11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [44]:

*# Import necessary libraries* **import pandas as pd**

**import seaborn as sns**

**import numpy as np**

**import matplotlib.pyplot as plt from matplotlib import** dates **from datetime import** datetime

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 1/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [45]:

*# Load dataset*

data = pd.read\_csv('Walmart\_Store\_sales.csv')

data

Out[45]:

**Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price CPI Unemploym** 05-

**0** 1 **1** 1 **2** 1 **3** 1 **4** 1

02-

2010

12-

02-

2010

19-

02-

2010

26-

02-

2010

05-

03-

2010

1643690.90 0 42.31 2.572 211.096358 8. 1641957.44 1 38.51 2.548 211.242170 8. 1611968.17 0 39.93 2.514 211.289143 8. 1409727.59 0 46.63 2.561 211.319643 8. 1554806.68 0 46.50 2.625 211.350143 8.

**...** ... ... ... ... ... ... ... 28-

**6430** 45 **6431** 45 **6432** 45 **6433** 45 **6434** 45

09-

2012

05-

10-

2012

12-

10-

2012

19-

10-

2012

26-

10-

2012

713173.95 0 64.88 3.997 192.013558 8. 733455.07 0 64.89 3.985 192.170412 8. 734464.36 0 54.47 4.000 192.327265 8. 718125.53 0 56.47 3.969 192.330854 8. 760281.43 0 58.85 3.882 192.308899 8.

6435 rows × 8 columns**Data Preparation**

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 2/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [46]:

*# Convert date to datetime format and show dataset information* data['Date'] = pd.to\_datetime(data['Date'])

data.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 datetime64[ns] 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: datetime64[ns](1), float64(5), int64(2) memory usage: 402.3 KB

In [47]:

*# checking for missing values*

data.isnull().sum()

Out[47]:

Store 0

Date 0

Weekly\_Sales 0

Holiday\_Flag 0

Temperature 0

Fuel\_Price 0

CPI 0

Unemployment 0

dtype: int64

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 3/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [48]:

*# Splitting Date and create new columns (Day, Month, and Year)* data["Day"]= pd.DatetimeIndex(data['Date']).day

data['Month'] = pd.DatetimeIndex(data['Date']).month

data['Year'] = pd.DatetimeIndex(data['Date']).year

data

Out[48]:

**Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price CPI Unemploym 0** 12010-

05-02 1643690.90 0 42.31 2.572 211.096358 8 **1** 12010-

12-02 1641957.44 1 38.51 2.548 211.242170 8 **2** 12010-

02-19 1611968.17 0 39.93 2.514 211.289143 8 **3** 12010-

02-26 1409727.59 0 46.63 2.561 211.319643 8 **4** 12010-

05-03 1554806.68 0 46.50 2.625 211.350143 8 **...** ... ... ... ... ... ... ... **6430** 45 2012-

09-28 713173.95 0 64.88 3.997 192.013558 8 **6431** 45 2012-

05-10 733455.07 0 64.89 3.985 192.170412 8 **6432** 45 2012-

12-10 734464.36 0 54.47 4.000 192.327265 8 **6433** 45 2012-

10-19 718125.53 0 56.47 3.969 192.330854 8 **6434** 45 2012-

10-26 760281.43 0 58.85 3.882 192.308899 8 6435 rows × 11 columns

**Q1: Which store has minimum and maximum sales?**

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 4/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [49]:

plt.figure(figsize=(15,7))

*# Sum Weekly\_Sales for each store, then sortded by total sales*

total\_sales\_for\_each\_store = data.groupby('Store')['Weekly\_Sales'].sum().sort\_values() total\_sales\_for\_each\_store\_array = np.array(total\_sales\_for\_each\_store) *# convert to array*

*# Assigning a specific color for the stores have the lowest and highest sales* clrs = ['lightsteelblue' **if** ((x < max(total\_sales\_for\_each\_store\_array)) **and** (x > min(tota l\_sales\_for\_each\_store\_array))) **else** 'midnightblue' **for** x **in** total\_sales\_for\_each\_store\_ar ray]

ax = total\_sales\_for\_each\_store.plot(kind='bar',color=clrs);

