```
import pandas as pd
from datetime import datetime
import numpy as np
```

```
dataset="/content/superstore_final_dataset (1).csv"
```

```
df=pd.read_csv(dataset,encoding="latin1")
```

```
df
```

	Row_ID	Order_ID	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment	Country	City	State	Postal_Cod
0	1	CA-2017-152156	8/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420.
1	2	CA-2017-152156	8/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420.
2	3	CA-2017-138688	12/6/2017	16/06/2017	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036.
3	4	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O Donnel	Consumer	United States	Fort Lauderdale	Florida	33311.
4	5	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O Donnel	Consumer	United States	Fort Lauderdale	Florida	33311.
...	...	...	...	...	...	...	...	...	...	...	...	..
9795	9796	CA-2017-125920	21/05/2017	28/05/2017	Standard Class	SH-19975	Sally Hughsby	Corporate	United States	Chicago	Illinois	60610.
9796	9797	CA-2016-128608	12/1/2016	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	Ohio	43615.
9797	9798	CA-2016-128608	12/1/2016	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	Ohio	43615.
9798	9799	CA-2016-128608	12/1/2016	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	Ohio	43615.
9799	9800	CA-2016-128608	12/1/2016	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	Ohio	43615.

9800 rows × 18 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape
```

```
(9800, 18)
```

```
df['Order_Date']=pd.to_datetime(df['Order_Date'],errors='coerce')
df['Order_Date']=df['Order_Date'].fillna(method='ffill')
```

```
<ipython-input-10-d0f10da09b12>:2: FutureWarning: Series.fillna with 'method' is deprecated and will raise in a future version. Use obj.
  df['Order_Date']=df['Order_Date'].fillna(method='ffill')
```

```
df['year']=df['Order_Date'].dt.year
df['month']=df['Order_Date'].dt.month
df['day']=df['Order_Date'].dt.day
```

```
df
```

	Row_ID	Order_ID	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment	Country	City	...	Postal_Code	Re
0	1	CA-2017-152156	2017-08-11	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420.0	S
1	2	CA-2017-152156	2017-08-11	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420.0	S
2	3	CA-2017-138688	2017-12-06	16/06/2017	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036.0	\
3	4	US-2016-108966	2016-11-10	18/10/2016	Standard Class	SO-20335	Sean O Donnel	Consumer	United States	Fort Lauderdale	...	33311.0	S
4	5	US-2016-108966	2016-11-10	18/10/2016	Standard Class	SO-20335	Sean O Donnel	Consumer	United States	Fort Lauderdale	...	33311.0	S
...	...	...	...	...	...	...	...	...	...	...	...	...	
9795	9796	CA-2017-125920	2015-05-11	28/05/2017	Standard Class	SH-19975	Sally Hughsby	Corporate	United States	Chicago	...	60610.0	Ce
9796	9797	CA-2016-128608	2016-12-01	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	...	43615.0	
9797	9798	CA-2016-128608	2016-12-01	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	...	43615.0	
9798	9799	CA-2016-128608	2016-12-01	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	...	43615.0	
9799	9800	CA-2016-128608	2016-12-01	17/01/2016	Standard Class	CS-12490	Cindy Schnelling	Corporate	United States	Toledo	...	43615.0	

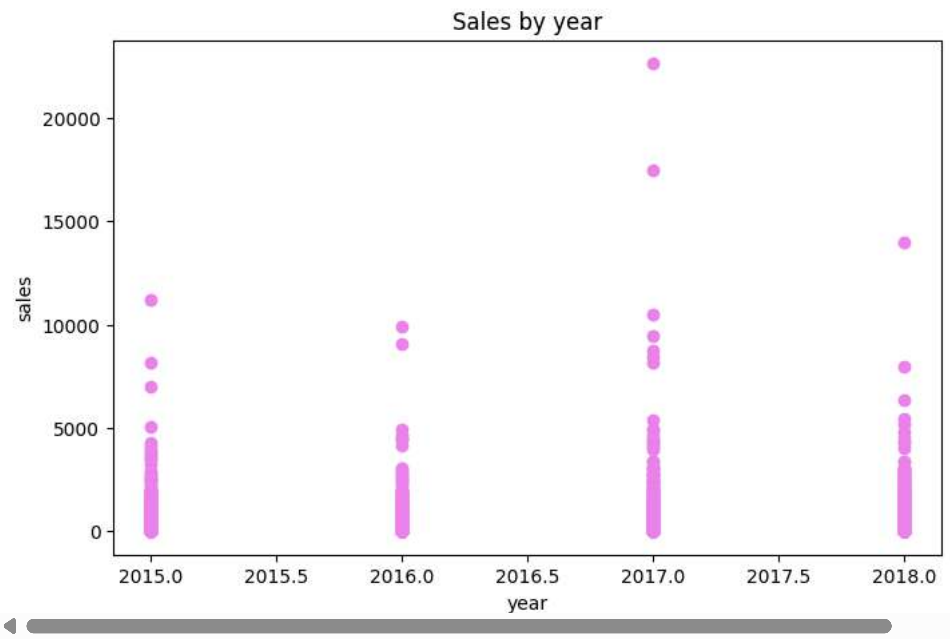
9800 rows × 21 columns

