Walmart

Data Science Methodology Project

TEAM MEMBERS:

MAHMOUD ESSAM FATHY GABR: 20221460231
WESSAM FATHY MASOUD OMAR: 20221451271
NOURHAN MUHAMMED FATHY ABDELAAL: 20221452411

Walmart Main Details:

- Walmart runs several promotional markdown events throughout the year. These markdowns precede prominent holidays, the four largest of all, which are the Super Bowl, Labour Day, Thanksgiving, and Christmas. The weeks including these holidays are weighted five times higher in the evaluation than non-holiday weeks. Part of the challenge presented by this assignment is modeling the effects of markdowns on these holiday weeks in the absence of complete/ideal historical data. Historical sales data for 45 Walmart stores located in different regions are available.

Walmart focus

Walmart focus sure is how to increase Revenue based on her historical data that covers sales from 2010-02-05 to 2012-11-01

As her attributes were

• Store •

• Date •

• Weekly Sales •

• Holiday Flag (0 for work, one for special holiday) •

• Temperature •

• Fuel Price •

• CPI •

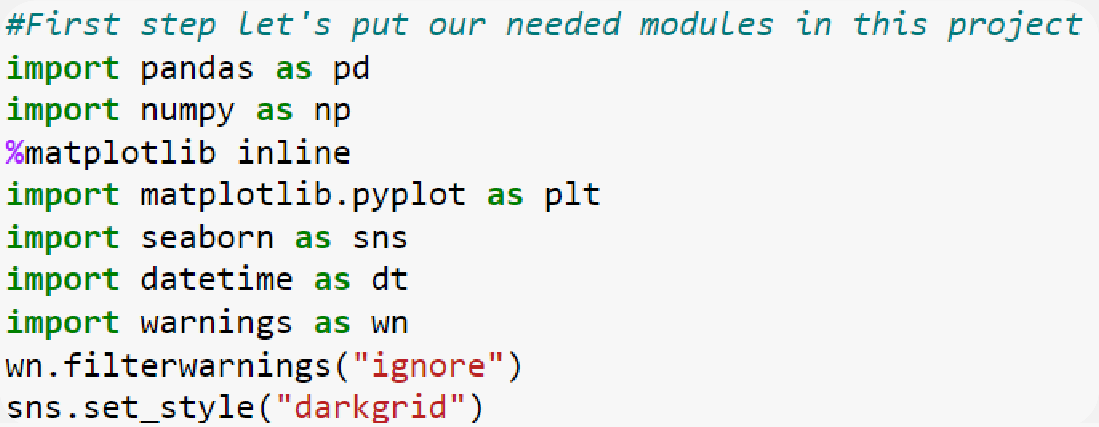
• Unemployment •

To solve Walmart Problem, we had to solve some questions:

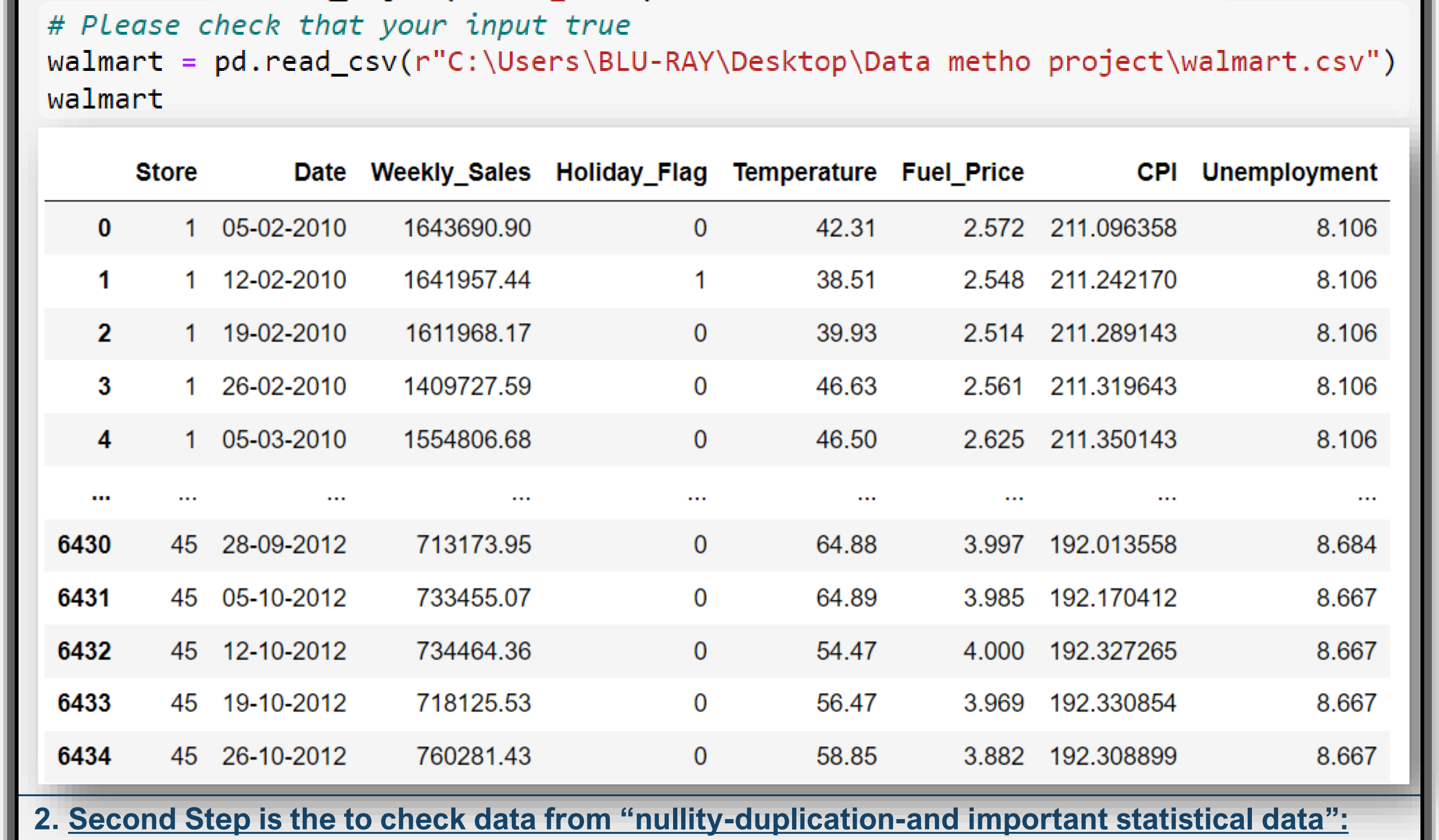
a) Which store has maximum sales?
b) Which store has maximum standard deviation i.e., the sales vary a lot.
c) Some holidays have a negative impact on sales. Find out holidays
that have higher sales than the mean sales in the non-holiday season for all stores together.
d) Provide a monthly and semester view of sales in units and give insights.
e) Plot the relations between weekly sales vs. other numeric features and give insights.