*# store have minimum sales*

p = ax.patches[0]

print(type(p.get\_height()))

ax.annotate("The store has minimum sales is 33 with **{0:.2f}** $".format((p.get\_height())), x y=(p.get\_x(), p.get\_height()), xycoords='data',

xytext=(0.17, 0.32), textcoords='axes fraction',

arrowprops=dict(arrowstyle="->", connectionstyle="arc3"), horizontalalignment='center', verticalalignment='center')

*# store have maximum sales*

p = ax.patches[44]

ax.annotate("The store has maximum sales is 20 with **{0:.2f}** $".format((p.get\_height())), x y=(p.get\_x(), p.get\_height()), xycoords='data',

xytext=(0.82, 0.98), textcoords='axes fraction',

arrowprops=dict(arrowstyle="->", connectionstyle="arc3"), horizontalalignment='center', verticalalignment='center')

*# plot properties*

plt.xticks(rotation=0)

plt.ticklabel\_format(useOffset=**False**, style='plain', axis='y')

plt.title('Total sales for each store')

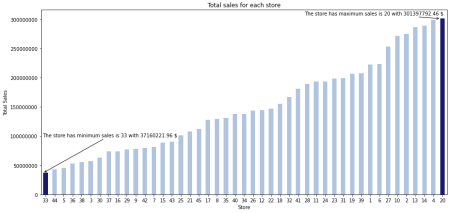
plt.xlabel('Store')

plt.ylabel('Total Sales');

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 5/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

<class 'numpy.float64'>



**Q2: Which store has maximum standard deviation i.e., the sales vary a lot. Also, find out the coefficient of mean to standard deviation?**

In [50]:

*# Which store has maximum standard deviation*

data\_std = pd.DataFrame(data.groupby('Store')['Weekly\_Sales'].std().sort\_values(ascending= **False**))

print("The store has maximum standard deviation is "+str(data\_std.head(1).index[0])+" with **{0:.0f}** $".format(data\_std.head(1).Weekly\_Sales[data\_std.head(1).index[0]]))

The store has maximum standard deviation is 14 with 317570 $

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 6/29

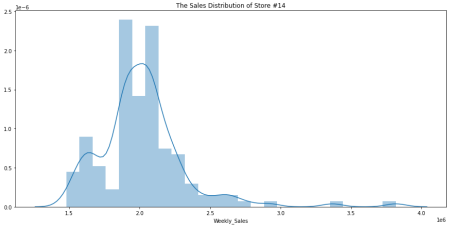
11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [51]:

*# Distribution of store has maximum standard deviation*

plt.figure(figsize=(15,7))

sns.distplot(data[data['Store'] == data\_std.head(1).index[0]]['Weekly\_Sales']) plt.title('The Sales Distribution of Store #'+ str(data\_std.head(1).index[0]));

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 7/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [52]:

*# Coefficient of mean to standard deviation*

coef\_mean\_std = pd.DataFrame(data.groupby('Store')['Weekly\_Sales'].std() / data.groupby('S tore')['Weekly\_Sales'].mean())

coef\_mean\_std = coef\_mean\_std.rename(columns={'Weekly\_Sales':'Coefficient of mean to stand ard deviation'})

coef\_mean\_std

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 8/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

Out[52]:

**Coefficient of mean to standard deviation Store**

**1** 0.100292 **2** 0.123424 **3** 0.115021 **4** 0.127083 **5** 0.118668 **6** 0.135823 **7** 0.197305 **8** 0.116953 **9** 0.126895

