```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
from google.colab import files
uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Superstore_USA.xlsx to Superstore_USA (1).xlsx
```

```
dataset=pd.read_excel("Superstore_USA.xlsx")
```

```
dataset.head(5)
```

	Row ID	Order Priority	Discount	Unit Price	Shipping Cost	Customer ID	Customer Name	Ship Mode	Customer Segment	Product Category	...	Region	State or Province	Cit
0	18606	Not Specified	0.01	2.88	0.50	2	Janice Fletcher	Regular Air	Corporate	Office Supplies	...	Central	Illinois	Addiso
1	20847	High	0.01	2.84	0.93	3	Bonnie Potter	Express Air	Corporate	Office Supplies	...	West	Washington	Anacorte
2	23086	Not Specified	0.03	6.68	6.15	3	Bonnie Potter	Express Air	Corporate	Office Supplies	...	West	Washington	Anacorte
3	23087	Not Specified	0.01	5.68	3.60	3	Bonnie Potter	Regular Air	Corporate	Office Supplies	...	West	Washington	Anacorte
4	23088	Not Specified	0.00	205.99	2.50	3	Bonnie Potter	Express Air	Corporate	Technology	...	West	Washington	Anacorte

5 rows × 24 columns

```
dataset.shape
```

```
(9426, 24)
```

```
dataset.isnull().sum()
```

	0
Row ID	0
Order Priority	0
Discount	0
Unit Price	0
Shipping Cost	0
Customer ID	0
Customer Name	0
Ship Mode	0
Customer Segment	0
Product Category	0
Product Sub-Category	0
Product Container	0
Product Name	0
Product Base Margin	72
Region	0
State or Province	0
City	0
Postal Code	0
Order Date	0
Ship Date	0
Profit	0
Quantity ordered new	0
Sales	0
Order ID	0

dtype: int64

```
dataset["Order Year"] = dataset["Order Date"].dt.year
```

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9426 entries, 0 to 9425
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Row ID                9426 non-null   int64
 1   Order Priority        9426 non-null   object
 2   Discount              9426 non-null   float64
 3   Unit Price            9426 non-null   float64
 4   Shipping Cost         9426 non-null   float64
 5   Customer ID           9426 non-null   int64
 6   Customer Name         9426 non-null   object
 7   Ship Mode             9426 non-null   object
 8   Customer Segment      9426 non-null   object
 9   Product Category      9426 non-null   object
 10  Product Sub-Category  9426 non-null   object
 11  Product Container     9426 non-null   object
 12  Product Name          9426 non-null   object
 13  Product Base Margin   9354 non-null   float64
 14  Region                9426 non-null   object
 15  State or Province     9426 non-null   object
 16  City                  9426 non-null   object
 17  Postal Code           9426 non-null   int64
 18  Order Date            9426 non-null   datetime64[ns]
 19  Ship Date             9426 non-null   datetime64[ns]
 20  Profit                9426 non-null   float64
 21  Quantity ordered new  9426 non-null   int64
 22  Sales                 9426 non-null   float64
 23  Order ID              9426 non-null   int64
 24  Order Year            9426 non-null   int32
```

```
dtypes: datetime64[ns](2), float64(6), int32(1), int64(5), object(11)
memory usage: 1.8+ MB
```

```
dataset['Product Base Margin'].fillna(dataset['Product Base Margin'].mean(), inplace=True)
```

```
/tmp/ipython-input-14-3445482844.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through cha
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are sett

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df


  dataset['Product Base Margin'].fillna(dataset['Product Base Margin'].mean(), inplace=True)
```

```
dataset['Order Priority'].value_counts()
```

```
                count
Order Priority
          High  1970
           Low  1926
 Not Specified  1881
        Medium  1844
      Critical  1804
      Critical     1

dtype: int64
```

Data Cleaning

```
dataset['Order Priority'].unique()
```

```
array(['Not Specified', 'High', 'Medium', 'Low', 'Critical', 'Critical '],
      dtype=object)
```

