Exploratory Data Analysis(SuperStore Sales EDA)

About Data

We have retail dataset of a global superstore for 4 the last years which has been obtained from Kaggle.

We have last four years data from previous sales nationwide. While checking the shape of the data we find out that it consists of 9800 Rows and 18 columns. Taking a look at the data structure we find that it mostly contains objects and has only two numeric values data. Upon further investigation we found 11 missing values in the postal code category and we filled it with real values of the actual location.

Features

- row_id int64
- order_id object
- order_date object
- ship_date object
- ship_mode object
- customer_id object
- customer_name object
- segment object
- country object
- city object
- state object
- postal_code int64
- region object
- product_id object
- category object
- sub_category object
- product_name object
- sales float64

Data Source: SuperStore Sales Data

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

Step-1:

Importing dataset

```
# Step-1: Importing Dataset
df = pd.read_csv('train1.csv')
```

```
df.head()
```

Date shape

```
df.shape
```

> Output

```
(9800, 18)
```

Step-2:

```
# Data Structure
df.info()
```

> Output

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9800 entries, 0 to 9799
Data columns (total 17 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   order_id       9800 non-null   object
 1   order_date     9800 non-null   object
 2   ship_date      9800 non-null   object
 3   ship_mode      9800 non-null   object
 4   customer_id    9800 non-null   object
 5   customer_name  9800 non-null   object
 6   segment        9800 non-null   object
 7   country        9800 non-null   object
 8   city           9800 non-null   object
 9   state          9800 non-null   object
 10  postal_code    9789 non-null   float64
 11  region         9800 non-null   object
 12  product_id     9800 non-null   object
 13  category       9800 non-null   object
 14  sub_category   9800 non-null   object
 15  product_name   9800 non-null   object
 16  sales          9800 non-null   float64
dtypes: float64(2), object(15)
memory usage: 1.3+ MB
```

Step-3: Finding missing values

Only 11 values were missing in this dataset in the postal code column and we filled it with real postal code.

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

> OutPut

```
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     0
region          0
product_id      0
category        0
sub_category    0
product_name    0
sales           0
dtype: int64
```

Step-4: Summary Statistics

```
df.describe()
```

> Output

```
    row_id  postal_code sales
count   9800.000000 9800.000000 9800.000000
mean    4900.500000 55217.343265    230.769059
std    2829.160653  32066.750532    626.651875
min     1.000000    1040.000000 0.444000
25%    2450.750000  23223.000000    17.248000
50%   4900.500000   57551.000000    54.490000
75%   7350.250000   90008.000000    210.605000
max   9800.000000   99301.000000    22638.480000
```

```
df.columns
```

> Output

```
Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'country', 'city', 'state',
       'postal_code', 'region', 'product_id', 'category', 'sub_category',
       'product_name', 'sales'],
      dtype='object')
```

Step-5: Value count of a specific column

```
df['postal_code'].value_counts().head()
```

> Output

```
10035    253
10024    225
10009    220
94122    195
10011    193
Name: postal_code, dtype: int64
```

Step-6: Unique Values of Specific Columns

```
print(df['category'].unique())
```

> Output

```
['Furniture' 'OfficeSupplies' 'Technology']
```

```
# Finding unique values in a column
df['ship_mode'].unique()
```

> Output

```
```

```
array(['SecondClass', 'StandardClass', 'FirstClass', 'SameDay'],
      dtype=object)
```

```
df['ship_mode'].value_counts()
```

Output

```
StandardClass    5859
SecondClass      1902
FirstClass       1501
SameDay           538
Name: ship_mode, dtype: int64
```

```
df['region'].value_counts()
```

Output

```
West       3140
East       2785
Central    2277
South      1598
Name: region, dtype: int64
```

Most orders are coming form the Western region, followed by East and central. Least orders are coming from the South.

```
df.sort_values(['region'],ascending=True).groupby('region').sum()
```

Output

```
    row_id  postal_code sales
region
Central 11230808    149080099    492646.9132
East    13450300    48962469     669518.7260
South   7995925         55117795     389151.4590
West    15347867    287969601    710219.6845
```

```
df['state'].value_counts().head()
```

> Output

```
California      1946
NewYork         1097
Texas            973
Pennsylvania     582
Washington       504
Name: state, dtype: int64
```