**10** 0.159133 **11** 0.122262 **12** 0.137925 **13** 0.132514 **14** 0.157137 **15** 0.193384 **16** 0.165181 **17** 0.125521 **18** 0.162845 **19** 0.132680 **20** 0.130903 **21** 0.170292 **22** 0.156783 **23** 0.179721 **24** 0.123637 **25** 0.159860 **26** 0.110111 **27** 0.135155 **28** 0.137330 **29** 0.183742 **30** 0.052008 **31** 0.090161 **32** 0.118310 **33** 0.092868

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 9/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

**Coefficient of mean to standard deviation**

**Store**

**34** 0.108225

**35** 0.229681

**36** 0.162579

**37** 0.042084

**38** 0.110875

**39** 0.149908

**40** 0.123430

**41** 0.148177

**42** 0.090335

**43** 0.064104

**44** 0.081793

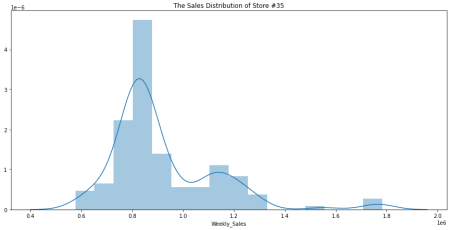
**45** 0.165613

In [53]:

*# Distribution of store has maximum coefficient of mean to standard deviation* coef\_mean\_std\_max = coef\_mean\_std.sort\_values(by='Coefficient of mean to standard deviatio n')

plt.figure(figsize=(15,7))

sns.distplot(data[data['Store'] == coef\_mean\_std\_max.tail(1).index[0]]['Weekly\_Sales']) plt.title('The Sales Distribution of Store #'+str(coef\_mean\_std\_max.tail(1).index[0]));

**Q3: Which store/s has good quarterly growth rate in Q3’2012**

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 10/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

plt.figure(figsize=(15,7))

**Sales for third quarterly in 2012**

Q3 = data[(data['Date'] > '2012-07-01') & (data['Date'] < '2012-09-30')].groupby('Store')['Weekly\_Sales'].sum() **Sales for second quarterly in 2012**

Q2 = data[(data['Date'] > '2012-04-01') & (data['Date'] < '2012-06-30')].groupby('Store')['Weekly\_Sales'].sum()

**Plotting the difference between sales for second and third quarterly**

Q2.plot(ax=Q3.plot('bar',legend=True),kind='bar',color='r',alpha=0.2,legend=True); plt.legend(["Q3' 2012", "Q2' 2012"]);

In [54]:

*# store/s has good quarterly growth rate in Q3’2012 - .sort\_values(by='Weekly\_Sales')* print('Store have good quarterly growth rate in Q3’2012 is Store '+str(Q3.idxmax())+' With '+str(Q3.max())+' $')

Store have good quarterly growth rate in Q3’2012 is Store 4 With 25652119.35 $

**Q4: 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**

**Holiday Events:**

Super Bowl: 12-Feb-10, 11-Feb-11, 10-Feb-12, 8-Feb-13

Labour Day: 10-Sep-10, 9-Sep-11, 7-Sep-12, 6-Sep-13

Thanksgiving: 26-Nov-10, 25-Nov-11, 23-Nov-12, 29-Nov-13

Christmas: 31-Dec-10, 30-Dec-11, 28-Dec-12, 27-Dec-13

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 11/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [55]:

**def** plot\_line(df,holiday\_dates,holiday\_label):

fig, ax = plt.subplots(figsize = (15,5))

ax.plot(df['Date'],df['Weekly\_Sales'],label=holiday\_label)

**for** day **in** holiday\_dates:

day = datetime.strptime(day, '**%d**-%m-%Y')

plt.axvline(x=day, linestyle='--', c='r')

plt.title(holiday\_label)

x\_dates = df['Date'].dt.strftime('%Y-%m-**%d**').sort\_values().unique() xfmt = dates.DateFormatter('**%d**-%m-%y')

ax.xaxis.set\_major\_formatter(xfmt)

ax.xaxis.set\_major\_locator(dates.DayLocator(1))

plt.gcf().autofmt\_xdate(rotation=90)

plt.show()

total\_sales = data.groupby('Date')['Weekly\_Sales'].sum().reset\_index() 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']

plot\_line(total\_sales,Super\_Bowl,'Super Bowl')

