WALMART

March 11, 2023

[2]:
```
#Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
```

[3]:
```
# Import Dataset

df = pd.read_csv('Walmart_Store_sales.csv')
```

[22]:
```
df.head()
```

[22]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106

[23]:
```
df.tail()
```

[23]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
6430	45	28-09-2012	713173.95	0	64.88	3.997		
6431	45	05-10-2012	733455.07	0	64.89	3.985		
6432	45	12-10-2012	734464.36	0	54.47	4.000		
6433	45	19-10-2012	718125.53	0	56.47	3.969		
6434	45	26-10-2012	760281.43	0	58.85	3.882		

```
6430  192.013558         8.684
6431  192.170412         8.667
6432  192.327265         8.667
6433  192.330854         8.667
6434  192.308899         8.667
```

[24]:
```
df.dtypes
```

[24]:
```
Store            int64
Date            object
Weekly_Sales   float64
Holiday_Flag     int64
Temperature    float64
Fuel_Price     float64
CPI            float64
Unemployment   float64
dtype: object
```

[25]:
```
df.shape
```

[25]:
```
(6435, 8)
```

[26]:
```
df.describe()
```

[26]:
```
              Store  Weekly_Sales  Holiday_Flag  Temperature   Fuel_Price  \
count  6435.000000  6.435000e+03   6435.000000  6435.000000  6435.000000
mean     23.000000  1.046965e+06      0.069930    60.663782     3.358607
std      12.988182  5.643666e+05      0.255049    18.444933     0.459020
min       1.000000  2.099862e+05      0.000000    -2.060000     2.472000
25%      12.000000  5.533501e+05      0.000000    47.460000     2.933000
50%      23.000000  9.607460e+05      0.000000    62.670000     3.445000
75%      34.000000  1.420159e+06      0.000000    74.940000     3.735000
max      45.000000  3.818686e+06      1.000000   100.140000     4.468000

                CPI  Unemployment
count  6435.000000   6435.000000
mean    171.578394      7.999151
std      39.356712      1.875885
min     126.064000      3.879000
25%     131.735000      6.891000
50%     182.616521      7.874000
75%     212.743293      8.622000
max     227.232807     14.313000
```

[27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 6435 entries, 0 to 6434
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Store         6435 non-null   int64
 1   Date          6435 non-null   object
 2   Weekly_Sales  6435 non-null   float64
 3   Holiday_Flag  6435 non-null   int64
 4   Temperature   6435 non-null   float64
 5   Fuel_Price    6435 non-null   float64
 6   CPI           6435 non-null   float64
 7   Unemployment  6435 non-null   float64
dtypes: float64(5), int64(2), object(1)
memory usage: 402.3+ KB
```

[5]:
```
#Changing the date datatype to datetime64
#Change dates into days by creating new variable

from datetime import datetime
df['Date'] = pd.to_datetime(df['Date'])
```

[29]:
```
df.dtypes
```

[29]:
```
Store                    int64
Date            datetime64[ns]
Weekly_Sales           float64
Holiday_Flag             int64
Temperature            float64
Fuel_Price             float64
CPI                    float64
Unemployment           float64
dtype: object
```

[6]:
```
# Checking Null Values
df.isnull().sum()
```

[6]:
```
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
dtype: int64
```

0.1 Find a Store with Max and Min Sales

[31]:
```
df.head()
```

[31]:
```
   Store       Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
0      1 2010-05-02    1643690.90             0        42.31       2.572
1      1 2010-12-02    1641957.44             1        38.51       2.548
2      1 2010-02-19    1611968.17             0        39.93       2.514
3      1 2010-02-26    1409727.59             0        46.63       2.561
4      1 2010-05-03    1554806.68             0        46.50       2.625

          CPI  Unemployment
0  211.096358         8.106
1  211.242170         8.106
2  211.289143         8.106
3  211.319643         8.106
4  211.350143         8.106
```

[10]:
```
df1 = df['Store'].unique()
df1
```

[10]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45])
```