Explaining Work of the project:

1. Importing Data and the important modules used in the project:



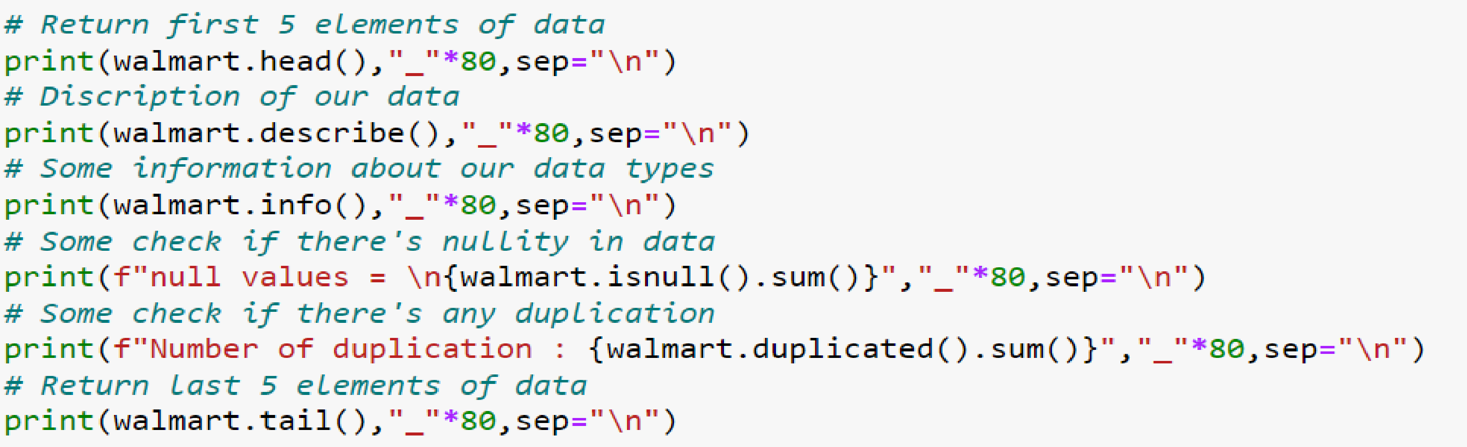
```
#First step let's put our needed modules in this project
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
import warnings as wn
wn.filterwarnings("ignore")
sns.set_style("darkgrid")
```



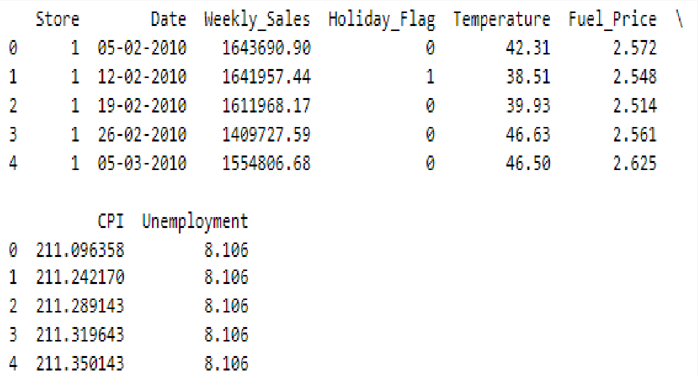
```
# Please check that your input true
walmart = pd.read_csv(r"C:\Users\BLU-RAY\Desktop\Data metho project\walmart.csv")
walmart
```

	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
...	...	...	...	...	...	...	...	...
6430	45	28-09-2012	713173.95	0	64.88	3.997	192.013558	8.684
6431	45	05-10-2012	733455.07	0	64.89	3.985	192.170412	8.667
6432	45	12-10-2012	734464.36	0	54.47	4.000	192.327265	8.667
6433	45	19-10-2012	718125.53	0	56.47	3.969	192.330854	8.667
6434	45	26-10-2012	760281.43	0	58.85	3.882	192.308899	8.667