```
df.groupby(['state']).sum()['sales'].nlargest()
```

> Output

```
 df.groupby(['state']).sum()['sales'].nlargest()
state
California      446306.4635
NewYork         306361.1470
Texas           168572.5322
Washington      135206.8500
Pennsylvania    116276.6500
Name: sales, dtype: float64
```

Step-7: Deal with duplicates

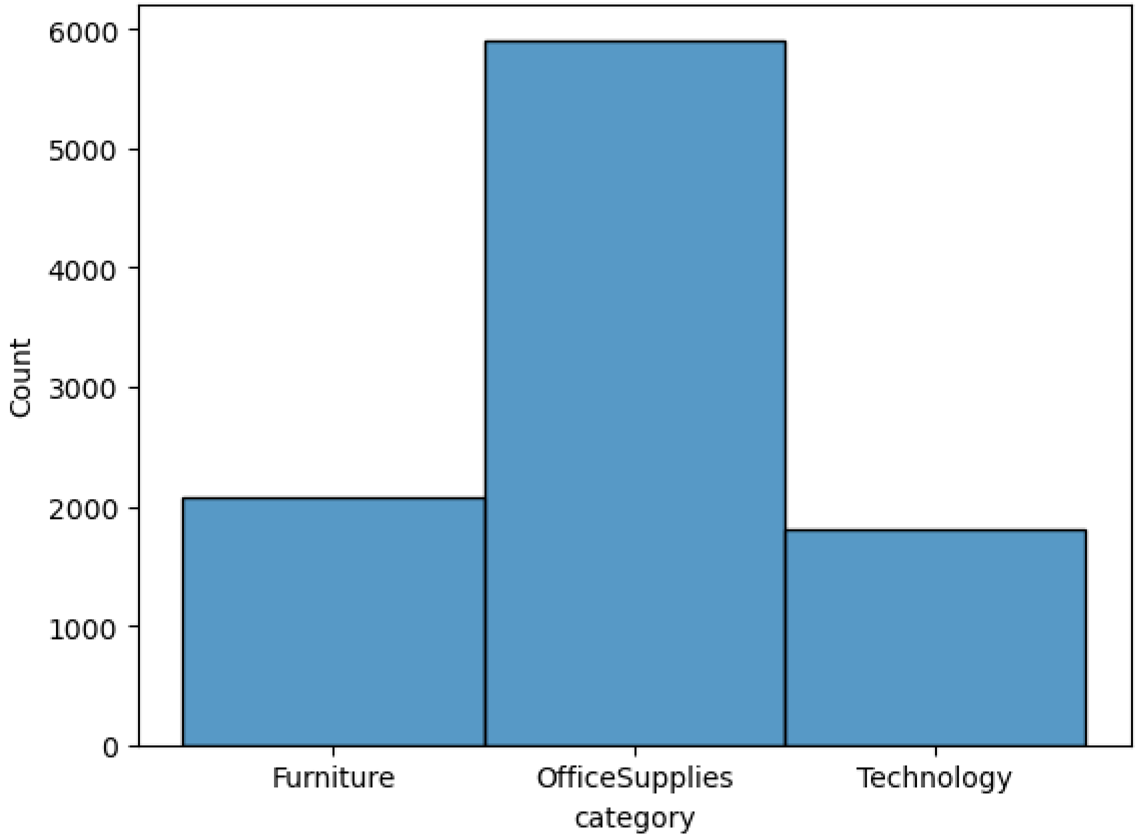
```
df.sample(10)
```

Step-8: Checking the normality / Standard normal distribution

Number of categorise

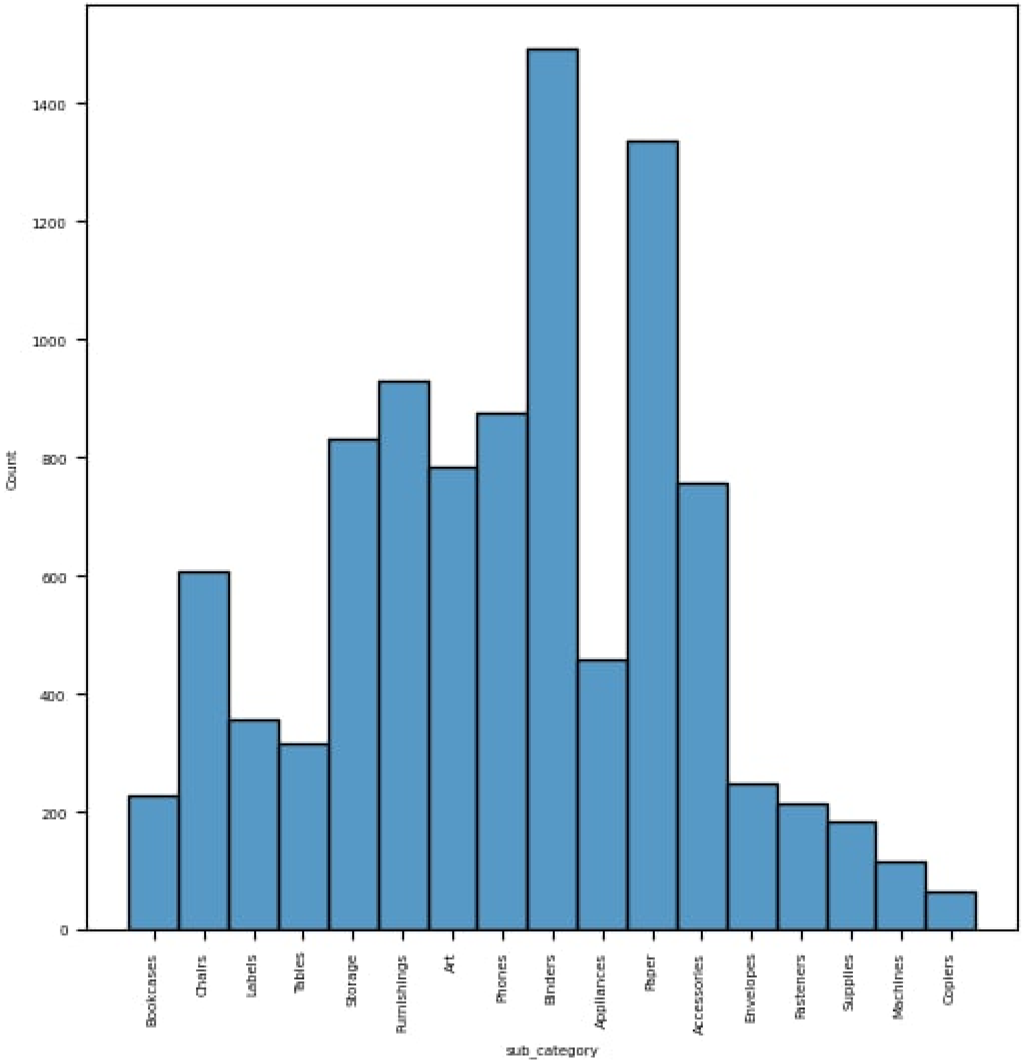
Here is the graph showing that the officeSupplies category has the maximum sale. while the Furniture and Technology both categorize have comparatively sales. So we can say the the customers nearby are using office supplies more than other both categorize.