```
dataset['Order Priority'].replace("Critical","Critical ",inplace=True)
```

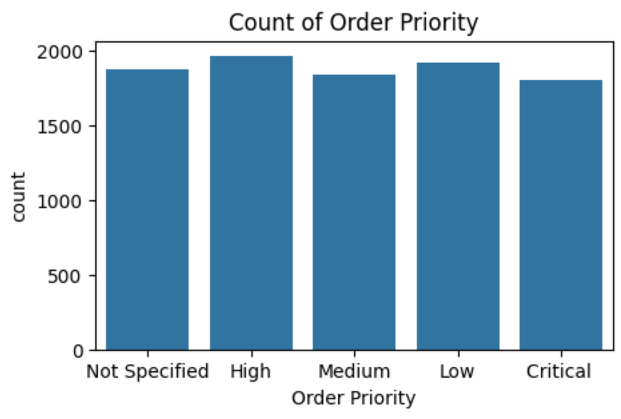
```
/tmp/ipython-input-17-1880193826.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through cha
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are sett

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df


  dataset['Order Priority'].replace("Critical","Critical ",inplace=True)
```

```
dataset['Order Priority'].value_counts()
```

```
                count
Order Priority
          High  1970
           Low  1926
 Not Specified  1881
        Medium  1844
      Critical  1805

dtype: int64
```

Order Priority

```
plt.figure(figsize=(5,3))
sns.countplot(x="Order Priority",data=dataset)
plt.title("Count of Order Priority")
plt.show()
#plt.savefig("Count of Order Priority.pdf")
```

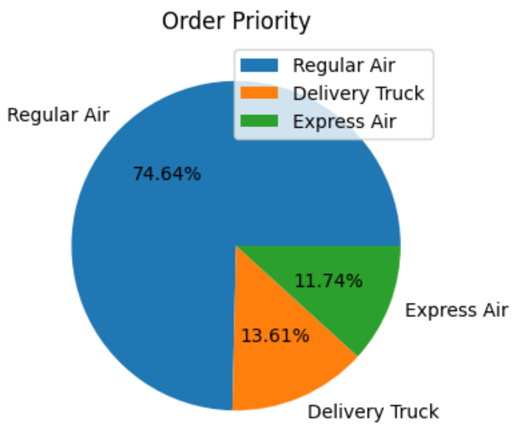


Shipping Mode

```
dataset['Ship Mode'].value_counts()
```

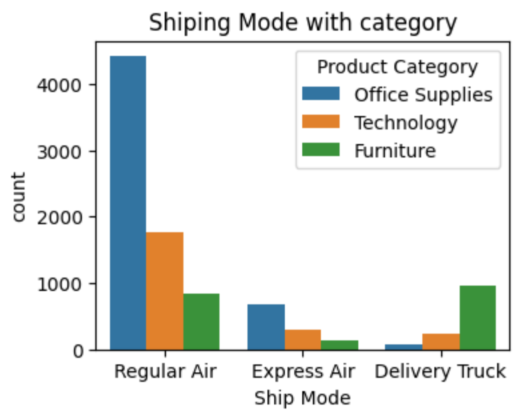
```
                 count
     Ship Mode

   Regular Air    7036
Delivery Truck    1283
   Express Air    1107

dtype: int64
```

```
=X = dataset['Ship Mode'].value_counts().index
Y = dataset['Ship Mode'].value_counts().values
plt.figure(figsize=(5,4))
plt.pie(Y,labels=X,autopct='%1.2f%%')
plt.title('Order Priority')
plt.legend()
plt.show()
```



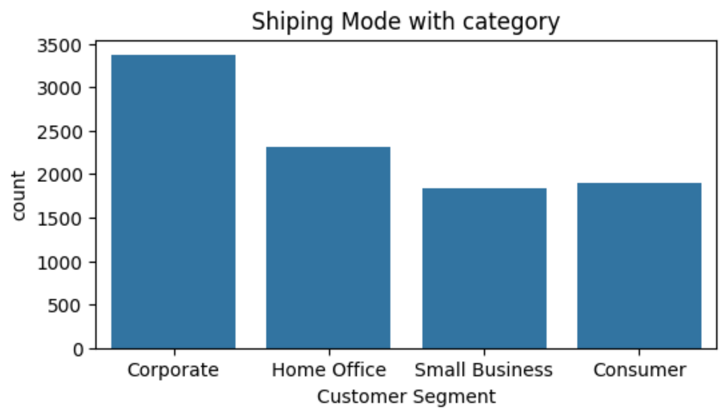
Shiping Mode with category

