Code

#import the libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from matplotlib import style

%matplotlib inline

#import the dataset

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

# print the first few rows

raw\_data.head()

#print the columns

raw\_data.columns

#check for missing values

raw\_data.isnull().sum()

#group the entire dataset

sales\_group = raw\_data.groupby(by=['Store']).sum()

sales\_group

group\_total\_store = raw\_data[['Store','Weekly\_Sales']]

group\_total\_store

#group sales by store

group\_by\_store = group\_total\_store.groupby(by=['Store']).sum()

group\_by\_store = group\_by\_store.sort\_values(by=['Weekly\_Sales'],ascending=False)

group\_by\_store

#group by store and then find the standard deviation of each store

df\_std = pd.DataFrame(raw\_data.groupby('Store').agg({'Weekly\_Sales':['std','mean']}))

df\_std

df\_std['Weekly\_Sales','std'].max()

df\_std.loc[df\_std[('Weekly\_Sales','std')]==df\_std[('Weekly\_Sales','std')].max()]

#converte the Date column to datetime

raw\_data.Date = pd.to\_datetime(raw\_data.Date)

#raw\_data.Date

#slide the dataset to get the Q2 saes

q2 = raw\_data[(raw\_data['Date']>='2012-04-01') & (raw\_data['Date']<='2012-06-30')]

#slide the dataset to get the Q3 sales

q3 = raw\_data[(raw\_data['Date']>='2012-07-01') & (raw\_data['Date']<='2012-09-30')]

q3

#finding the sum weekly sales of each store in Q2

q2\_sales= pd.DataFrame(q2.groupby('Store')['Weekly\_Sales'].sum())

q2\_sales.reset\_index(inplace=True)

q2\_sales.rename(columns={'Weekly\_Sales': 'Q2\_Weekly\_Sales'},inplace=True)

q2\_sales.head()

#finding the sum weekly sales of each store in Q3

q3\_sales = pd.DataFrame(q3.groupby(by=['Store'])['Weekly\_Sales'].sum())

q3\_sales.reset\_index(inplace=True)

q3\_sales.rename(columns={'Weekly\_Sales': 'Q3\_Weekly\_Sales'},inplace=True)

q3\_sales.head()

#mergeing Q2 and Q3 data on Store as a common column

q3\_Growth= q2\_sales.merge(q3\_sales,how='inner',on='Store')

q3\_Growth.head()##### Calculating Growth rate of each Store. Formula (q3-q2)/q2\*100

q3\_Growth['Growth\_Rate'] = (q3\_Growth['Q3\_Weekly\_Sales'] - q3\_Growth['Q2\_Weekly\_Sales'])/ q3\_Growth['Q2\_Weekly\_Sales']

q3\_Growth['Groth\_Rate']= round(q3\_Growth['Growth\_Rate'],2)

q3\_Growth.sort\_values('Growth\_Rate',ascending=False).head()

#split the dataset into holidays and non-holidays dataset

#holidays dattaset

holidays\_df = raw\_data[raw\_data['Holiday\_Flag']==1]

holidays\_df

#non hilidays dataset

non\_holidays\_df= raw\_data[raw\_data['Holiday\_Flag']==0]

non\_holidays\_df

#find the mean sales in non-holiday season for all stores together

non\_holidays\_sales\_mean = non\_holidays\_df['Weekly\_Sales'].mean()

print(f'The non holidays sales mean is:{non\_holidays\_sales\_mean:.02f}')

#holidays which have higher sales than the mean sales in non-holiday season

holidays\_high\_sales = holidays\_df[holidays\_df['Weekly\_Sales'] > non\_holidays\_sales\_mean]

holidays\_high\_sales

#Monthly sales

monthly = raw\_data.groupby(pd.Grouper(key='Date', freq='1M')).sum()# groupby each 1 month

monthly=monthly.reset\_index()

fig, ax = plt.subplots(figsize=(10,8))

X = monthly['Date']

Y = monthly['Weekly\_Sales']

plt.plot(X,Y)

plt.title('Month Wise Sales')

plt.xlabel('Monthly')

plt.ylabel('Weekly\_Sales')

#Sales by semester

semester = raw\_data.groupby(pd.Grouper(key='Date', freq='6M')).sum()

semester = semester.reset\_index()

semester

fig, ax = plt.subplots(figsize=(10,8))

X = semester['Date']

Y = semester['Weekly\_Sales']

plt.plot(X,Y)

plt.title('Semester Wise Sales')

plt.xlabel('Semester')

plt.ylabel('Weekly\_Sales')

#encoding the column data

from sklearn.preprocessing import LabelEncoder

LabEncoder = LabelEncoder()

#get the store 1 data

store\_1 = raw\_data[raw\_data['Store']==1]

store\_1

#encode the date column

store\_1['New\_date'] = LabEncoder.fit\_transform(store\_1['Date'])

store\_1

#drop columns

store\_1.drop(['Store','Date','Holiday\_Flag'],axis=1 , inplace = True)

store\_1.head()

#get Correlation matrix of Variables.

corrmatrix = store\_1.corr()

print("Correlation matrix is : ")

print(corrmatrix)

#let's drop Fuel\_Price and CPI

df\_model = store\_1[['Weekly\_Sales','Temperature','Unemployment','New\_date']]

df\_model.shape

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

features = df\_model.drop('Weekly\_Sales',axis=1)

label = df\_model['Weekly\_Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features, label, test\_size=0.2, random\_state = 43)

from sklearn.linear\_model import LinearRegression

line\_model = LinearRegression()

line\_model.fit(X\_train,y\_train)

##Prediction

predict\_val = line\_model.predict(X\_test)

predict\_val = predict\_val.round(2)

predict\_val

df\_act\_pred = pd.DataFrame({"Actual Sales" : y\_test, "Predicted Sales": predict\_val})

df\_act\_pred.head()