Sales_Analytics using Python

1.cumulative sales

```
import pandas as pd
data=pd.read_csv("sales_data.csv")
df=pd.DataFrame(data)
df['CumSales'] = df['Sales_Amount'].cumsum()
print(df["CumSales"].head())
```

```
0     5053.97
1     9437.99
2    14069.22
3    16237.16
4    19987.36
Name: CumSales, dtype: float64
```

2. Percentage of Total

```
print(df["Sales_Amount"].dtype)
df['Sales_Amount'] = df['Sales_Amount'].fillna(0)
df['SalesPct']=df['Sales_Amount']/df['Sales_Amount'].sum()*100
df['SalesPct']=df['SalesPct'].round(2)
print(df[['Sales_Amount','SalesPct']].head())
```

```
float64
   Sales_Amount  SalesPct
0       5053.97      0.10
1       4384.02      0.09
2       4631.23      0.09
3       2167.94      0.04
4       3750.20      0.07
```

3. Quartile Segmentation

```
import pandas as pd
df['SalesGroup']=pd.qcut(df['Sales_Amount'],q=4,labels=['low','mediam','high','top'])
print(df['SalesGroup'].head(5))
```

```
0      high
1    mediam
2    mediam
3       low
4    mediam
Name: SalesGroup, dtype: category
Categories (4, object): ['low' < 'mediam' < 'high' < 'top']
```

4. Running Average (Rolling)

```
df['7D_Avg'] = df['Sales'].rolling(7).mean()
```

```
0            NaN
1            NaN
2            NaN
3            NaN
4            NaN
5            NaN
6    3480.974286
7    3858.824286
8    3835.877143
9    4351.355714
10   5258.571429
11   5036.934286
12   5443.598571
13   6037.495714
14   6196.728571
Name: Sales_Amount, dtype: float64
```

5. Customer Recency

```
print(df['Sale_Date'].dtype)
df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])
df['Recency']=(df['Sale_Date'].max() - df['Sale_Date']).dt.days
print(df[['Sale_Date','Recency']].head())
```

```
datetime64[ns]
   Sale_Date  Recency
0 2023-02-03      332
1 2023-04-21      255
2 2023-09-21      102
3 2023-08-24      130
4 2023-03-24      283
```

6. Cumulative % (Pareto 80/20)

```
df_sorted=df.sort_values('Sales_Amount',ascending=False)
df['CumPct']=df_sorted['Sales_Amount'].cumsum()/df['Sales_Amount'].sum() *100
print(df['CumPct'].head())
```

```
0    74.273497
1    80.460660
2    78.409732
3    95.406976
4    85.530313
Name: CumPct, dtype: float64
```

7. Sales Growth %

```
df['Growth'] = df['Sales_Amount'].pct_change() * 100
df['Growth'].head()
```

```
0          NaN
1   -13.255916
2     5.638889
3   -53.188678
4    72.984492
Name: Growth, dtype: float64
```

8. Profit Margin %

```
df['Actual_Amount']=df['Sales_Amount']*df['Discount']
df['Profit']=df['Sales_Amount']-df['Actual_Amount']
df['ProfitMargin'] = df['Profit'] / df['Sales_Amount'] * 100
df.sort_values('ProfitMargin',ascending=False).head(10)
df.head()
```

_Category	Unit_Cost	Unit_Price	Customer_Type	Discount	Payment_Method	Sales_Channel	Region_and_Sales_Rep	Month	Actual_Amount	Profit	ProfitMargin
Furniture	152.75	267.22	Returning	0.09	Cash	Online	North-Bob	2	454.8573	4599.1127	91.0
Furniture	3816.39	4209.44	Returning	0.11	Cash	Retail	West-Bob	4	482.2422	3901.7778	89.0
Food	261.56	371.40	Returning	0.20	Bank Transfer	Retail	South-David	9	926.2460	3704.9840	80.0
Clothing	4330.03	4467.75	New	0.02	Credit Card	Retail	South-Bob	8	43.3588	2124.5812	98.0
Electronics	637.37	692.71	New	0.08	Credit Card	Online	East-Charlie	3	300.0160	3450.1840	92.0

9. Correlation Check
df[['Discount','Profit','Sales']].corr()

	Discount	Sales_Amount	Profit
Discount	1.000000	0.023153	-0.15690
Sales_Amount	0.023153	1.000000	0.97841
Profit	-0.156900	0.978410	1.00000

10. Top N Products
top_products = df.groupby('Product')['Sales'].sum().nlargest(5)

```
Product_Category
Clothing     1313474.36
Furniture    1260517.69
Name: Sales_Amount, dtype: float64
```

11. Bottom N Products
low_products = df.groupby('Product')['Sales'].sum().nsmallest(5)

```
Product_Category
Food           1201773.54
Electronics    1243499.64
Name: Sales_Amount, dtype: float64
```