```
df.isnull().sum()
```

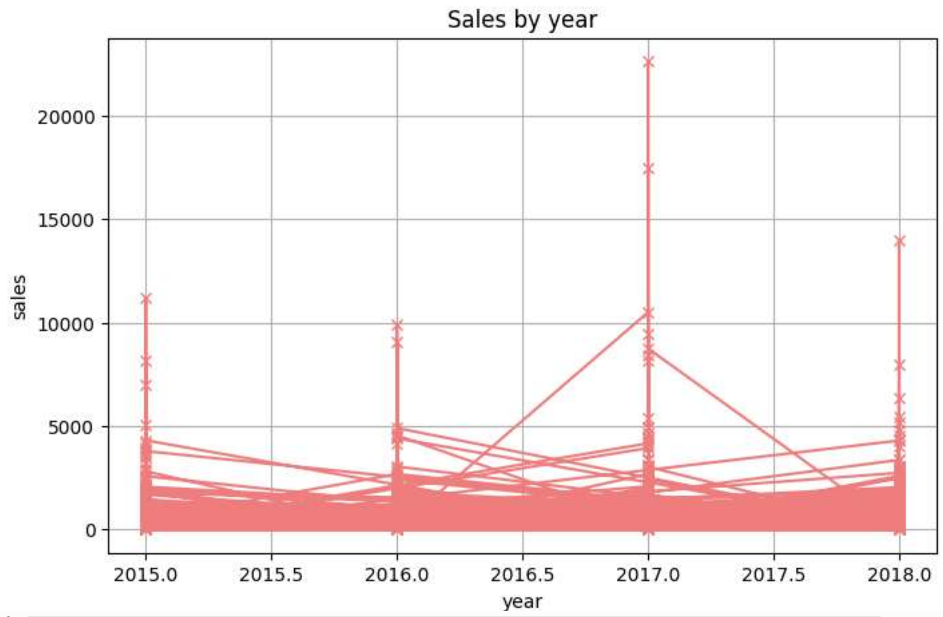
	0
Row_ID	0
Order_ID	0
Order_Date	0
Ship_Date	0
Ship_Mode	0
Customer_ID	0
Customer_Name	0
Segment	0
Country	0
City	0
State	0
Postal_Code	11
Region	0
Product_ID	0
Category	0
Sub_Category	0
Product_Name	0
Sales	0
year	0
month	0
day	0

```
dtype: int64
```

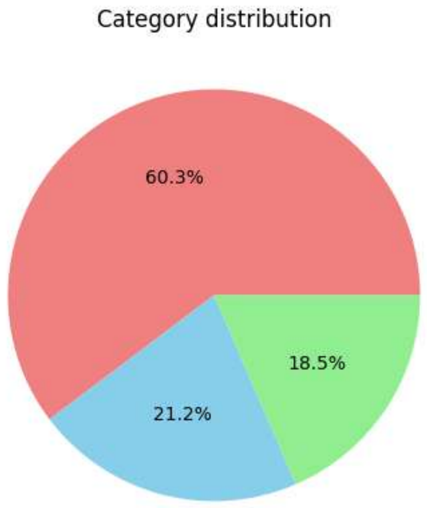
```
import matplotlib.pyplot as plt


