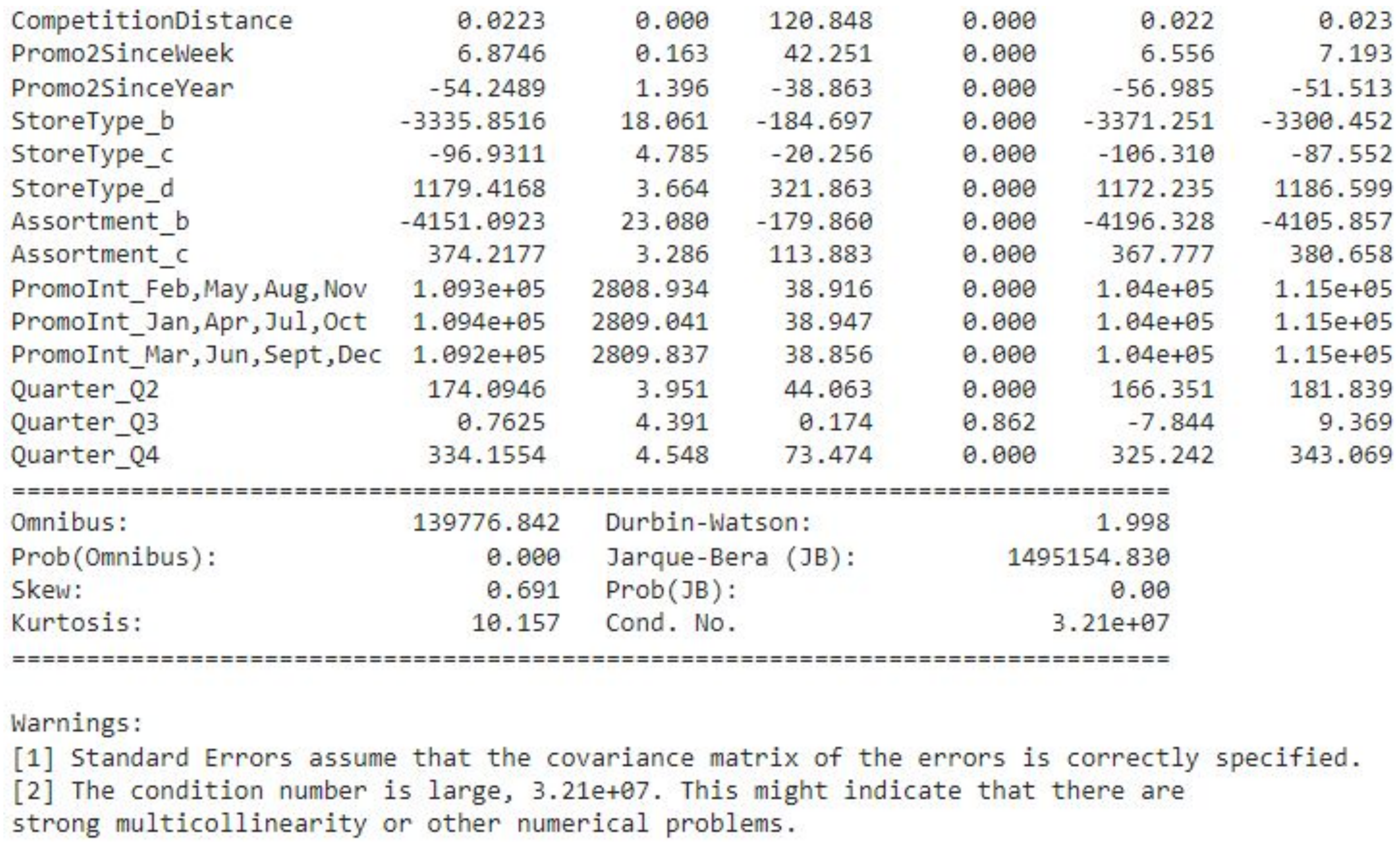
The mean sales has incremented each year, at a higher rate from 2013 to 14 and at a comparatively slower rate from 2014-15.