```
plt.figure(figsize=(4,3))
sns.countplot(x="Ship Mode",data=dataset,hue="Product Category")
plt.title("Shiping Mode with category")
plt.show()
```



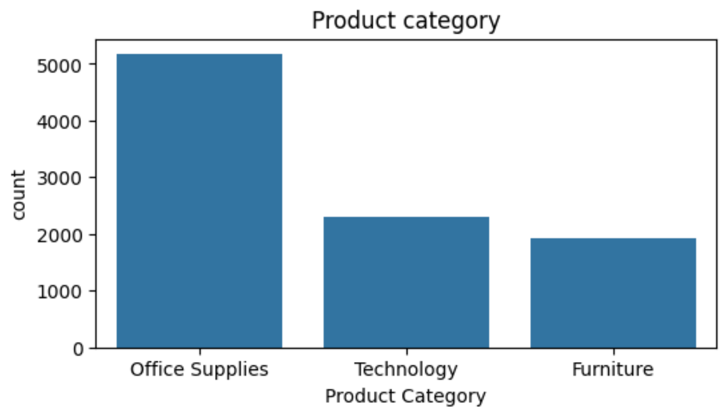
Customer Segment

```
plt.figure(figsize=(6,3))
sns.countplot(x="Customer Segment",data=dataset)
plt.title("Shiping Mode with category")
plt.show()
```



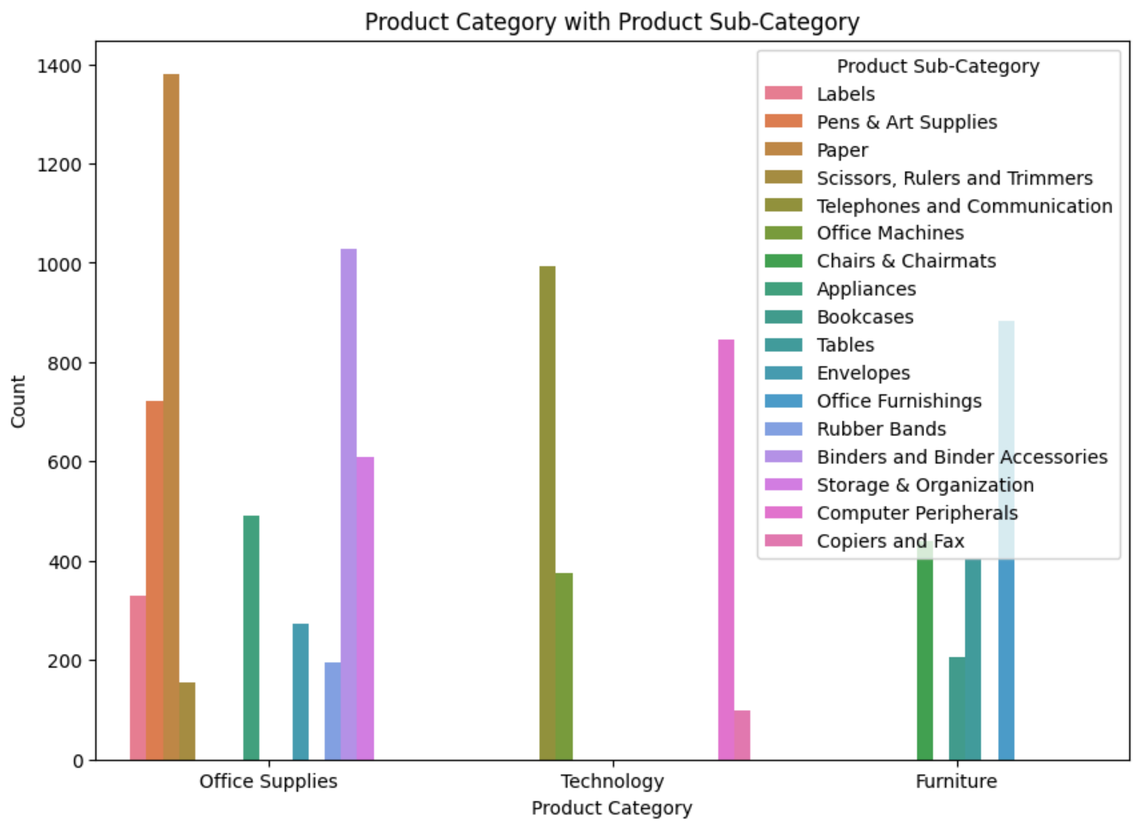
Product Category