[11]:
```
total_sales = df.groupby('Store')['Weekly_Sales'].sum().round().
 ↪sort_values(ascending = False)
```

[34]:
```
total_sales
```

[34]:
```
Store
20    301397792.0
4     299543953.0
14    288999911.0
13    286517704.0
2     275382441.0
10    271617714.0
27    253855917.0
6     223756131.0
1     222402809.0
39    207445542.0
19    206634862.0
31    199613905.0
23    198750618.0
24    194016021.0
11    193962787.0
28    189263681.0
```

```
41    181341935.0
32    166819246.0
18    155114734.0
22    147075649.0
12    144287230.0
26    143416394.0
34    138249763.0
40    137870310.0
35    131520672.0
8     129951181.0
17    127782139.0
45    112395341.0
21    108117879.0
25    101061179.0
43     90565435.0
15     89133684.0
7      81598275.0
42     79565752.0
9      77789219.0
29     77141554.0
16     74252425.0
37     74202740.0
30     62716885.0
3      57586735.0
38     55159626.0
36     53412215.0
5      45475689.0
44     43293088.0
33     37160222.0
Name: Weekly_Sales, dtype: float64
```

[12]:
```
# Maximum Sales
pd.DataFrame(total_sales).head(1)
```

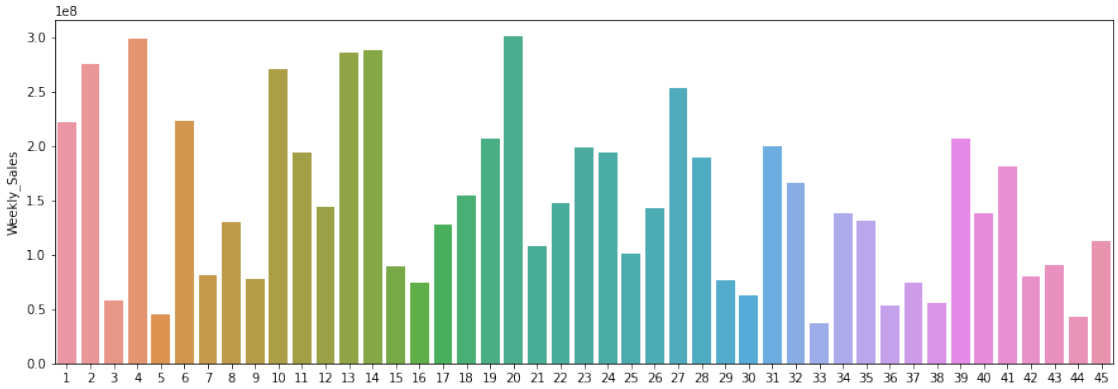
[12]:
```
       Weekly_Sales
Store
20      301397792.0
```

[36]:
```
# Minimum Sales
pd.DataFrame(total_sales).tail(1)
```

[36]:
```
       Weekly_Sales
Store
33       37160222.0
```

[37]:
```
t = total_sales.to_frame()
t1 = t.sort_values(by=['Store'])
```

[38]:
```
plt.figure(figsize=(15,5))
sns.barplot(x=df1, y=t1['Weekly_Sales'])
plt.savefig('p1.png')
```



[39]:
```
df.Weekly_Sales
```

[39]:
```
0       1643690.90
1       1641957.44
2       1611968.17
3       1409727.59
4       1554806.68
           ...
6430     713173.95
6431     733455.07
6432     734464.36
6433     718125.53
6434     760281.43
Name: Weekly_Sales, Length: 6435, dtype: float64
```

1 Store - 20 has a maximum sales = $301397792

Store - 33 has a minimum sales = $37160222

** Find the store with maximum standard deviation also find the coefficient of mean to standard deviation.

[40]:
```
df.head()
```

[40]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	\
0	1	2010-05-02	1643690.90	0	42.31	2.572	
1	1	2010-12-02	1641957.44	1	38.51	2.548	
2	1	2010-02-19	1611968.17	0	39.93	2.514	

```
3      1 2010-02-26    1409727.59             0        46.63       2.561
4      1 2010-05-03    1554806.68             0        46.50       2.625

           CPI  Unemployment
0  211.096358         8.106
1  211.242170         8.106
2  211.289143         8.106
3  211.319643         8.106
4  211.350143         8.106
```