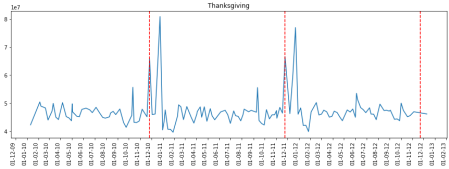
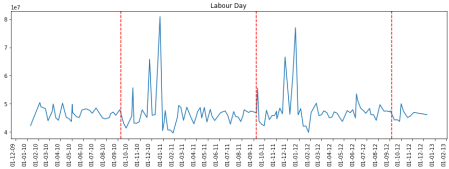
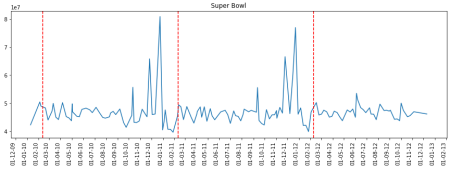
plot\_line(total\_sales,Labour\_Day,'Labour Day')

plot\_line(total\_sales,Thanksgiving,'Thanksgiving')

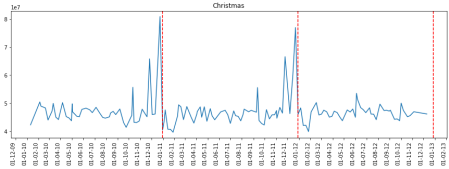
plot\_line(total\_sales,Christmas,'Christmas')

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 12/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data



https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 13/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data 

**The sales increased during thanksgiving. And the sales decreased during christmas.**

In [56]:

data.loc[data.Date.isin(Super\_Bowl)]

Out[56]:

**Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price CPI Unemploym 1** 12010-

12-02 1641957.44 1 38.51 2.548 211.242170 8 **53** 12011-

11-02 1649614.93 1 36.39 3.022 212.936705 7 **105** 12012-

10-02 1802477.43 1 48.02 3.409 220.265178 7 **144** 22010-

12-02 2137809.50 1 38.49 2.548 210.897994 8 **196** 22011-

11-02 2168041.61 1 33.19 3.022 212.592862 8 **...** ... ... ... ... ... ... ... **6202** 44 2011-

11-02 307486.73 1 30.83 3.034 127.859129 7 **6254** 44 2012-

10-02 325377.97 1 33.73 3.116 130.384903 5 **6293** 45 2010-

12-02 656988.64 1 27.73 2.773 181.982317 8 **6345** 45 2011-

11-02 766456.00 1 30.30 3.239 183.701613 8 **6397** 45 2012-

10-02 803657.12 1 37.00 3.640 189.707605 8 135 rows × 11 columns

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 14/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [57]:

*# Yearly Sales in holidays*

Super\_Bowl\_df = pd.DataFrame(data.loc[data.Date.isin(Super\_Bowl)].groupby('Year')['Weekly\_ Sales'].sum())

Thanksgiving\_df = pd.DataFrame(data.loc[data.Date.isin(Thanksgiving)].groupby('Year')['Wee kly\_Sales'].sum())

Labour\_Day\_df = pd.DataFrame(data.loc[data.Date.isin(Labour\_Day)].groupby('Year')['Weekly\_ Sales'].sum())

Christmas\_df = pd.DataFrame(data.loc[data.Date.isin(Christmas)].groupby('Year')['Weekly\_Sa les'].sum())

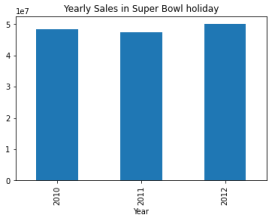
Super\_Bowl\_df.plot(kind='bar',legend=**False**,title='Yearly Sales in Super Bowl holiday') Thanksgiving\_df.plot(kind='bar',legend=**False**,title='Yearly Sales in Thanksgiving holiday') Labour\_Day\_df.plot(kind='bar',legend=**False**,title='Yearly Sales in Labour\_Day holiday') Christmas\_df.plot(kind='bar',legend=**False**,title='Yearly Sales in Christmas holiday')