2. Second Step is the to check data from “nullity-duplication-and important statistical data”:

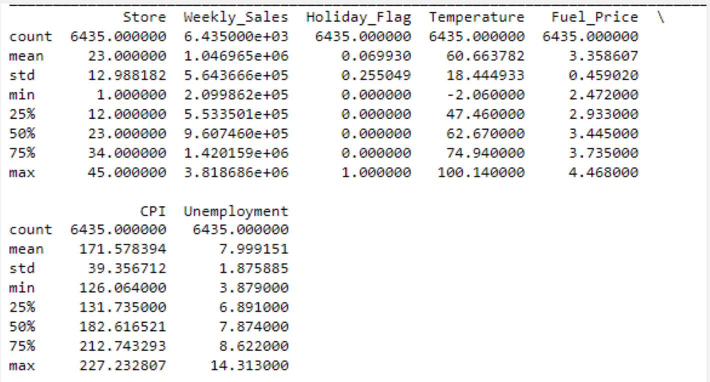


```
# Return first 5 elements of data
print(walmart.head(),"_"*80,sep="\n")
# Discription of our data
print(walmart.describe(),"_"*80,sep="\n")
# Some information about our data types
print(walmart.info(),"_"*80,sep="\n")
# Some check if there's nullity in data
print(f"null values = \n{walmart.isnull().sum()}","_"*80,sep="\n")
# Some check if there's any duplication
print(f"Number of duplication : {walmart.duplicated().sum()}","_"*80,sep="\n")
# Return last 5 elements of data
print(walmart.tail(),"_"*80,sep="\n")
```



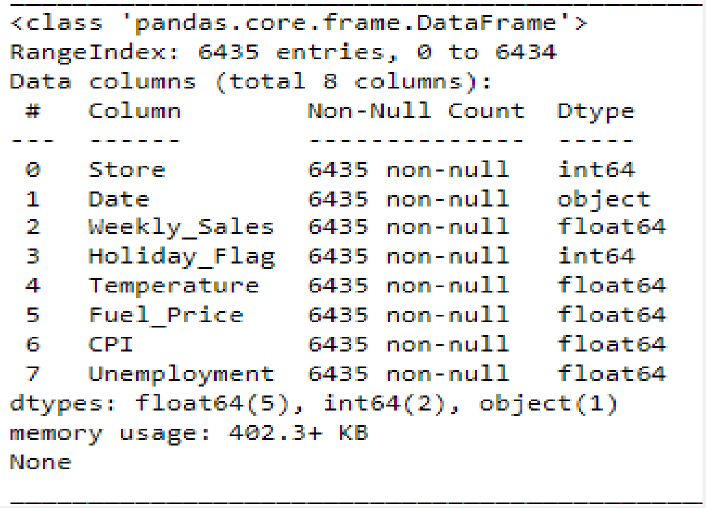
```
   Store        Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
0      1  05-02-2010    1643690.90             0        42.31       2.572
1      1  12-02-2010    1641957.44             1        38.51       2.548
2      1  19-02-2010    1611968.17             0        39.93       2.514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

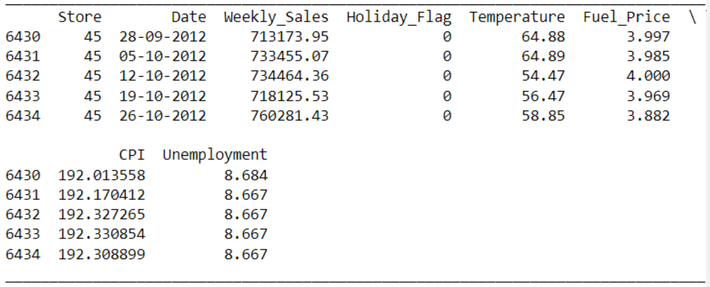


```
             Store  Weekly_Sales  Holiday_Flag  Temperature   Fuel_Price  \
count  6435.000000  6.435000e+03   6435.000000  6435.000000  6435.000000
mean     23.000000  1.046965e+06      0.069930    60.663782     3.358607
std      12.988182  5.643666e+05      0.255049    18.444933     0.459020
min       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
```

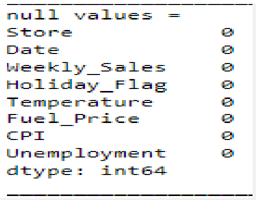


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6435 entries, 0 to 6434
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Store         6435 non-null   int64
 1   Date          6435 non-null   object
 2   Weekly_Sales  6435 non-null   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
None
```

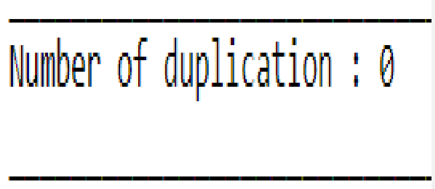


```
      Store        Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
6430     45  28-09-2012     713173.95             0        64.88       3.997
6431     45  05-10-2012     733455.07             0        64.89       3.985
6432     45  12-10-2012     734464.36             0        54.47       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

             CPI  Unemployment
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
```