```
sns.histplot(df['category'])
```



From the above graph , it’s very much clear that Binders and Papers are the hot selling products. so these products can provide more sale and revenue. While the Copiers and Machines Subcategory needs improvement.

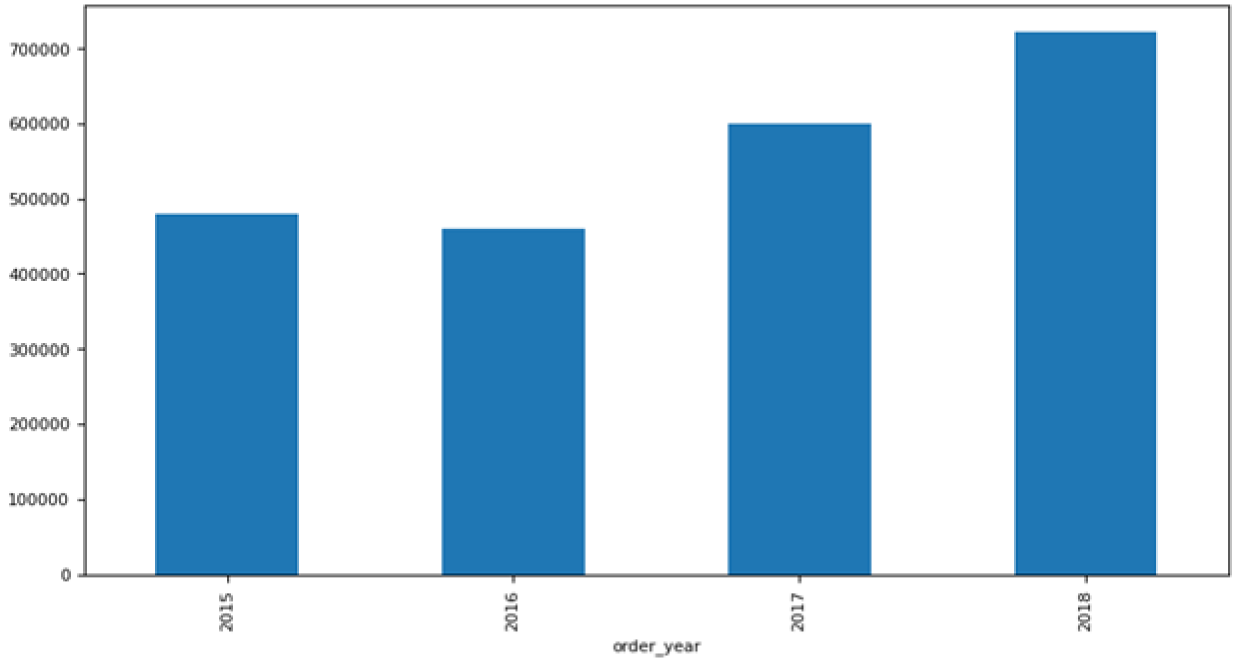
```
plt.figure(figsize=(6,6))
plt.xticks(rotation='vertical')
sns.histplot(df['sub_category'])
```



Sales Over the years

This graph is showing sales over the years, and we can see that sales are increasing over the years.

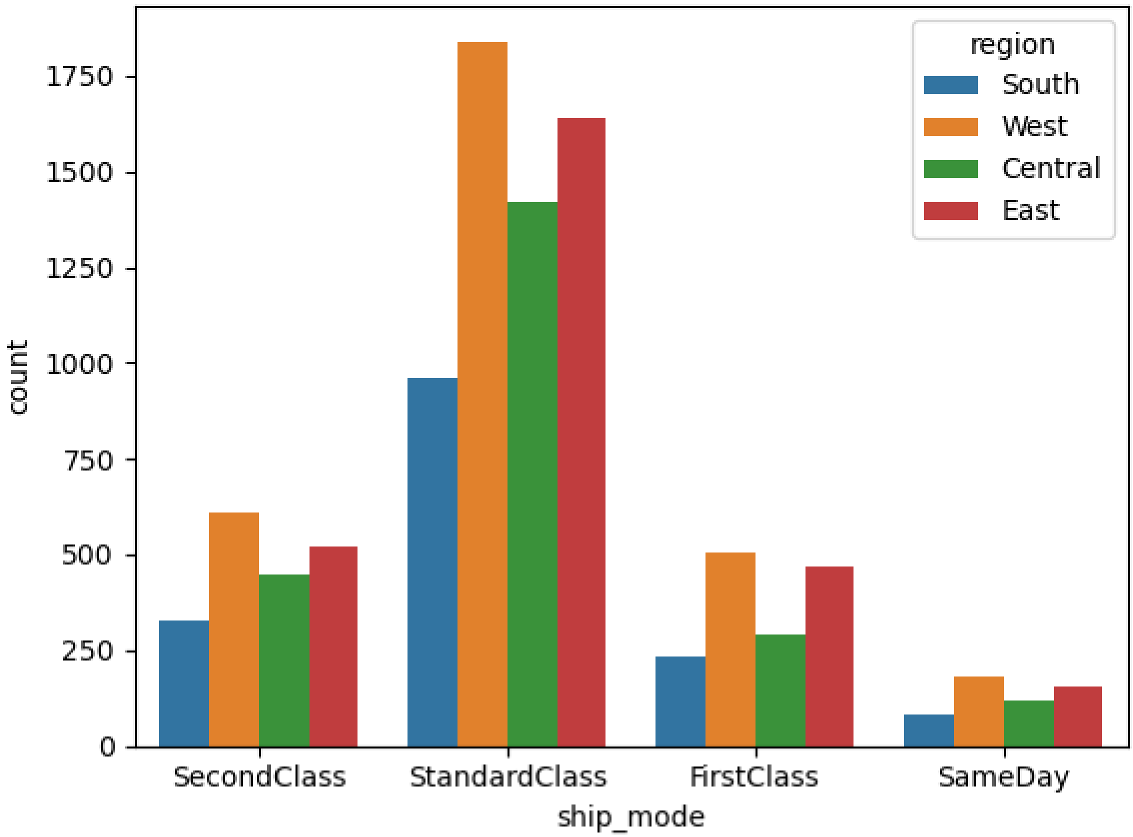
```
plt.figure(figsize=(10,5))
df['order_year'] = df['order_date'].dt.year
df.groupby('order_year')['sales'].sum().plot(kind='bar')