plt.figure(figsize=(8,5))
plt.scatter(df['year'],df['Sales'],color='violet')
plt.xlabel("year")
plt.ylabel("sales")
plt.title("Sales by year")
plt.show()
```



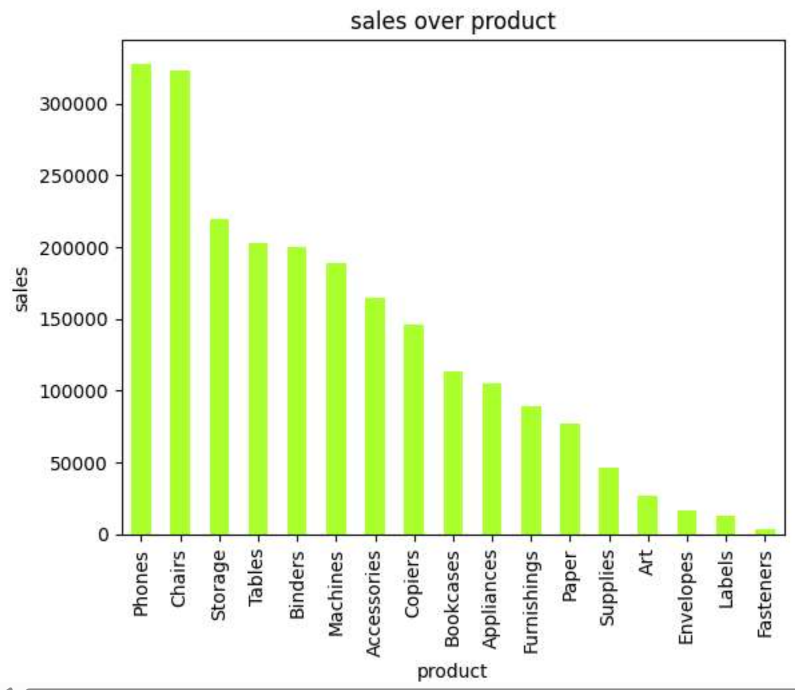
```
plt.figure(figsize=(8,5))
plt.plot(df['year'],df['Sales'],color='lightcoral',marker='x')
plt.xlabel("year")
plt.ylabel("sales")
plt.title("Sales by year")
plt.grid(True)
plt.show()
```



```
plt.figure(figsize=(8,5))
category=df["Category"].value_counts()
plt.pie(category,autopct='%1.1f%%',colors=['lightcoral','skyblue','lightgreen'])
plt.title("Category distribution")
plt.show()
```



```
products=df.groupby('Sub_Category')['Sales'].sum().sort_values(ascending=False)
products.plot(kind="bar",color='greenyellow')
plt.xlabel("product")
plt.ylabel("sales")
plt.title("sales over product")
plt.show()
```



```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression


X = df[['year','month','day']]
Y = df['Sales']


X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size=0.2,random_state=42)
model = LinearRegression()
model.fit(X_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
from sklearn.metrics import mean_absolute_error,mean_squared_error

y_pred = model.predict(X_test)

MAE = mean_absolute_error(y_test,y_pred)
MSE = mean_squared_error(y_test,y_pred)
RMSE = np.sqrt(MSE)

print(f'Mean Absolute Error : {MAE}')
print(f'Mean squared Error : {MSE}')
print(f'Root Mean squared Error : {RMSE}')
```

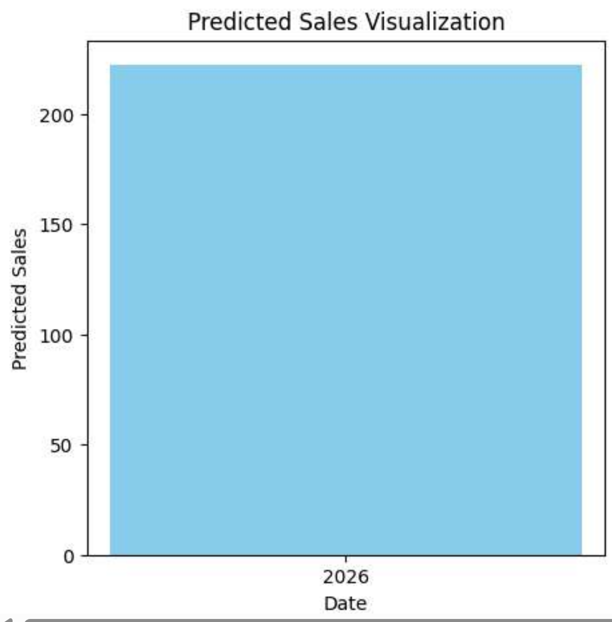
```
Mean Absolute Error : 303.36236440467957
Mean squared Error : 670696.4220762155
Root Mean squared Error : 818.9605741891459
```