[13]:
```
df_std = df.groupby('Store')['Weekly_Sales'].std().round().
 ↪sort_values(ascending=False)
```

[14]:
```
pd.DataFrame(df_std).head()
```

[14]:
```
       Weekly_Sales
Store
14         317570.0
10         302262.0
20         275901.0
4          266201.0
13         265507.0
```

Store - 14 has a maximum standard deviation = $317569.949

[15]:
```
store14 = df[df.Store == 14].Weekly_Sales
```

[16]:
```
mean_to_stddev = store14.std()/store14.mean()*100
```

[17]:
```
#mean_to_stddev.round(2)
mean_to_stddev
```

[17]:
```
15.713673600948338
```

Mean to Standard Deviation = 15.71%

** Which Store has good quarterly growth rate in Q3 2012?**

[18]:
```
q2_sales = df[(df['Date'] >= '2012-04-01') & (df['Date'] <= '2012-06-30')].
 ↪groupby('Store')['Weekly_Sales'].sum().round()
```

[19]:
```
q3_sales = df[(df['Date'] >= '2012-07-01') & (df['Date'] <= '2012-09-30')].
 ↪groupby('Store')['Weekly_Sales'].sum().round()
```

[48]:
```
q2_sales
```

[48]:
```
Store
1     21036966.0
```

```
2     25085124.0
3      5562668.0
4     28384185.0
5      4427262.0
6     20728970.0
7      7613594.0
8     11934276.0
9      7431320.0
10    23598434.0
11    17879096.0
12    13193365.0
13    26803226.0
14    24427769.0
15     7867952.0
16     6626133.0
17    12918892.0
18    13834706.0
19    18315279.0
20    27550181.0
21     9226280.0
22    13329065.0
23    18283425.0
24    17768192.0
25     9247467.0
26    13218290.0
27    22593641.0
28    16986000.0
29     7034493.0
30     5786335.0
31    18249155.0
32    15415236.0
33     3512138.0
34    12858028.0
35    10753571.0
36     4090379.0
37     6859778.0
38     5732363.0
39    20191586.0
40    12849747.0
41    17560036.0
42     7608247.0
43     8239793.0
44     4322555.0
45    10278900.0
Name: Weekly_Sales, dtype: float64
```

[49]:
```
q3_sales
```

[49]:
```
Store
1     18633210.0
2     22396868.0
3      4966496.0
4     25652119.0
5      3880622.0
6     18341221.0
7      7322394.0
8     10873860.0
9      6528240.0
10    21169356.0
11    16094363.0
12    11777508.0
13    24319994.0
14    20140430.0
15     6909374.0
16     6441311.0
17    11533998.0
18    12507522.0
19    16644341.0
20    24665938.0
21     8403508.0
22    11818544.0
23    17103654.0
24    16126000.0
25     8309440.0
26    12417575.0
27    20191238.0
28    15055660.0
29     6127862.0
30     5181974.0
31    16454328.0
32    14142165.0
33     3177072.0
34    11476259.0
35    10252123.0
36     3578124.0
37     6250524.0
38     5129298.0
39    18899955.0
40    11647661.0
41    16373588.0
42     6830840.0
43     7376726.0
44     4020486.0
45     8851242.0
Name: Weekly_Sales, dtype: float64
```

[20]:
```
pd.DataFrame({'Q2 Sales' : q2_sales,
              'Q3 Sales' : q3_sales,
              'Difference' : (q3_sales - q2_sales),
              'Growth Rate' : (q3_sales - q2_sales)/q3_sales*100}).
↪sort_values(by=['Store'])
```

[20]:

Store	Q2 Sales	Q3 Sales	Difference	Growth Rate
1	21036966.0	18633210.0	-2403756.0	-12.900386
2	25085124.0	22396868.0	-2688256.0	-12.002821
3	5562668.0	4966496.0	-596172.0	-12.003876
4	28384185.0	25652119.0	-2732066.0	-10.650450
5	4427262.0	3880622.0	-546640.0	-14.086402
6	20728970.0	18341221.0	-2387749.0	-13.018484
7	7613594.0	7322394.0	-291200.0	-3.976841
8	11934276.0	10873860.0	-1060416.0	-9.751974
9	7431320.0	6528240.0	-903080.0	-13.833437
10	23598434.0	21169356.0	-2429078.0	-11.474501
11	17879096.0	16094363.0	-1784733.0	-11.089181
12	13193365.0	11777508.0	-1415857.0	-12.021703
13	26803226.0	24319994.0	-2483232.0	-10.210660
14	24427769.0	20140430.0	-4287339.0	-21.287227
15	7867952.0	6909374.0	-958578.0	-13.873587
16	6626133.0	6441311.0	-184822.0	-2.869323
17	12918892.0	11533998.0	-1384894.0	-12.007059
18	13834706.0	12507522.0	-1327184.0	-10.611087
19	18315279.0	16644341.0	-1670938.0	-10.039076
20	27550181.0	24665938.0	-2884243.0	-11.693222
21	9226280.0	8403508.0	-822772.0	-9.790816
22	13329065.0	11818544.0	-1510521.0	-12.780940
23	18283425.0	17103654.0	-1179771.0	-6.897772
24	17768192.0	16126000.0	-1642192.0	-10.183505
25	9247467.0	8309440.0	-938027.0	-11.288691
26	13218290.0	12417575.0	-800715.0	-6.448240
27	22593641.0	20191238.0	-2402403.0	-11.898245
28	16986000.0	15055660.0	-1930340.0	-12.821358
29	7034493.0	6127862.0	-906631.0	-14.795225
30	5786335.0	5181974.0	-604361.0	-11.662756
31	18249155.0	16454328.0	-1794827.0	-10.907933
32	15415236.0	14142165.0	-1273071.0	-9.001953
33	3512138.0	3177072.0	-335066.0	-10.546377
34	12858028.0	11476259.0	-1381769.0	-12.040239
35	10753571.0	10252123.0	-501448.0	-4.891163
36	4090379.0	3578124.0	-512255.0	-14.316301
37	6859778.0	6250524.0	-609254.0	-9.747247
38	5732363.0	5129298.0	-603065.0	-11.757262
39	20191586.0	18899955.0	-1291631.0	-6.834043

```
40     12849747.0  11647661.0  -1202086.0   -10.320407
41     17560036.0  16373588.0  -1186448.0    -7.246109
42      7608247.0   6830840.0   -777407.0   -11.380840
43      8239793.0   7376726.0   -863067.0   -11.699865
44      4322555.0   4020486.0   -302069.0    -7.513246
45     10278900.0   8851242.0  -1427658.0   -16.129465
```

No Store has shown growth in Q3 2012

Some holidays have a negative impact on sales. Find out holidays which have higher sales than the mean sales in non-holiday season for all stores together.

[22]:
```
#Holiday Events

Super_Bowl = ['12-2-2010', '11-2-2011', '10-2-2012']
Labour_Day = ['10-9-2010', '9-9-2011', '7-9-2012']
Thanksgiving = ['26-11-2010', '25-11-2011', '23-11-2012']
Christmas = ['31-12-2010', '30-12-2011', '28-12-2012']
```

[25]:
```
# Calculating Holiday Event Sales

Super_Bowl_sales = df.loc[df.Date.isin(Super_Bowl)]['Weekly_Sales'].mean()
Labour_Day_sales = df.loc[df.Date.isin(Labour_Day)]['Weekly_Sales'].mean()
Thanksgiving_sales = df.loc[df.Date.isin(Thanksgiving)]['Weekly_Sales'].mean()
Christmas_sales = df.loc[df.Date.isin(Christmas)]['Weekly_Sales'].mean()
```