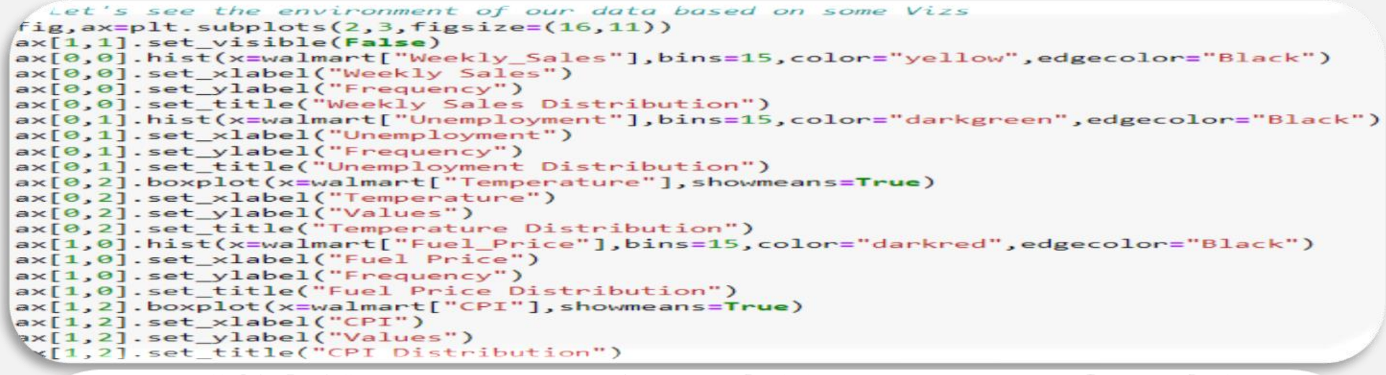
```
null values =
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
dtype: int64
```



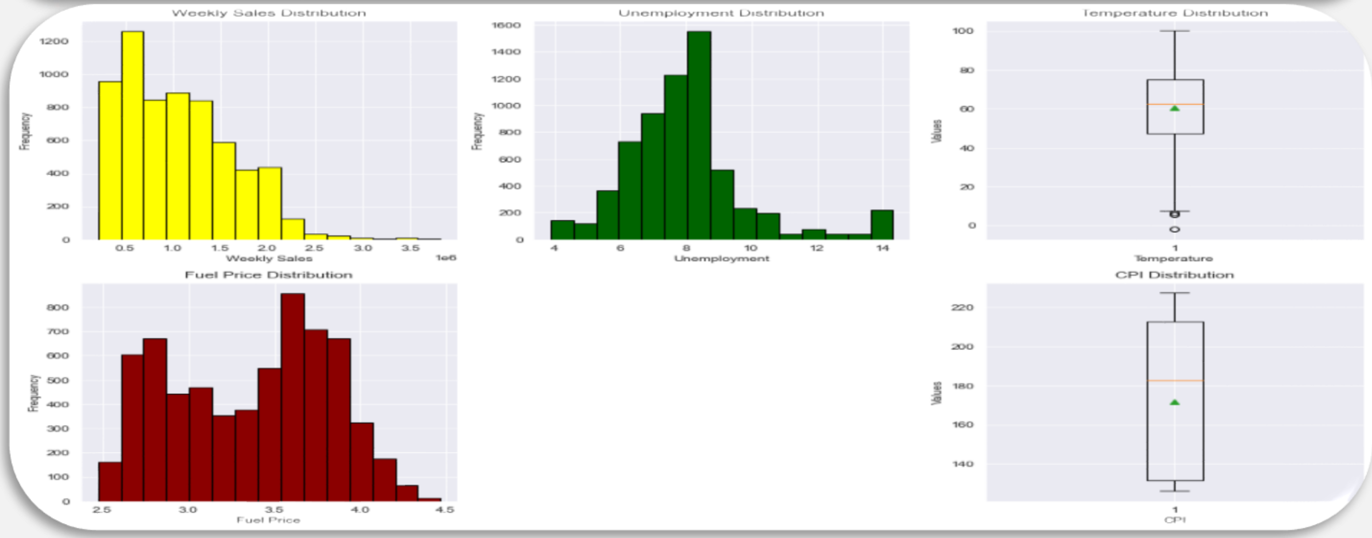
```
Number of duplication : 0
```

☑ Cleaned Data ☑

3. Third step visualize quantitative variables distributions:



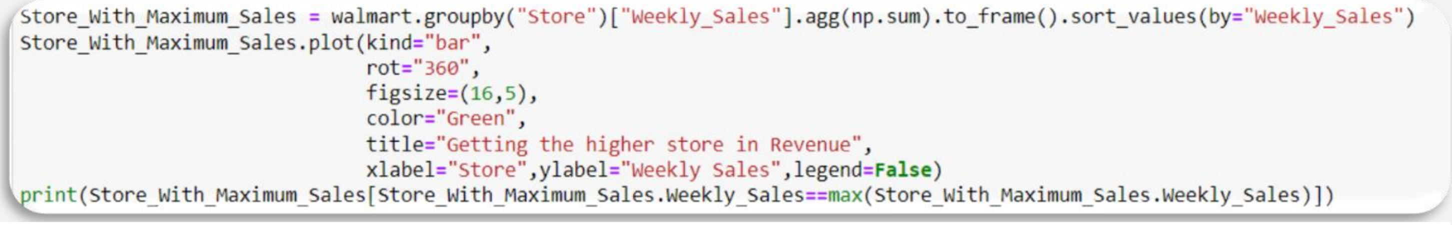
```
Let's see the environment of our data based on some Vizs
fig,ax=plt.subplots(2,3,figsize=(16,11))
ax[1,1].set_visible(False)
ax[0,0].hist(x=walmart["Weekly_Sales"],bins=15,color="yellow",edgecolor="Black")
ax[0,0].set_xlabel("Weekly Sales")
ax[0,0].set_ylabel("Frequency")
ax[0,0].set_title("Weekly Sales Distribution")
ax[0,1].hist(x=walmart["Unemployment"],bins=15,color="darkgreen",edgecolor="Black")
ax[0,1].set_xlabel("Unemployment")
ax[0,1].set_ylabel("Frequency")
ax[0,1].set_title("Unemployment Distribution")
ax[0,2].boxplot(x=walmart["Temperature"],showmeans=True)
ax[0,2].set_xlabel("Temperature")
ax[0,2].set_ylabel("Values")
ax[0,2].set_title("Temperature Distribution")
ax[1,0].hist(x=walmart["Fuel_Price"],bins=15,color="darkred",edgecolor="Black")
ax[1,0].set_xlabel("Fuel Price")
ax[1,0].set_ylabel("Frequency")
ax[1,0].set_title("Fuel Price Distribution")
ax[1,2].boxplot(x=walmart["CPI"],showmeans=True)
ax[1,2].set_xlabel("CPI")
ax[1,2].set_ylabel("Values")
ax[1,2].set_title("CPI Distribution")
```