```



Number of Shipments in Different Regions

This graph is showing number of sales over order by ship mode - (in-numbers).
Most of the order were made through standard class, it also shows that the region 'west' is having most of the sales.

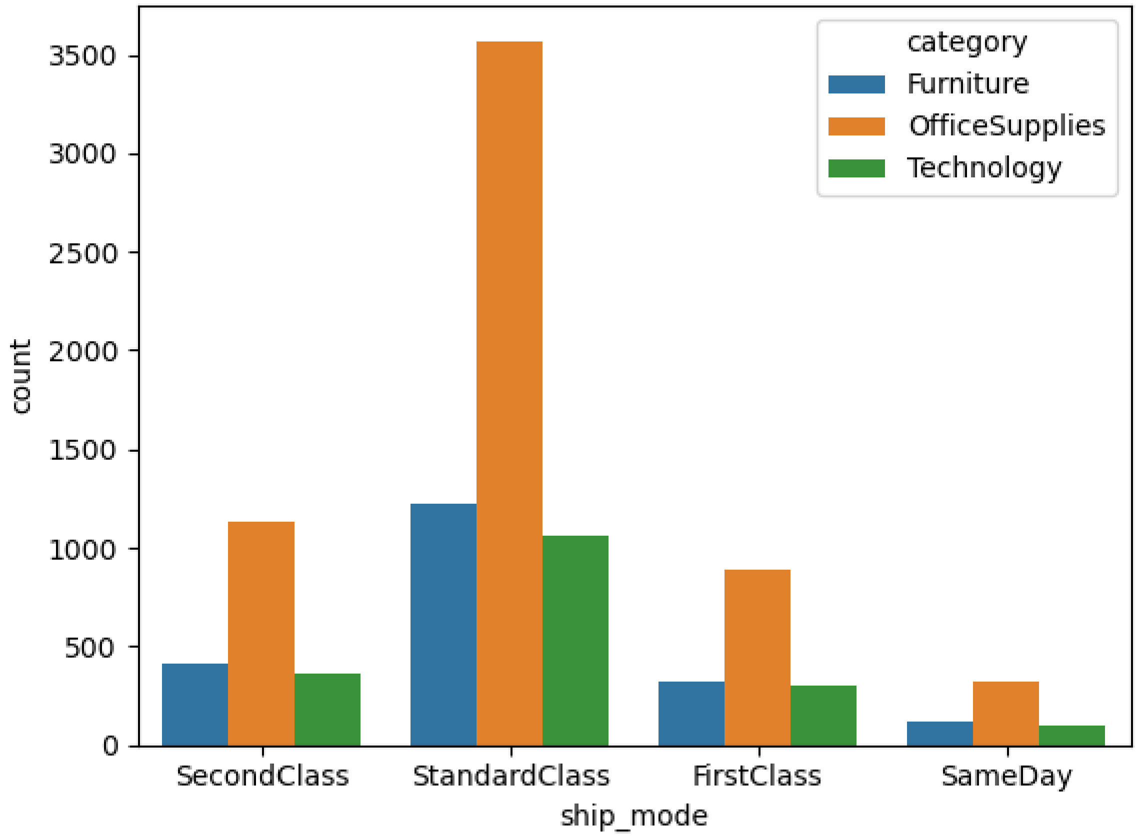
```
sns.countplot(x = df['ship_mode'], hue = df['region'])
```



Shipment by Class

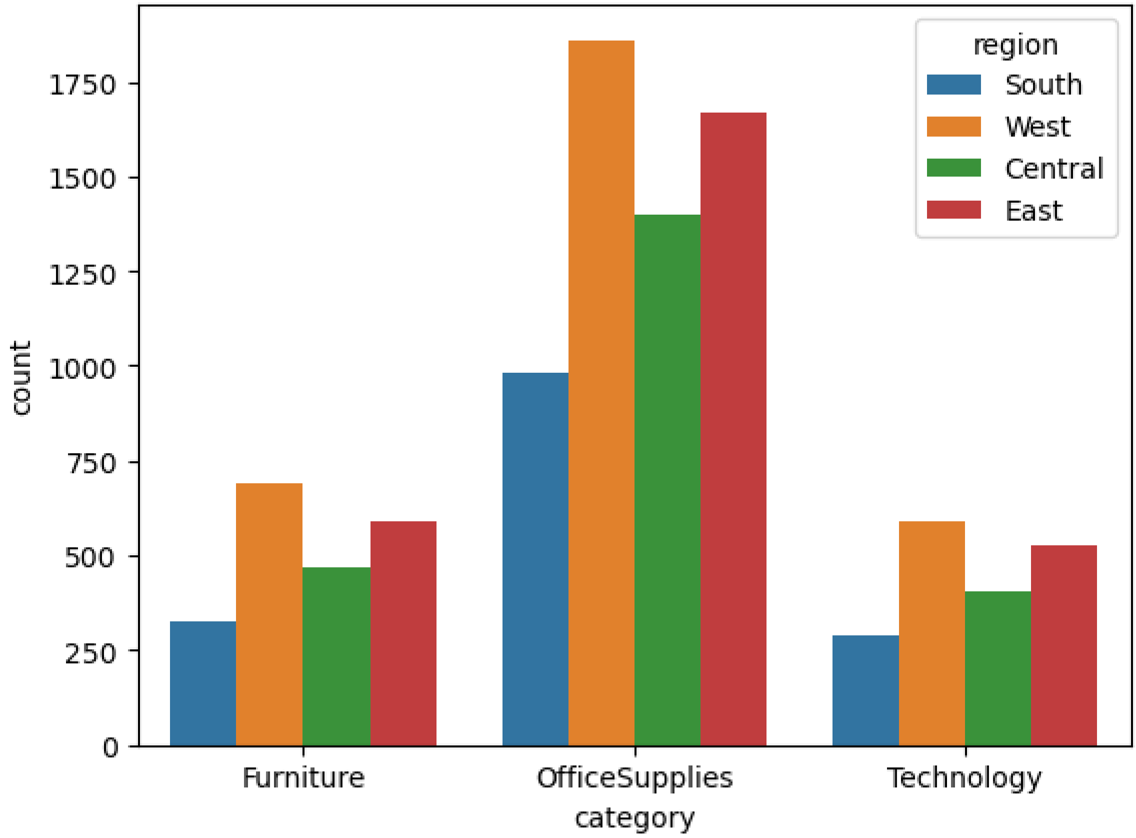
This graph is showing comparison of product categories and shipment_mode -(in numbers). This graph is showing comparison of product categories and shipment_mode, and we can see that the product category 'office supplies' is having most of the sales.