```
print("choose prediction 1-yearly, 2-monthly, 3-daily ")
value=int(input("choose (1/2/3) :"))
dates= []
predictions= []
if value==1:
  years=int(input("enter your year for prediction(YYYY):"))
  newvalue=np.array([[years,1,1]])
  predictedsales=model.predict(newvalue)
  print(f"The Predicted Sales for {years} is : {predictedsales[0]}")
  dates.append(str(years))
  predictions.append(predictedsales[0])
elif value==2:
  years=int(input("enter your year for prediction(YYYY):"))
  months=int(input("enter your month for prection(1-12):"))
  newvalue=np.array([[years,months,1]])
```

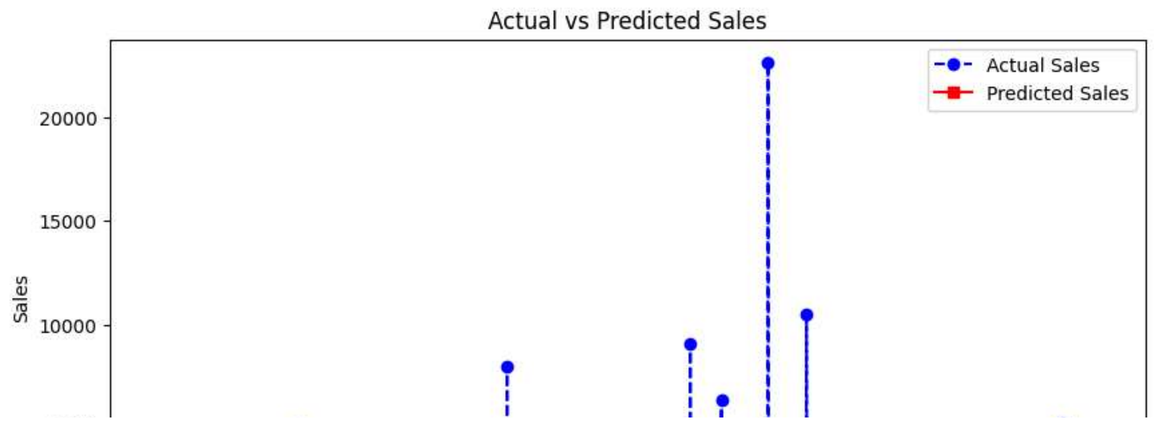
```
    predictedsales=model.predict(newvalue)
    print(f"The Predicted Sales for {years}-{months} is : {predictedsales[0]}")
    dates.append(str(years)+str(months))
    predictions.append(predictedsales[0])
  elif value==3:
    years=int(input("enter your year for prediction(YYYY):"))
    months=int(input("enter your month for prection(1-12):"))
    days=int(input("enter your days for prediction(1-31):"))
    newvalue=np.array([[years,months,days]])
    predictedsales=model.predict(newvalue)
    print(f"The Predicted Sales for {years}-{months}-{days} is : {predictedsales[0]}")
    dates.append(str(years)+str(months)+str(days))
    predictions.append(predictedsales[0])
  else:
    print("please enter a valid choice")
```

```
choose prediction 1-yearly, 2-monthly, 3-daily
choose (1/2/3) :1
enter your year for prediction(YYYY):2026
The Predicted Sales for 2026 is : 222.17017648867113
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but LinearRe
  warnings.warn(
```

```
plt.figure(figsize=(5, 5))
plt.bar(dates, predictions, color='skyblue')
plt.xlabel("Date")
plt.ylabel("Predicted Sales")
plt.title("Predicted Sales Visualization")
plt.show()
```



```
y_pred = model.predict(X_test)
plt.figure(figsize=(10, 5))
plt.plot(y_test.values, label="Actual Sales", marker='o', linestyle='dashed', color='blue')
plt.plot(y_pred, label="Predicted Sales", marker='s', linestyle='solid', color='red')
plt.xlabel("Data")
plt.ylabel("Sales")
plt.title("Actual vs Predicted Sales")
plt.legend()
plt.show()
```


Actual vs Predicted Sales
Actual Sales
Predicted Sales
Sales
20000
15000
10000