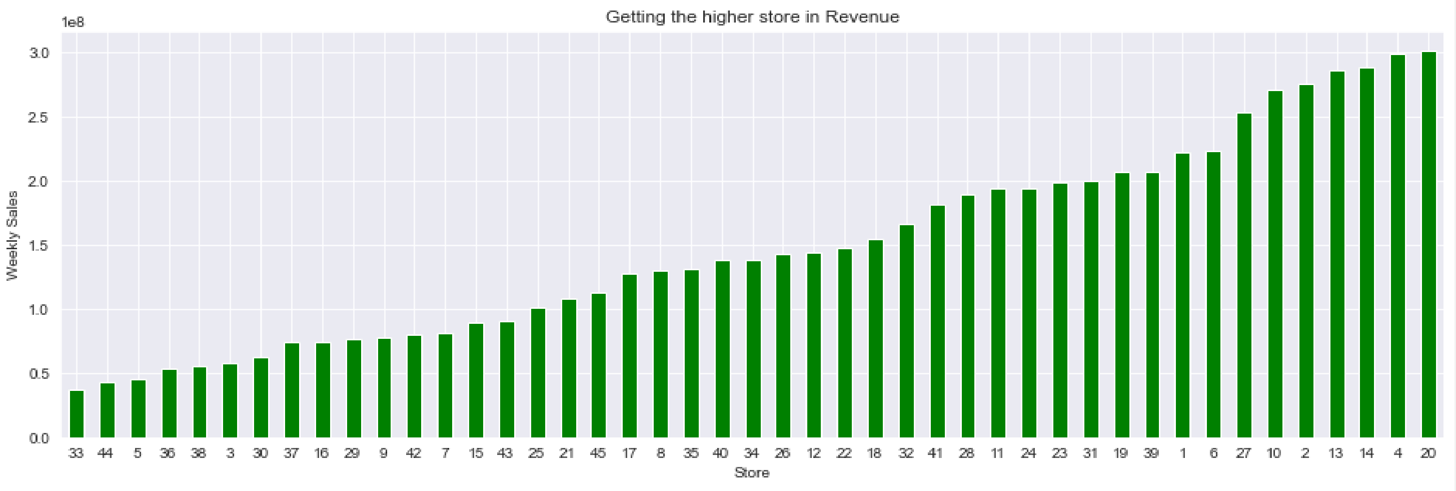
• Getting Insights based on answering questions •

a) Which store has maximum sales?

to perform this topic, we need to make pivoting between stores and sum of weekly sales to know who has the most sales between all weeks



```
Store_With_Maximum_Sales = walmart.groupby("Store")["Weekly_Sales"].agg(np.sum).to_frame().sort_values(by="Weekly_Sales")
Store_With_Maximum_Sales.plot(kind="bar",
                              rot="360",
                              figsize=(16,5),
                              color="Green",
                              title="Getting the higher store in Revenue",
                              xlabel="Store",ylabel="Weekly Sales",legend=False)
print(Store_With_Maximum_Sales[Store_With_Maximum_Sales.Weekly_Sales==max(Store_With_Maximum_Sales.Weekly_Sales)])
```