```
sns.countplot(x = df['ship_mode'], hue = df['category'])
```



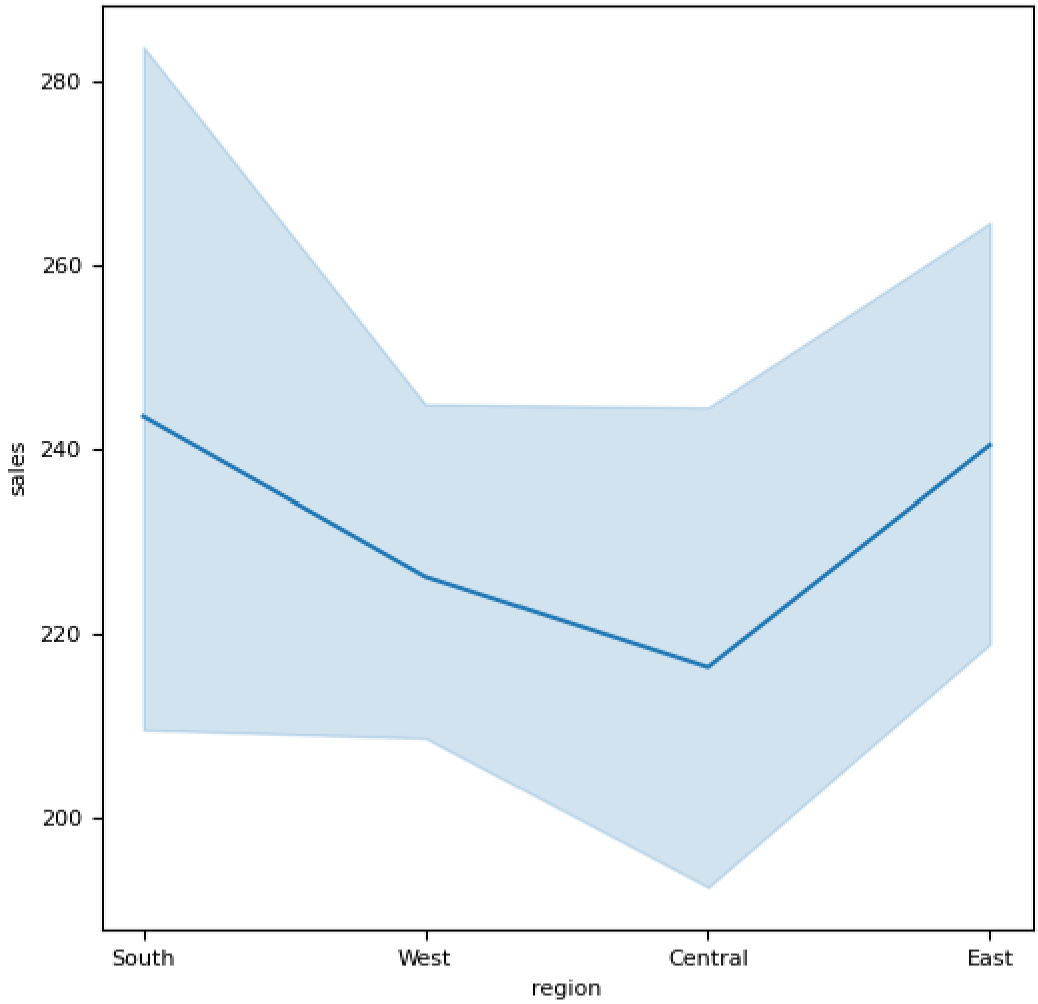
> This graph is showing comparison of categories VS region.