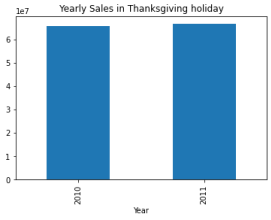
https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 15/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

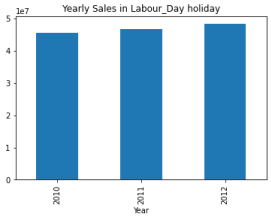
Out[57]:

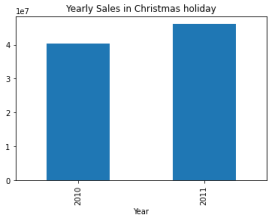
<AxesSubplot:title={'center':'Yearly Sales in Christmas holiday'}, xlabel='Ye ar'>





https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 16/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data 



**Q5: Provide a monthly and semester view of sales in units and give insights**

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 17/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [58]:

*# Monthly view of sales for each years*

plt.scatter(data[data.Year==2010]["Month"],data[data.Year==2010]["Weekly\_Sales"]) plt.xlabel("months")

plt.ylabel("Weekly Sales")

plt.title("Monthly view of sales in 2010")

plt.show()

plt.scatter(data[data.Year==2011]["Month"],data[data.Year==2011]["Weekly\_Sales"]) plt.xlabel("months")

plt.ylabel("Weekly Sales")

plt.title("Monthly view of sales in 2011")

plt.show()

plt.scatter(data[data.Year==2012]["Month"],data[data.Year==2012]["Weekly\_Sales"]) plt.xlabel("months")

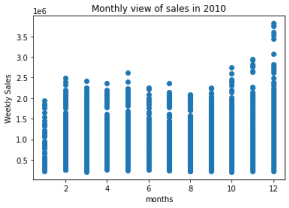
plt.ylabel("Weekly Sales")

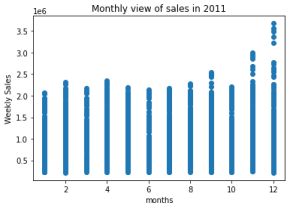
plt.title("Monthly view of sales in 2012")

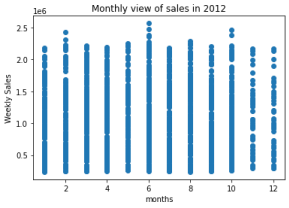
plt.show()

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 18/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data







https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 19/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [59]:

*# Monthly view of sales for all years*

plt.figure(figsize=(10,6))

plt.bar(data["Month"],data["Weekly\_Sales"])

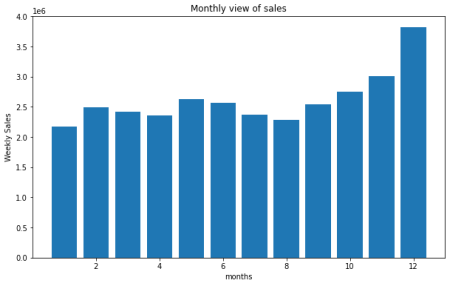
plt.xlabel("months")

plt.ylabel("Weekly Sales")

plt.title("Monthly view of sales")

Out[59]:

Text(0.5, 1.0, 'Monthly view of sales')

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 20/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [60]:

*# Yearly view of sales*

plt.figure(figsize=(10,6))

data.groupby("Year")[["Weekly\_Sales"]].sum().plot(kind='bar',legend=**False**) plt.xlabel("years")

plt.ylabel("Weekly Sales")

plt.title("Yearly view of sales");

<Figure size 720x432 with 0 Axes>



**Build prediction models to forecast demand (Modeling)**

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 21/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [61]:

*# find outliers*

fig, axs = plt.subplots(4,figsize=(6,18))