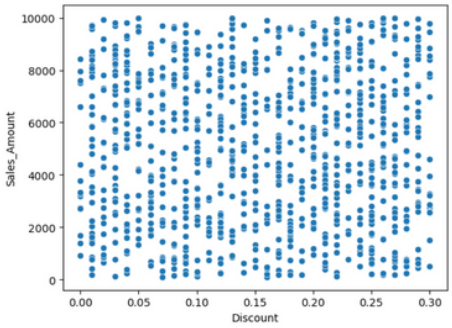
12. Monthly Sales Pivot
import datetime as dt
df['Sale_Date']=pd.to_datetime(df['Sale_Date'])
df['Month']=df['Sale_Date'].dt.month
df.pivot_table(values='Sales_Amount',index='Month',aggfunc='sum')

	Sales_Amount
Month	
1	495420.37
2	368919.36
3	402638.77
4	438992.61
5	389078.76
6	418458.34
7	374242.88
8	443171.28
9	367837.60
10	460378.78
11	467482.90
12	392643.58

13. Region-Category Mix
pd.crosstab(df['Region'], df['Category'], values=df['Sales'], aggfunc='sum', normalize='index')

Product_Category	Clothing	Electronics	Food	Furniture
Region				
East	0.283118	0.240596	0.258665	0.217620
North	0.272323	0.250192	0.189094	0.288391
South	0.233500	0.254420	0.260938	0.251143
West	0.254376	0.246088	0.255530	0.244006

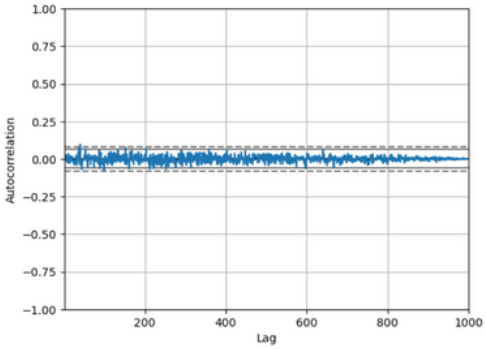
14. Discount Impact
import seaborn as sns
sns.scatterplot(x='Discount',y='Sales_Amount',data=df)



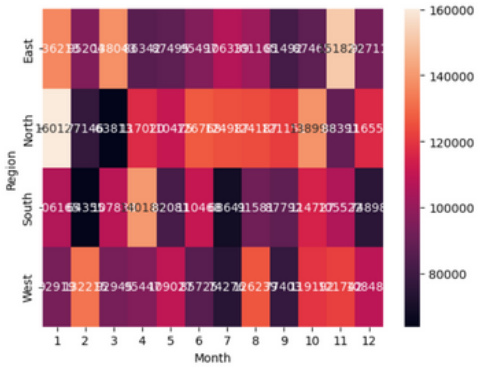
15. Product Value (PV)
df.groupby('Product_ID')['Sales_Amount'].sum().sort_values(ascending=False).nlargest(5)

```
Product_ID
1099    101773.87
1092     90615.62
1033     89130.41
1090     88043.25
1086     82269.71
Name: Sales_Amount, dtype: float64
```

16. Seasonality Detection
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(df['Sales_Amount'])



17. Heatmap of Sales by Region & Month
sales_matrix = df.pivot_table(values='Sales_Amount', index='Region', columns='Month', aggfunc='sum')
sns.heatmap(sales_matrix, annot=True, fmt='.0f')



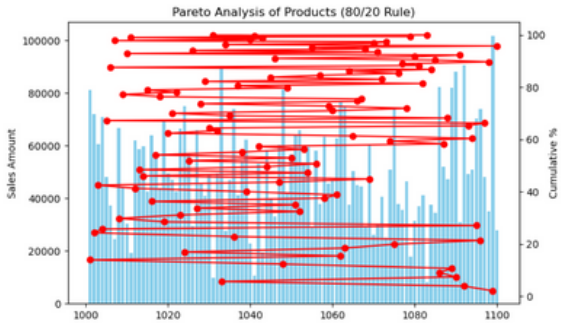
18. Plot Pareto chart

```
import matplotlib.pyplot as plt
df_sorted = df.groupby('Product_ID')['Sales_Amount'].sum().sort_values(ascending=False)
cum_pct = df_sorted.cumsum() / df_sorted.sum() * 100

fig, ax1 = plt.subplots(figsize=(8,5))

ax1.bar(df_sorted.index, df_sorted, color="skyblue")
ax2 = ax1.twinx()
ax2.plot(df_sorted.index, cum_pct, color="red", marker="o")

ax1.set_ylabel("Sales Amount")
ax2.set_ylabel("Cumulative %")
ax1.set_title("Pareto Analysis of Products (80/20 Rule)")
plt.xticks(rotation=90)
plt.show()
```



19.Monthly Sales

```
monthly = df.groupby(df['Sale_Date'].dt.to_period("M"))['Sales_Amount'].sum()

monthly.plot(kind="line", marker="o", figsize=(8,5), title="Monthly Sales Trend")
plt.ylabel("Sales Amount")
plt.show()
```

20.Recency Distribution (RFM Analysis Start)

```
import seaborn as sns

df['Sale_Date'] = pd.to_datetime(df['Sale_Date'])
df['Recency'] = (df['Sale_Date'].max() - df['Sale_Date']).dt.days

sns.histplot(df['Recency'], bins=20, kde=True)
plt.title("Distribution of Customer Recency")
plt.xlabel("Days Since Last Purchase")
plt.show()
```