```
sns.countplot(x = df['category'], hue = df['region'])
```



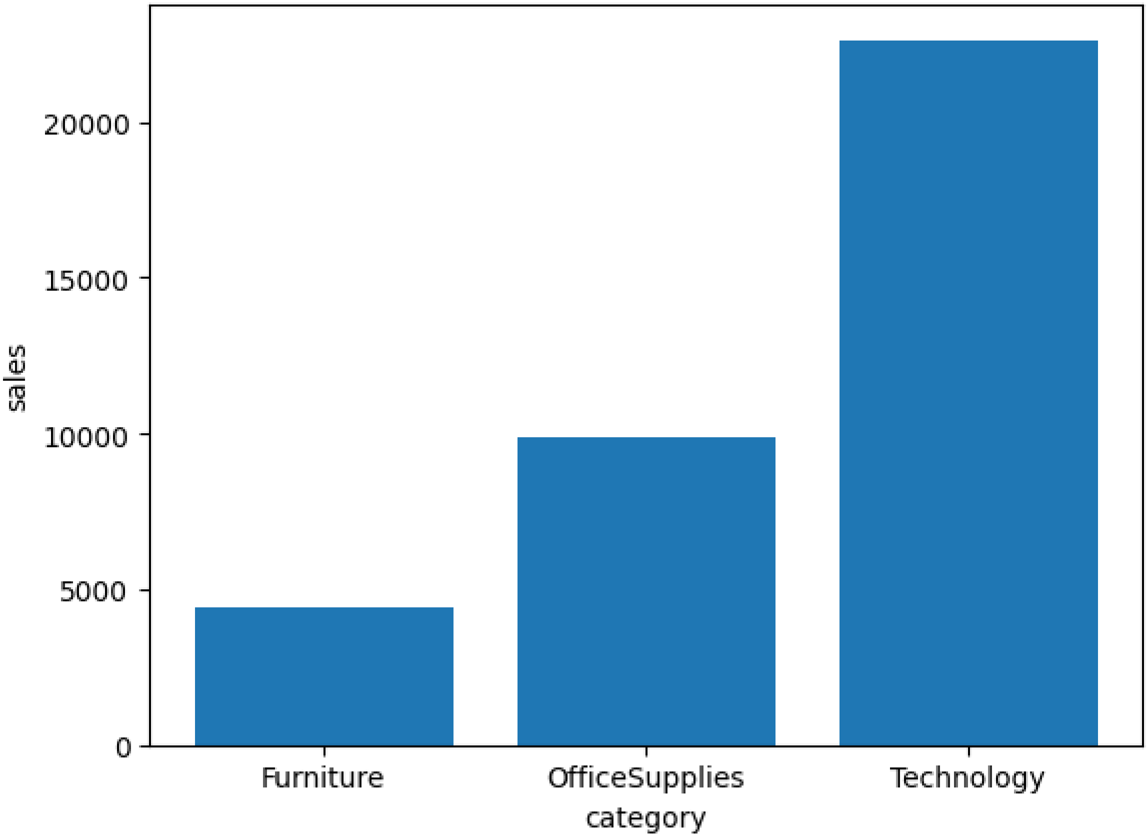
> This graph is showing volume of sales in different regions.