[26]:
```
Super_Bowl_sales, Labour_Day_sales, Thanksgiving_sales, Christmas_sales
```

[26]:
```
(1079127.9877037033, 1042427.2939259257, 1471273.427777778, 960833.1115555551)
```

[27]:
```
# Calculate the vale of mean on non-holidays

df.head()
```

[27]:
```
   Store       Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
0      1 2010-05-02    1643690.90             0        42.31       2.572
1      1 2010-12-02    1641957.44             1        38.51       2.548
2      1 2010-02-19    1611968.17             0        39.93       2.514
3      1 2010-02-26    1409727.59             0        46.63       2.561
4      1 2010-05-03    1554806.68             0        46.50       2.625

           CPI  Unemployment
0  211.096358         8.106
1  211.242170         8.106
2  211.289143         8.106
3  211.319643         8.106
4  211.350143         8.106
```

[29]:
```
non_holiday_sales = df[(df['Holiday_Flag'] == 0)]['Weekly_Sales'].mean()
```

[30]:
```
non_holiday_sales
```

[30]:
```
1041256.3802088564
```

[31]:
```
result = pd.DataFrame([{'Super Bowl Sales' : Super_Bowl_sales,
                        'Labour Day Sales' : Labour_Day_sales,
                        'Thanksgiving Sales' : Thanksgiving_sales,
                        'Christmas Sales' : Christmas_sales,
                        'Non Holiday Sales' : non_holiday_sales}]).T
```

[32]:
```
result
```

[32]:
```
                              0
Super Bowl Sales    1.079128e+06
Labour Day Sales    1.042427e+06
Thanksgiving Sales  1.471273e+06
Christmas Sales     9.608331e+05
Non Holiday Sales   1.041256e+06
```

Thanksgiving, Superbowl, Labour Day sales has the sales > Non holiday Sales

** Display monthly and semester view of sales in units and give insights **

[33]:
```
df['Day'] = pd.DatetimeIndex(df['Date']).day
df['Month'] = pd.DatetimeIndex(df['Date']).month
df['Year'] = pd.DatetimeIndex(df['Date']).year
```