4. MACHINE LEARNING

a. Linear Regression

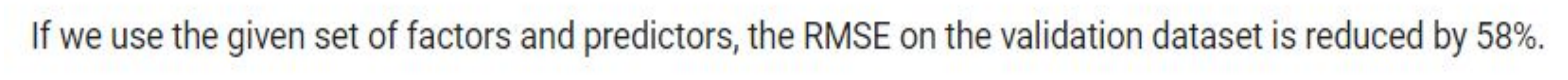


```
                            OLS Regression Results
=======================================================================================
Dep. Variable:                  Sales   R-squared (uncentered):                   0.972
Model:                            OLS   Adj. R-squared (uncentered):              0.972
Method:                 Least Squares   F-statistic:                          8.891e+05
Date:                Mon, 21 Mar 2022   Prob (F-statistic):                        0.00
Time:                        16:52:29   Log-Likelihood:                     -5.7932e+06
No. Observations:              675470   AIC:                                  1.159e+07
Df Residuals:                  675444   BIC:                                  1.159e+07
Df Model:                          26
Covariance Type:            nonrobust
=============================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------
Customers               7.4990      0.004   1942.924      0.000       7.491       7.507
Promo                1331.7313      3.473    383.502      0.000    1324.925    1338.537
SchoolHoliday         200.9177      4.219     47.617      0.000     192.648     209.188
StateHoliday_a        146.5141     54.619      2.682      0.007      39.463     253.565
StateHoliday_b         28.9675    118.120      0.245      0.806    -202.544     260.479
StateHoliday_c       2587.6186    169.039     15.308      0.000    2256.307    2918.930
DayOfWeek_2          -310.0645      4.884    -63.485      0.000    -319.637    -300.492
DayOfWeek_3          -443.1637      4.872    -90.962      0.000    -452.713    -433.615
DayOfWeek_4          -521.3875      4.956   -105.204      0.000    -531.101    -511.674
DayOfWeek_5          -378.7603      4.931    -76.813      0.000    -388.425    -369.096
DayOfWeek_6            57.5471      4.971     11.576      0.000      47.803      67.291
DayOfWeek_7           153.4938     25.390      6.045      0.000     103.730     203.257
```