```
sns.lineplot(x=df["region"],y=df["sales"])
```



> This graph is showing highest selling categories.

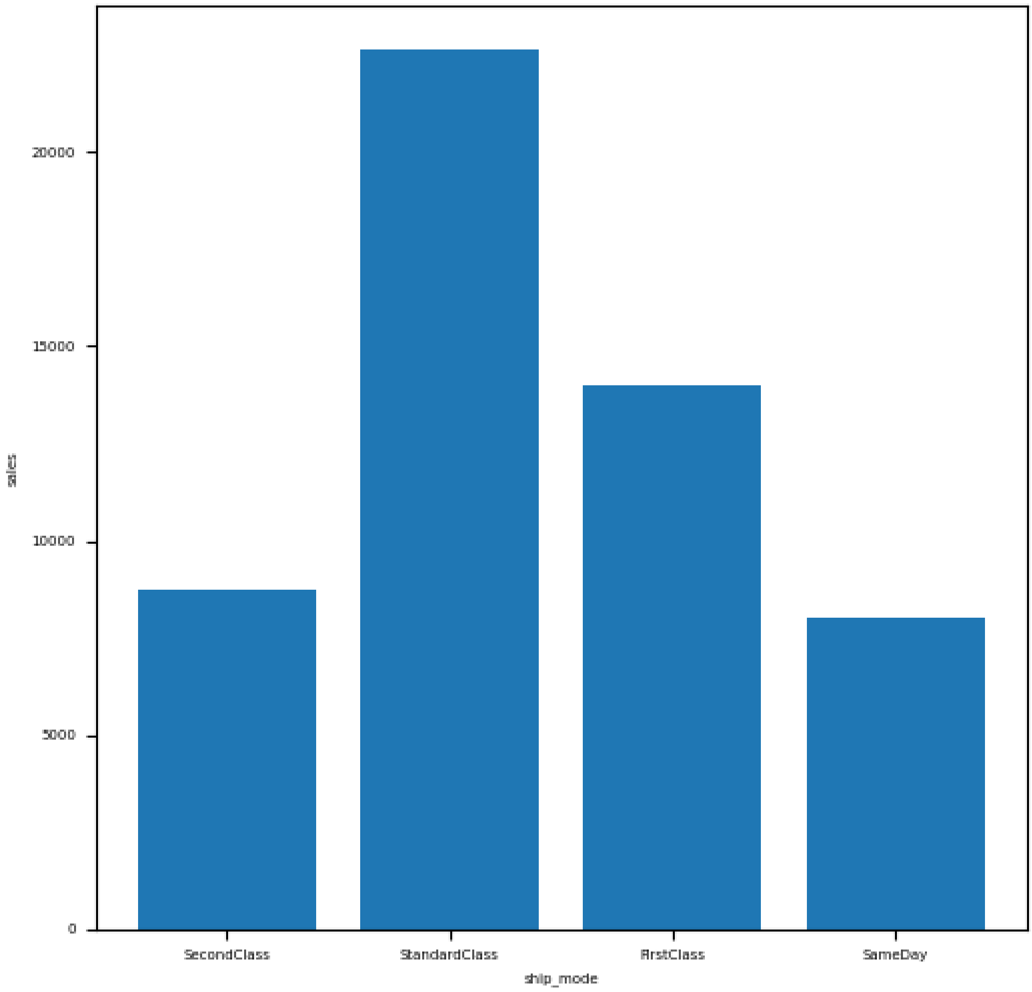
```
plt.bar(df['category'],df['sales'])
plt.rcParams.update({'font.size':5})
plt.xlabel('category')
plt.ylabel('sales')
```



Comparison Vs Sales and Ship

Shipments and retail sales for a single retail customer must be extremely close, particularly over an extended period. Even for covered channels, coverage is only sometimes 100 percent. In this scenario, we have to calculate a coverage factor that can be applied to the retail sales data to align it with shipments. The graph illustrates the relationship between shipment and sales. The standard class has the highest sale ratio, while same-day and second-class sales are nearly identical.

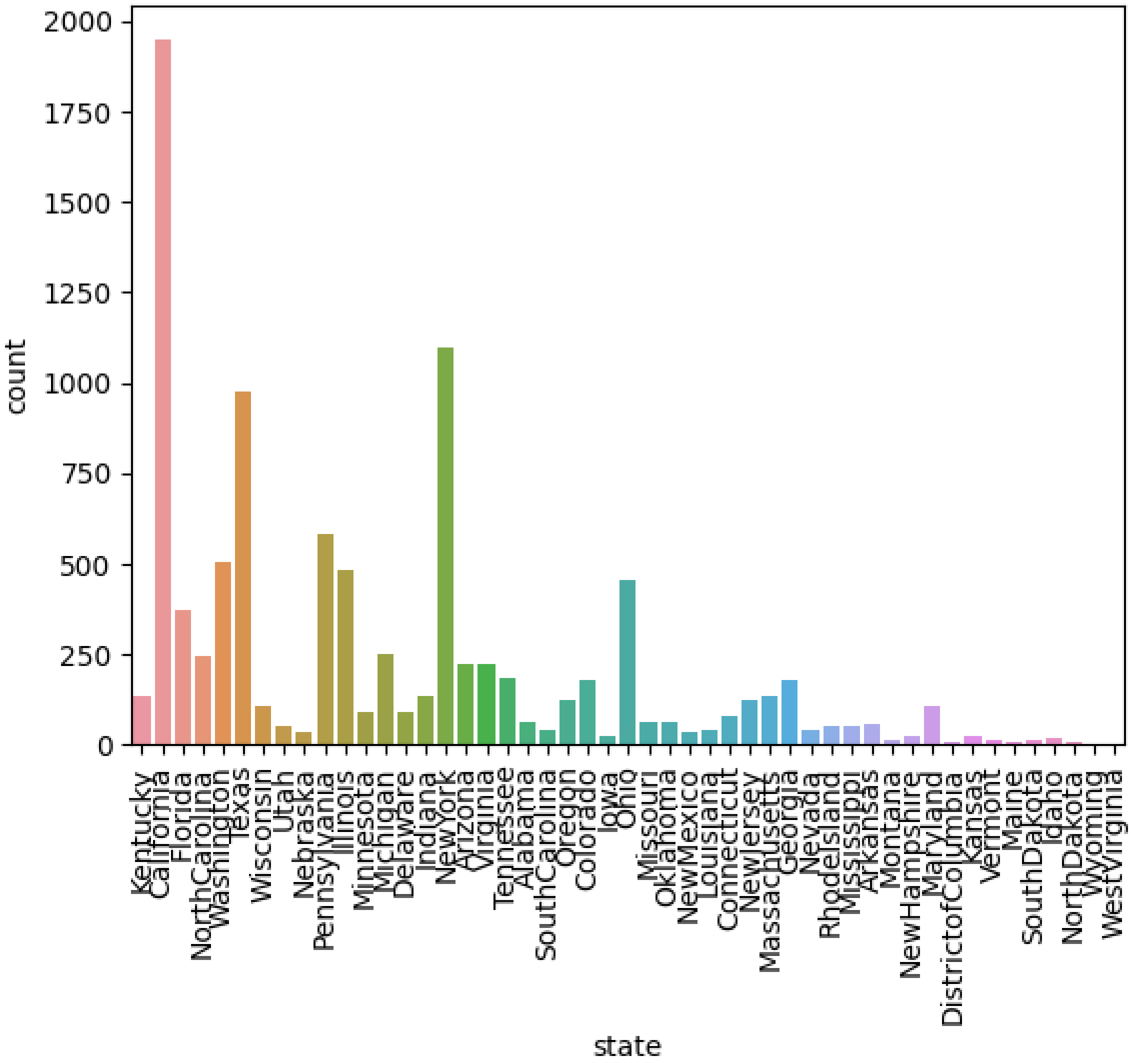
```
plt.rcParams['figure.figsize']=(6,6)
plt.bar(df['ship_mode'],df['sales'])
plt.rcParams.update({'font.size':8})
plt.xlabel('ship_mode')
plt.ylabel('sales')
```



State wide sale.

According to this bar chart, the vast majority of sales occur in California, approximately 1900 followed by New York and Texas.

```
plt.figure(figsize=(20, 10))
plt.xticks(rotation='vertical')
sns.countplot(x='state',data=df)
```




```
corr = df.corr(method='pearson')
corr
```

Results Deduced from the Graphs

1. Our customers are making orders mostly through the 'Standard Shipping' option, followed by 'Second Class', then 'First Class'.
2. Least order were made using 'Same Day' shipping option.
3. Most of the orders are coming from the 'West' region.
4. Least orders are coming from 'East" region.
5. The most shipped category is 'Office Supplies'
6. Majority of our customers are based in California, followed by New York and Texas
7. Over 90% of our sales orders are in the Range of 1-5000.
8. Around 50% of our sales comes from Technology related products.
9. Around 30% of sales comes from Office Supplies.
10. Around 20% of sales is from the Furniture category.