```
plt.figure(figsize=(6,3))
sns.countplot(x="Product Category",data=dataset)
plt.title("Product category")
plt.show()
```



Product Sub category with Count

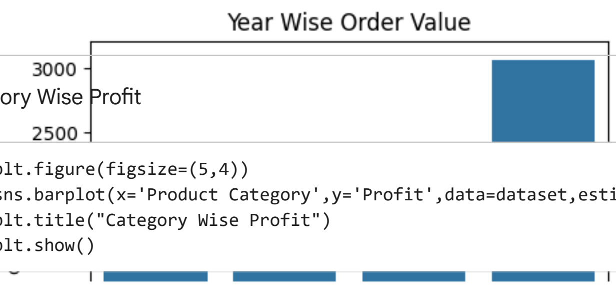
```
plt.figure(figsize=(10,7))
sns.countplot(x="Product Category",data=dataset,hue="Product Sub-Category")
plt.title("Product Category with Product Sub-Category")
plt.xlabel("Product Category")
plt.ylabel("Count")
plt.show()
```



```
dataset['Order Year'].value_counts()
```

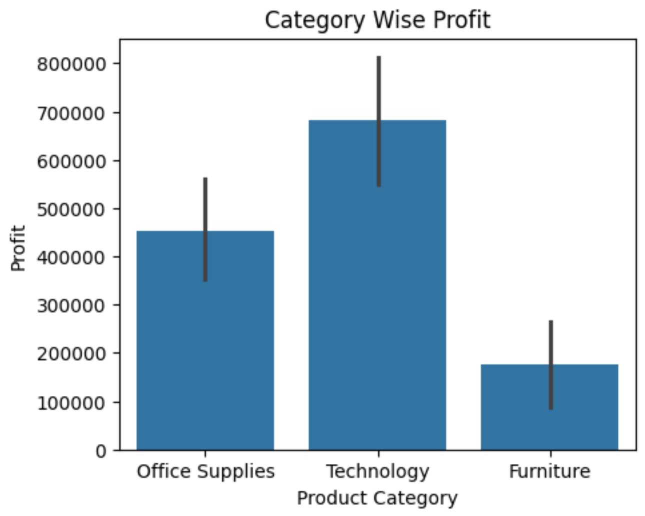
```
            count
Order Year
      2013   3054
      2012   2241
      2011   2179
      2010   1952

dtype: int64
```

```
plt.figure(figsize=(5,4))
sns.countplot(x="Order Year",data=dataset)
plt.title("Year Wise Order Value")
plt.show()
```



Category Wise Profit

```
plt.figure(figsize=(5,4))
sns.barplot(x='Product Category',y='Profit',data=dataset,estimator=sum)
plt.title("Category Wise Profit")
plt.show()
```



State wise Sales /Orders

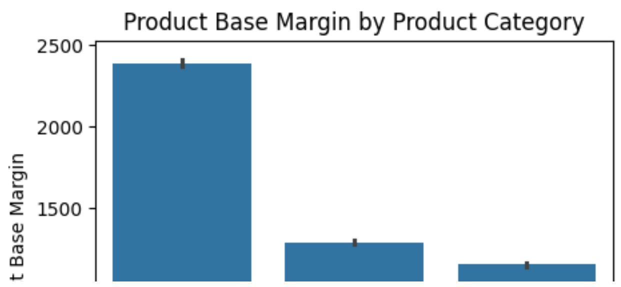
```
dataset['State or Province'].value_counts()[:5]#top five
```

State or Province	count
California	1021
Texas	646
Illinois	584
New York	574
Florida	522

dtype: int64

Product Based Margin with Category

```
plt.figure(figsize=(5,4))
sns.barplot(x='Product Category',y='Product Base Margin',data=dataset,estimator=sum)
plt.title("Product Base Margin by Product Category")
plt.xlabel("Product Category")
plt.ylabel("Product Base Margin")
plt.show()
```


Product Base Margin by Product Category
t Base Margin