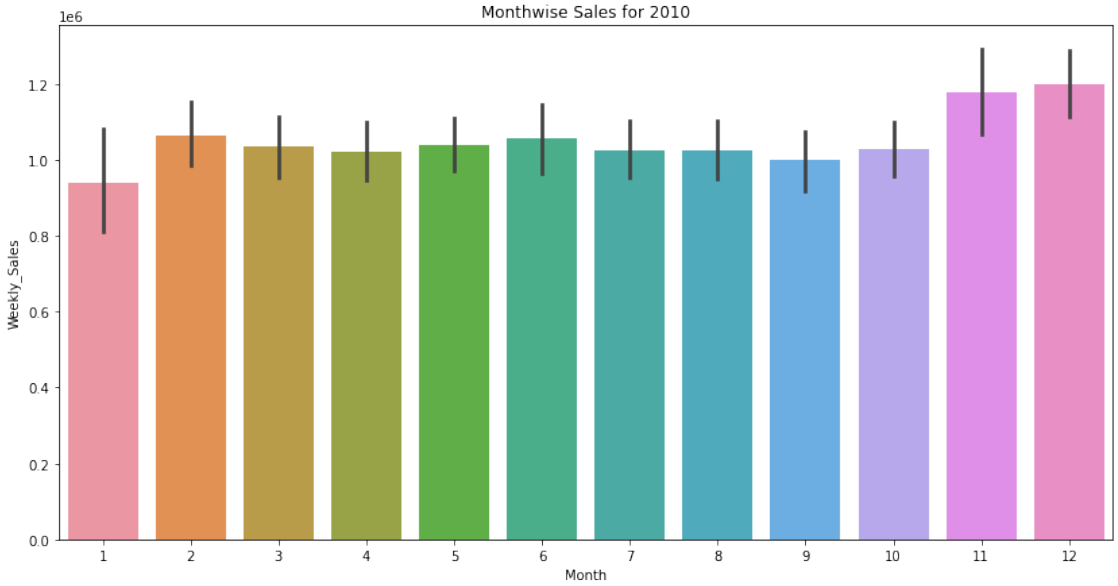
[34]:
```
df['Year'].unique()
```

[34]:
```
array([2010, 2011, 2012])
```

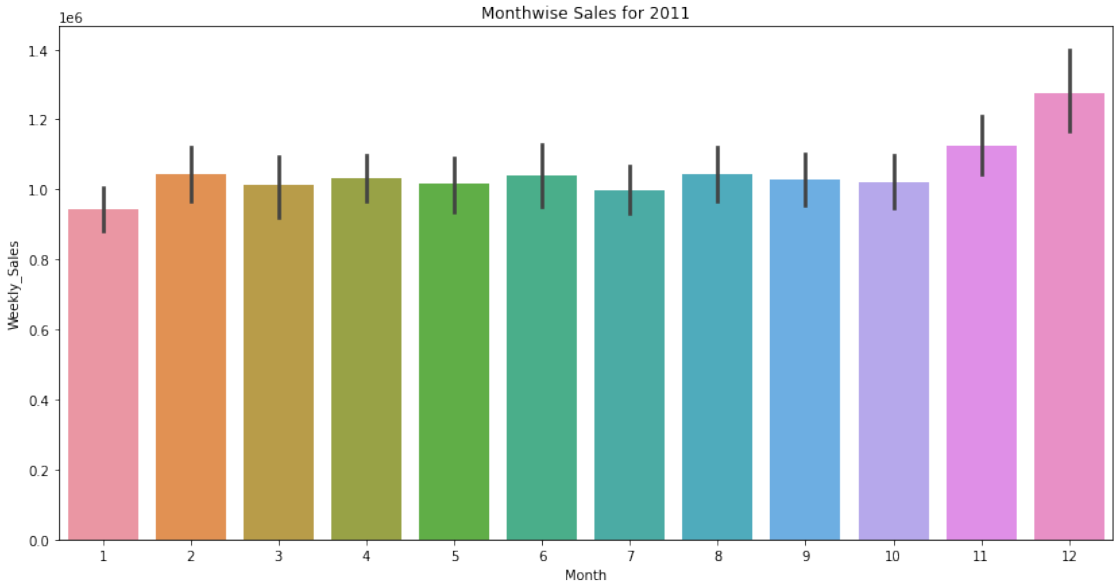
[35]:
```
df.head()
```

[35]:
```
   Store       Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
0      1 2010-05-02    1643690.90             0        42.31       2.572
1      1 2010-12-02    1641957.44             1        38.51       2.548
2      1 2010-02-19    1611968.17             0        39.93       2.514
3      1 2010-02-26    1409727.59             0        46.63       2.561
4      1 2010-05-03    1554806.68             0        46.50       2.625

          CPI  Unemployment  Day  Month  Year
0  211.096358         8.106    2      5  2010
1  211.242170         8.106    2     12  2010
2  211.289143         8.106   19      2  2010
3  211.319643         8.106   26      2  2010
4  211.350143         8.106    3      5  2010
```

[36]:
```
# sales for the year - 2010

plt.figure(figsize=(14,7))
graph1 = sns.barplot(data=df, x=df[df.Year==2010]['Month'], y=df[df.
 ↪Year==2010]['Weekly_Sales'])
graph1.set(title='Monthwise Sales for 2010')

plt.savefig('2010.png')
```



[37]:
```
# Sales for the Year 2011

plt.figure(figsize=(14,7))
graph1 = sns.barplot(data=df, x=df[df.Year==2011]['Month'], y=df[df.
 ↪Year==2011]['Weekly_Sales'])
graph1.set(title='Monthwise Sales for 2011')

plt.savefig('2011.png')
```