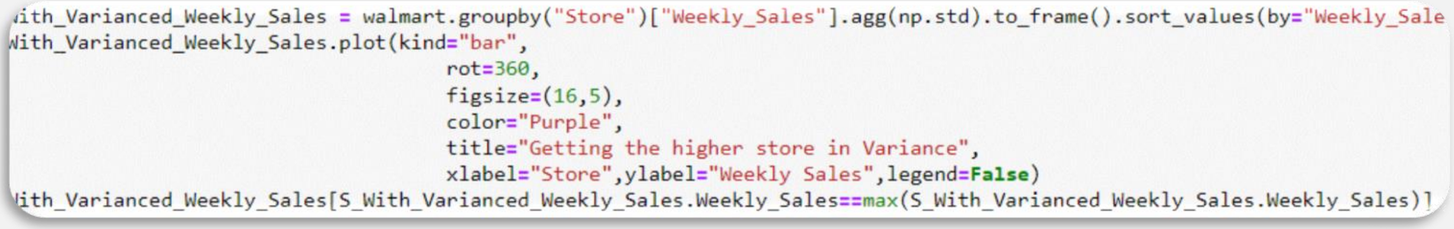


Based on help of Visualization to make sure that result is true

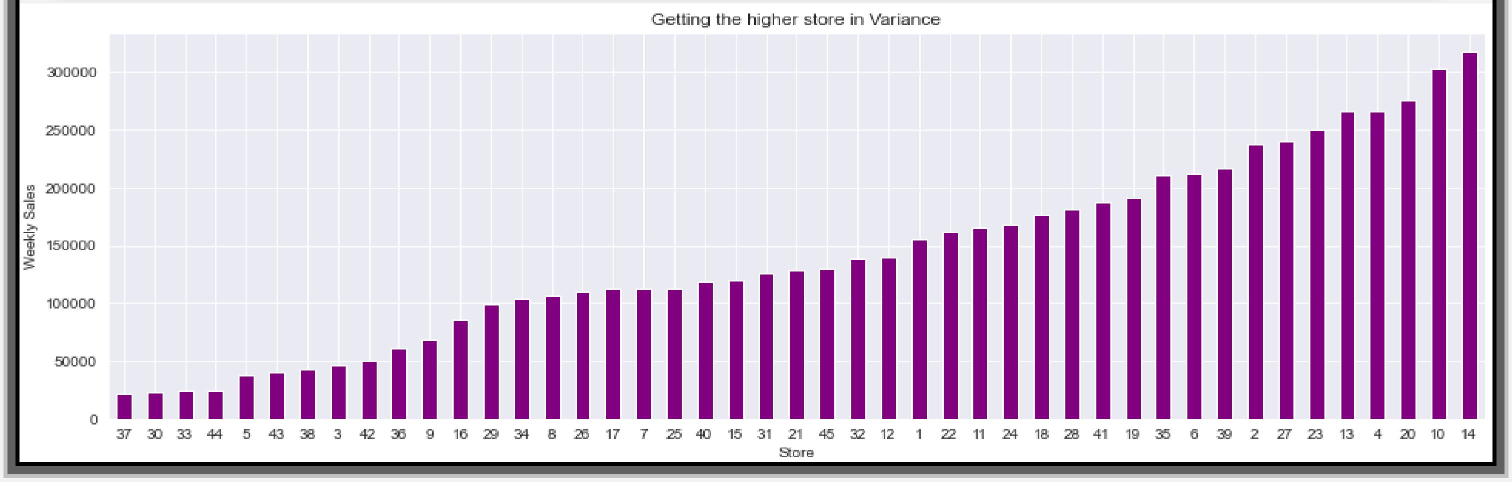
### Then the highest in Weekly sales was Store number "20" with weekly sales = 3.013978e+08 ###

b) Which store has maximum standard deviation i.e., the sales vary a lot:

To perform this topic we need to pivot between Store and the variance in sales



```
ith_Varianced_Weekly_Sales = walmart.groupby("Store")["Weekly_Sales"].agg(np.std).to_frame().sort_values(by="Weekly_Sale
With_Varianced_Weekly_Sales.plot(kind="bar",
                                  rot=360,
                                  figsize=(16,5),
                                  color="Purple",
                                  title="Getting the higher store in Variance",
                                  xlabel="Store",ylabel="Weekly Sales",legend=False)
ith_Varianced_Weekly_Sales[S_With_Varianced_Weekly_Sales.Weekly_Sales==max(S_With_Varianced_Weekly_Sales.Weekly_Sales)]
```



Sure as same as store with highly revenue style, You have noticed the store with Highest Variance Based on the Bar photo

### Then the store who has a great vary in the sales is store Number “14” with SD = 317569.949476 ###

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

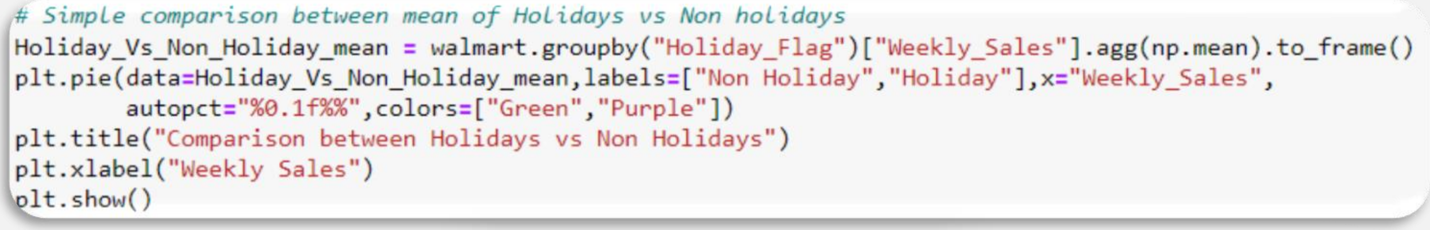
I can consider this question as the weirdest one in our data based on Holidays and the questions itself

So I solved this question in 2 ways to at least get the important insight from my point of view

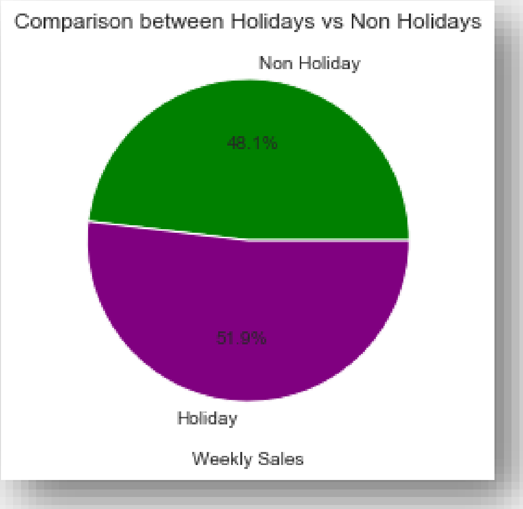
1. To perform that one then we need for Holiday flag, as 0 tends to non-holiday and 1 tends to holiday , and will make comparison between it and the weekly sales, Then get the stores with Negative Impact with Holidays