X = data[['Temperature','Fuel\_Price','CPI','Unemployment']] **for** i,column **in** enumerate(X):

sns.boxplot(data[column], ax=axs[i])

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 22/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 23/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [62]:

*# drop the outliers*

data\_new = data[(data['Unemployment']<10) & (data['Unemployment']>4.5) & (data['Temperatur e']>10)]

data\_new

Out[62]:

**Store Date Weekly\_Sales Holiday\_Flag Temperature Fuel\_Price CPI Unemploym 0** 12010-

05-02 1643690.90 0 42.31 2.572 211.096358 8

**1** 12010-

12-02 1641957.44 1 38.51 2.548 211.242170 8

**2** 12010-

02-19 1611968.17 0 39.93 2.514 211.289143 8

**3** 12010-

02-26 1409727.59 0 46.63 2.561 211.319643 8

**4** 12010-

05-03 1554806.68 0 46.50 2.625 211.350143 8

**...** ... ... ... ... ... ... ...

**6430** 45 2012-

09-28 713173.95 0 64.88 3.997 192.013558 8

**6431** 45 2012-

05-10 733455.07 0 64.89 3.985 192.170412 8

**6432** 45 2012-

12-10 734464.36 0 54.47 4.000 192.327265 8

**6433** 45 2012-

10-19 718125.53 0 56.47 3.969 192.330854 8

**6434** 45 2012-

10-26 760281.43 0 58.85 3.882 192.308899 8

5658 rows × 11 columns

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 24/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [63]:

*# check outliers*

fig, axs = plt.subplots(4,figsize=(6,18))

X = data\_new[['Temperature','Fuel\_Price','CPI','Unemployment']] **for** i,column **in** enumerate(X):

sns.boxplot(data\_new[column], ax=axs[i])

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 25/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 26/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

**Build Model**

In [64]:

*# Import sklearn*

**from sklearn.ensemble import** RandomForestRegressor

**from sklearn.model\_selection import** train\_test\_split

**from sklearn import** metrics

**from sklearn.linear\_model import** LinearRegression

In [65]:

*# Select features and target*

X = data\_new[['Store','Fuel\_Price','CPI','Unemployment','Day','Month','Year']] y = data\_new['Weekly\_Sales']

*# Split data to train and test (0.80:0.20)*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,test\_size=0.2)

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 27/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [66]:

*# Linear Regression model*

print('Linear Regression:')

print()

reg = LinearRegression()

reg.fit(X\_train, y\_train)

y\_pred = reg.predict(X\_test)

print('Accuracy:',reg.score(X\_train, y\_train)\*100)

print('Mean Absolute Error:', metrics.mean\_absolute\_error(y\_test, y\_pred)) print('Mean Squared Error:', metrics.mean\_squared\_error(y\_test, y\_pred)) print('Root Mean Squared Error:', np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred)))

sns.scatterplot(y\_pred, y\_test);

Linear Regression:

Accuracy: 12.512726429399478

Mean Absolute Error: 445859.2353432851

Mean Squared Error: 279505769725.82086

Root Mean Squared Error: 528683.0522400173



# Random Forest Regressor print('Random Forest Regressor:') print() rfr = RandomForestRegressor(n\_estimators = 400,max\_depth=15,n\_jobs=5) rfr.fit(X\_train,y\_train) y\_pred=rfr.predict(X\_test) print('Accuracy:',rfr.score(X\_test, y\_test)\*100) print('Mean Absolute Error:', metrics.mean\_absolute\_error(y\_test, y\_pred)) print('Mean Squared Error:', metrics.mean\_squared\_error(y\_test, y\_pred)) print('Root Mean Squared Error:',

np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred))) sns.scatterplot(y\_pred, y\_test);

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 28/29

11/15/21, 8:27 PM Retail Analysis with Walmart Data

In [ ]: In [ ]: In [ ]:

https://bosch.lms.simplilearn.com/courses/2772/Data-Science-with-Python/practice-labs 29/29