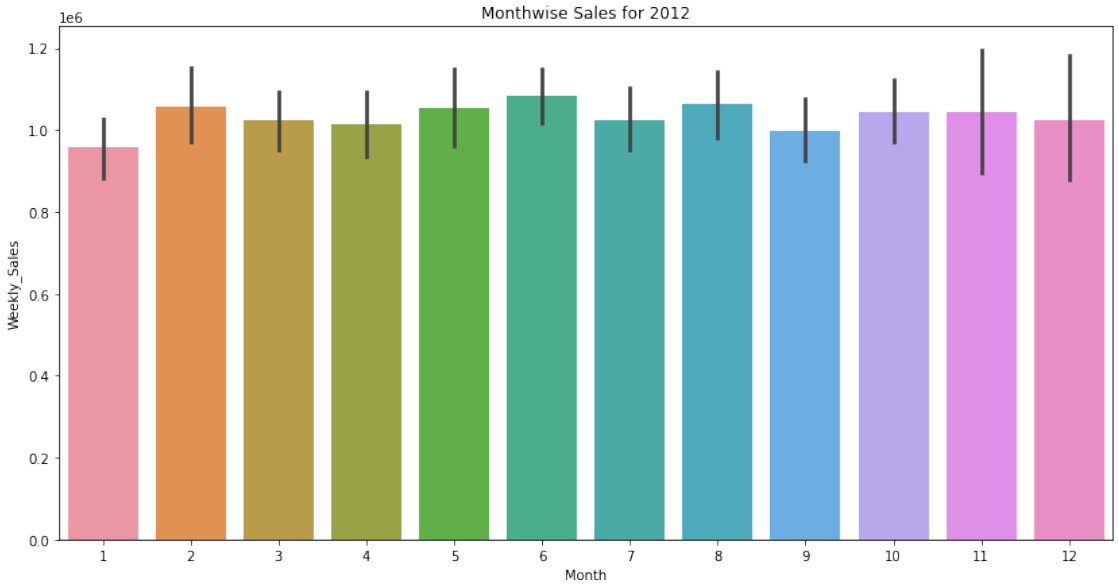


[38]:
```
graph1.patches
```

[38]: <Axes.ArtistList of 12 patches>

[40]:
```
# Sales for the Year 2012

plt.figure(figsize=(14,7))
graph1 = sns.barplot(data=df, x=df[df.Year==2012]['Month'], y=df[df.
 ↪Year==2012]['Weekly_Sales'])
graph1.set(title='Monthwise Sales for 2012')
```

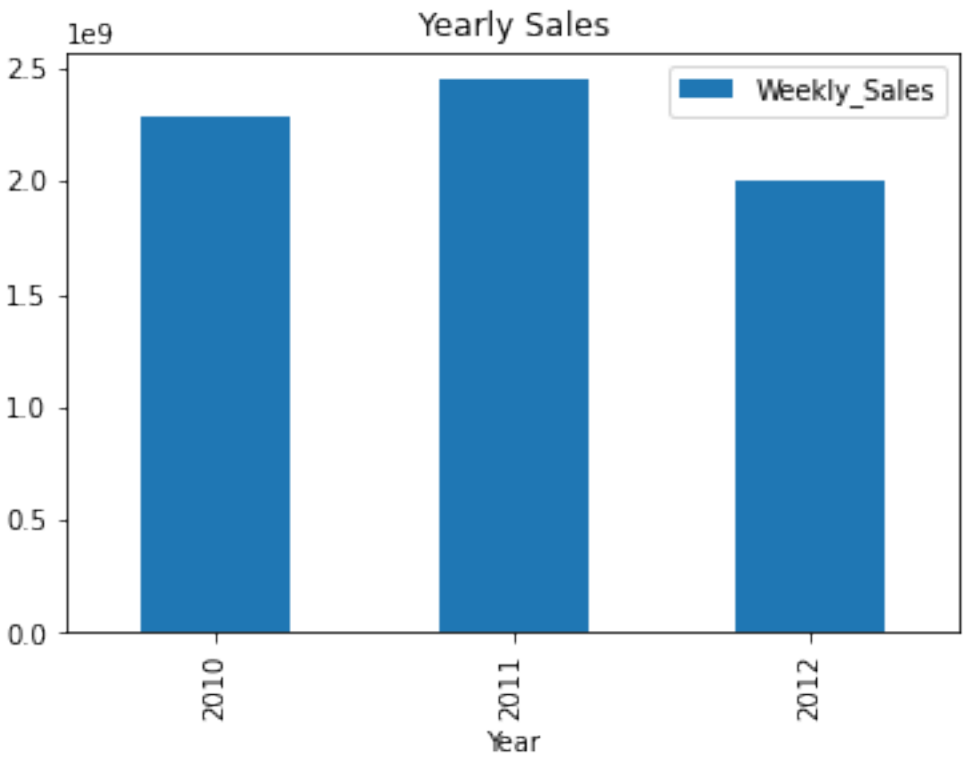
[40]: [Text(0.5, 1.0, 'Monthwise Sales for 2012')]