2. Get Real Date from Kaggle and make Compare by equaling it with the same date in data and extract mean of each one then gains the insight easily

Before Starting about this topic
Here is a little simple comparison between
Holidays vs non-Holidays

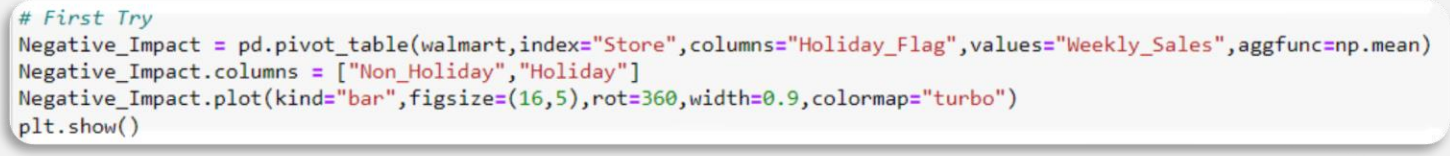


```
# Simple comparison between mean of Holidays vs Non holidays
Holiday_Vs_Non_Holiday_mean = walmart.groupby("Holiday_Flag")["Weekly_Sales"].agg(np.mean).to_frame()
plt.pie(data=Holiday_Vs_Non_Holiday_mean,labels=["Non Holiday","Holiday"],x="Weekly_Sales",
        autopct="%0.1f%%",colors=["Green","Purple"])
plt.title("Comparison between Holidays vs Non Holidays")
plt.xlabel("Weekly Sales")
plt.show()
```

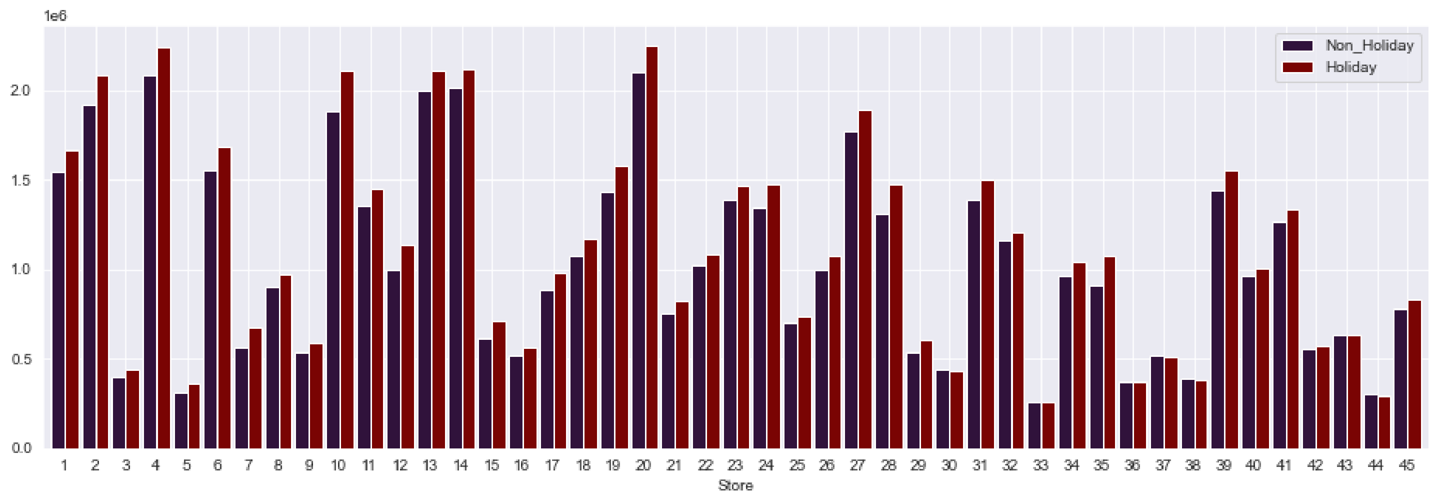


So Based in this comparison will see that the difference between revenue in Holidays and non-holidays is about 3.8%!

Performing first Try in Holidays Question:

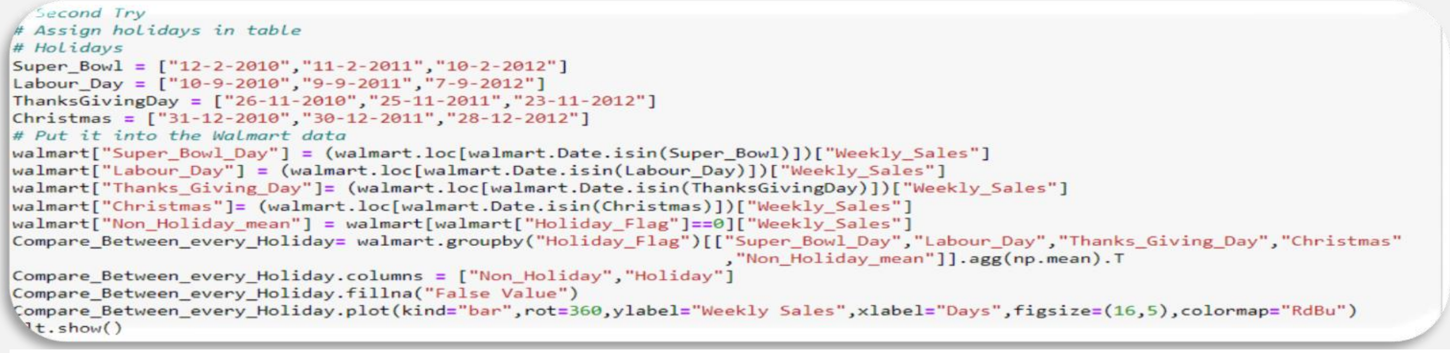


```
# First Try
Negative_Impact = pd.pivot_table(walmart,index="Store",columns="Holiday_Flag",values="Weekly_Sales",aggfunc=np.mean)
Negative_Impact.columns = ["Non_Holiday","Holiday"]
Negative_Impact.plot(kind="bar",figsize=(16,5),rot=360,width=0.9,colormap="turbo")
plt.show()
```

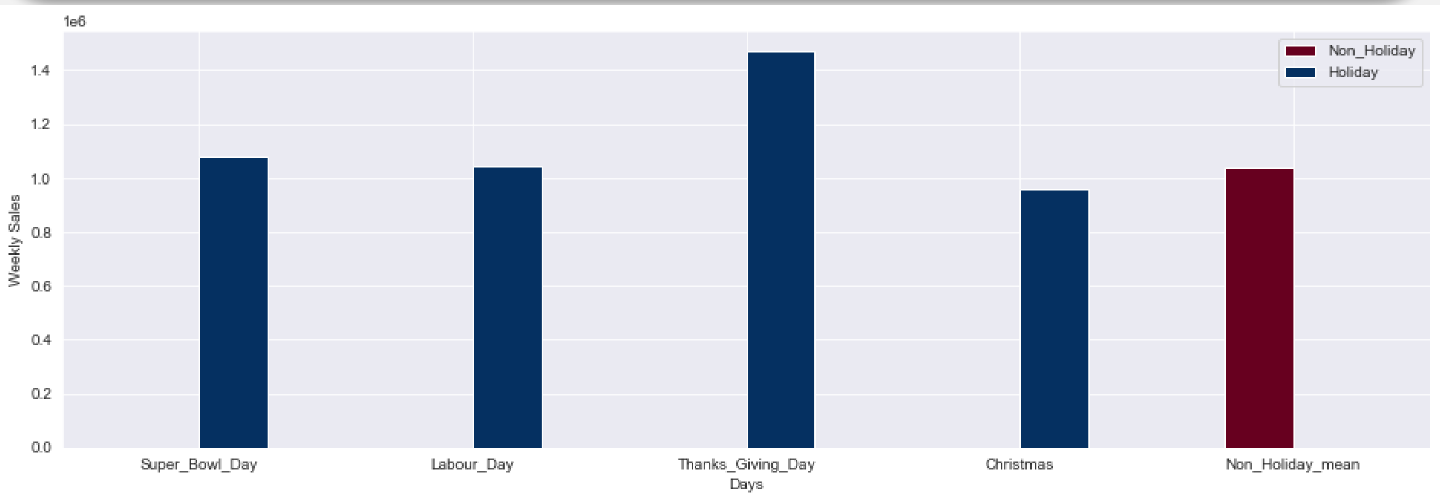


Simple view of this Viz has resulted a great insight, that he stores "30,36,37,38 and 44" Have a badly impact in revenue

Performing Second Try in Holidays Question:



```
Second Try
# Assign holidays in table
# Holidays
Super_Bowl = ["12-2-2010","11-2-2011","10-2-2012"]
Labour_Day = ["10-9-2010","9-9-2011","7-9-2012"]
ThanksGivingDay = ["26-11-2010","25-11-2011","23-11-2012"]
Christmas = ["31-12-2010","30-12-2011","28-12-2012"]
# Put it into the Walmart data
walmart["Super_Bowl_Day"] = (walmart.loc[walmart.Date.isin(Super_Bowl)])["Weekly_Sales"]
walmart["Labour_Day"] = (walmart.loc[walmart.Date.isin(Labour_Day)])["Weekly_Sales"]
walmart["Thanks_Giving_Day"]= (walmart.loc[walmart.Date.isin(ThanksGivingDay)])["Weekly_Sales"]
walmart["Christmas"]= (walmart.loc[walmart.Date.isin(Christmas)])["Weekly_Sales"]
walmart["Non_Holiday_mean"] = walmart[walmart["Holiday_Flag"]==0]["Weekly_Sales"]
Compare_Between_every_Holiday= walmart.groupby("Holiday_Flag")[["Super_Bowl_Day","Labour_Day","Thanks_Giving_Day","Christmas"
                                                                  ,"Non_Holiday_mean"]].agg(np.mean).T
Compare_Between_every_Holiday.columns = ["Non_Holiday","Holiday"]
Compare_Between_every_Holiday.fillna("False Value")
ompare_Between_every_Holiday.plot(kind="bar",rot=360,ylabel="Weekly Sales",xlabel="Days",figsize=(16,5),colormap="RdBu")
lt.show()
```



Then Notice again that the "Thanksgiving day" Was the best day in revenue, then "Super Bowl" then "Labour", and Christmas in last rank