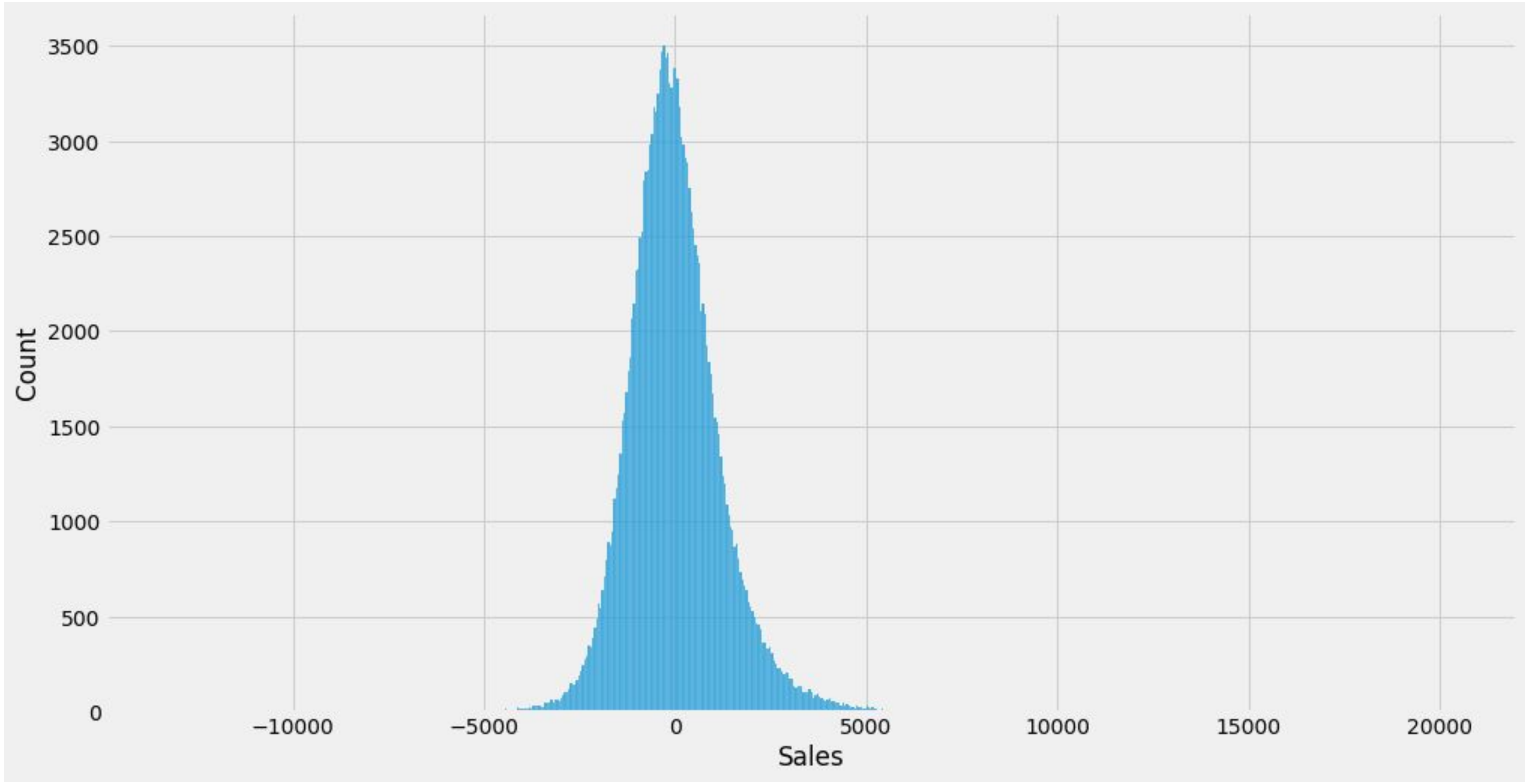
```
CompetitionDistance              0.0223       0.000     120.848      0.000       0.022       0.023
Promo2SinceWeek                  6.8746       0.163      42.251      0.000       6.556       7.193
Promo2SinceYear                -54.2489       1.396     -38.863      0.000     -56.985     -51.513
StoreType_b                  -3335.8516      18.061    -184.697      0.000   -3371.251   -3300.452
StoreType_c                    -96.9311       4.785     -20.256      0.000    -106.310     -87.552
StoreType_d                   1179.4168       3.664     321.863      0.000    1172.235    1186.599
Assortment_b                 -4151.0923      23.080    -179.860      0.000   -4196.328   -4105.857
Assortment_c                   374.2177       3.286     113.883      0.000     367.777     380.658
PromoInt_Feb,May,Aug,Nov      1.093e+05    2808.934      38.916      0.000    1.04e+05    1.15e+05
PromoInt_Jan,Apr,Jul,Oct      1.094e+05    2809.041      38.947      0.000    1.04e+05    1.15e+05
PromoInt_Mar,Jun,Sept,Dec     1.092e+05    2809.837      38.856      0.000    1.04e+05    1.15e+05
Quarter_Q2                     174.0946       3.951      44.063      0.000     166.351     181.839
Quarter_Q3                       0.7625       4.391       0.174      0.862      -7.844       9.369
Quarter_Q4                     334.1554       4.548      73.474      0.000     325.242     343.069
==============================================================================
Omnibus:                   139776.842   Durbin-Watson:                   1.998
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1495154.830
Skew:                           0.691   Prob(JB):                         0.00
Kurtosis:                      10.157   Cond. No.                     3.21e+07