[42]:
```
# Yearly Sales
plt.figure(figsize=(10,7))
df.groupby('Year')[['Weekly_Sales']].sum().plot(kind='bar')
plt.title('Yearly Sales')
```

[42]: Text(0.5, 1.0, 'Yearly Sales')

<Figure size 720x504 with 0 Axes>



2 Model Building

[8]:
```
# Define Independent and dependent variable

x = df[['Store', 'Fuel_Price', 'CPI', 'Unemployment']]  #independent variable
y = df['Weekly_Sales'] #dependent variable
df.head()
```

[8]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	\
0	1	05-02-2010	1643690.90	0	42.31	2.572	
1	1	12-02-2010	1641957.44	1	38.51	2.548	
2	1	19-02-2010	1611968.17	0	39.93	2.514	
3	1	26-02-2010	1409727.59	0	46.63	2.561	
4	1	05-03-2010	1554806.68	0	46.50	2.625	

	CPI	Unemployment
0	211.096358	8.106
1	211.242170	8.106
2	211.289143	8.106
3	211.319643	8.106

```
4  211.350143         8.106
```

[5]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.2)
```

[7]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

3 Linear Regression

[11]:
```
from sklearn.linear_model import LinearRegression
from sklearn import metrics

lr = LinearRegression()
lr.fit(x_train, y_train)
lr_y_pred = lr.predict(x_test)

print('* Linear Regression *\n')

print('Mean Absolute Error: ', metrics.mean_absolute_error(y_test, lr_y_pred).
 ↪round(3))
print('Mean Squared Error: ', metrics.mean_squared_error(y_test, lr_y_pred).
 ↪round(3))
print('Root Mean Sqsuared Error: ', np.sqrt(metrics.mean_squared_error(y_test,␣
 ↪lr_y_pred)).round(3))
```

```
* Linear Regression *

Mean Absolute Error:  431970.173
Mean Squared Error:  285405043020.282
Root Mean Sqsuared Error:  534233.135
```

4 Conclusion

Here we have used Linear Regression to predict the weekly sales and got above result

5 Change dates into days by creating new variable

[13]:
```
df['Day1'] = pd.to_datetime(df['Date']).dt.day_name()
```

[14]:
```
df.head()
```

[14]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	\
0	1	05-02-2010	1643690.90	0	42.31	2.572	
1	1	12-02-2010	1641957.44	1	38.51	2.548	
2	1	19-02-2010	1611968.17	0	39.93	2.514	
3	1	26-02-2010	1409727.59	0	46.63	2.561	
4	1	05-03-2010	1554806.68	0	46.50	2.625	

	CPI	Unemployment	Day1
0	211.096358	8.106	Sunday
1	211.242170	8.106	Thursday
2	211.289143	8.106	Friday
3	211.319643	8.106	Friday
4	211.350143	8.106	Monday

[ ]:
```

```