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.21e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Reduction in RMSE



If we use the given set of factors and predictors, the RMSE on the validation dataset is reduced by 58%.

Distribution of Residuals



b. Decision Tree Regressor

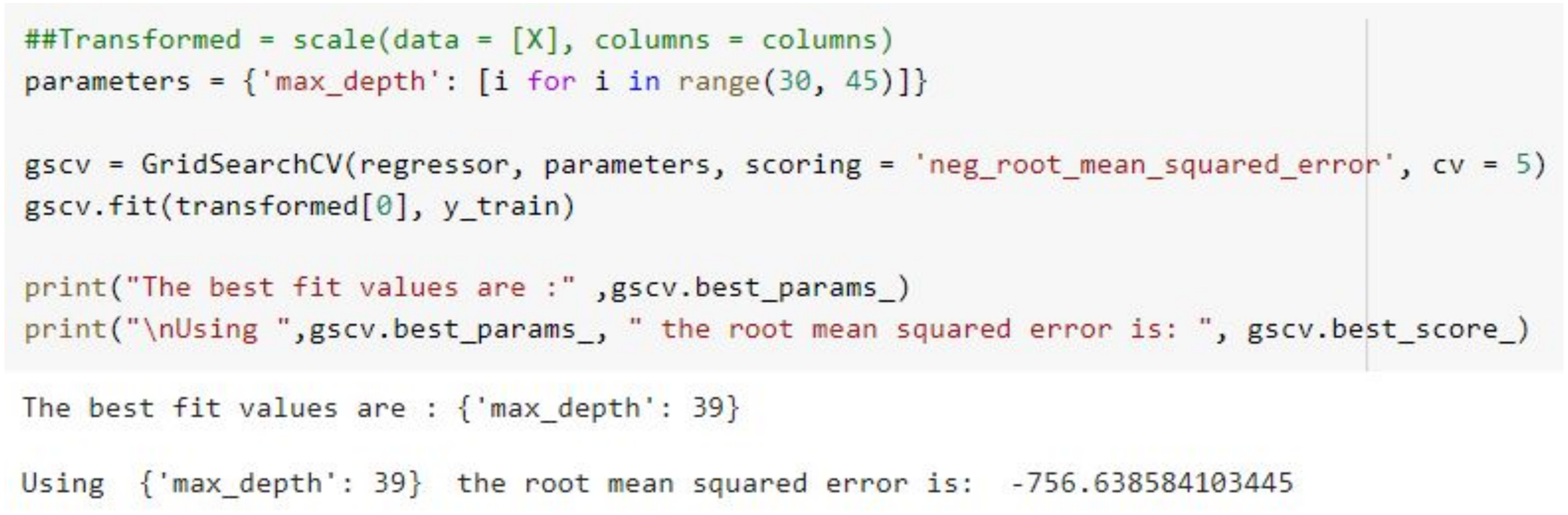
Reduction in RMSE

Using the Decision Tree Regressor, the RMSE is reduced by more than 70% which is better than our previous model (OLS).

Another advantage of the decision tree is that it gives 'discrete' values as outcome because of its very nature. Considering that we're dealing with the sales data which essentially is discrete, the outcome of decision tree is more interpretable.

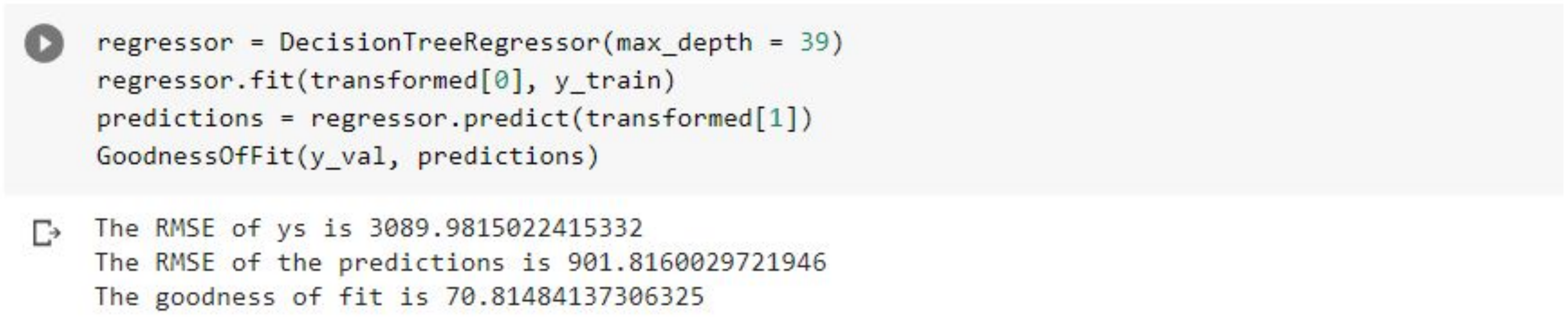
c. Hyperparameter Tuning

1. One hyperparameter of the Decision tree was tuned, namely “max_depth”.
2. Sklearn’s “GridSearchCV” package was employed for the purpose.
3. The code and the output is given below



```
##Transformed = scale(data = [X], columns = columns)
parameters = {'max_depth': [i for i in range(30, 45)]}

gscv = GridSearchCV(regressor, parameters, scoring = 'neg_root_mean_squared_error', cv = 5)
gscv.fit(transformed[0], y_train)

print("The best fit values are :" ,gscv.best_params_)
print("\nUsing ",gscv.best_params_, " the root mean squared error is: ", gscv.best_score_)
```

```
The best fit values are : {'max_depth': 39}

Using  {'max_depth': 39}  the root mean squared error is:  -756.638584103445
```



```python
regressor = DecisionTreeRegressor(max_depth = 39)
regressor.fit(transformed[0], y_train)
predictions = regressor.predict(transformed[1])
GoodnessOfFit(y_val, predictions)
```

```
The RMSE of ys is 3089.9815022415332
The RMSE of the predictions is 901.8160029721946
The goodness of fit is 70.81484137306325
```

Using the tuned value for the 'max_depth' hyperparameter, the RMSE of the predictions is slightly decreased.

THANK YOU!