import library

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

creat the data set

```
np.random.seed(42)

data = {
    'product_id'  : range(1,21),
    'product_name' : [f'Product{i}' for i in range(1,21)],
    'category'  : np.random.choice(['Electronic','Clothing' , 'Home' ,
'Sports'],20),
    'units_sold' : np.random.poisson(lam=20, size=20),
    'sales_date' : pd.date_range (start = '2023-01-01' ,
periods=20 ,freq='D')
}

sales_data = pd.DataFrame(data)
print(sales_data)
```

```
    product_id product_name    category  units_sold sales_date
0            1     Product1        Home          25 2023-01-01
1            2     Product2      Sports          15 2023-01-02
2            3     Product3  Electronic          17 2023-01-03
3            4     Product4        Home          19 2023-01-04
4            5     Product5        Home          21 2023-01-05
5            6     Product6      Sports          17 2023-01-06
6            7     Product7  Electronic          19 2023-01-07
7            8     Product8  Electronic          16 2023-01-08
8            9     Product9        Home          21 2023-01-09
9           10    Product10    Clothing          21 2023-01-10
10          11    Product11        Home          17 2023-01-11
11          12    Product12        Home          22 2023-01-12
12          13    Product13        Home          14 2023-01-13
13          14    Product14        Home          17 2023-01-14
14          15    Product15      Sports          17 2023-01-15
15          16    Product16  Electronic          21 2023-01-16
16          17    Product17      Sports          21 2023-01-17
17          18    Product18      Sports          13 2023-01-18
```

```
18          19    Product19      Sports          18 2023-01-19
19          20    Product20        Home          25 2023-01-20
```

```
sales_data.to_csv('sales_data.csv', index= False)
```

```
import os
os.getcwd()
```

```
'c:\\Users\\Appala nithin\\Downloads\\STATS-WORKSHOP'
```

descriptive stats

```
descriptive_stats = sales_data['units_sold'].describe()

print("\nDescriptive statistic for Units Sold:")
print(descriptive_stats)

mean_sales=sales_data['units_sold'].mean()
median_sales=sales_data['units_sold'].median()
mode_sales=sales_data['units_sold'].mode()[0]
variance_sales=sales_data['units_sold'].var()
std_deviation_sales=sales_data['units_sold'].std()

print("\nCategory Statistics")
category_stats=sales_data.groupby('category')
['units_sold'].agg(['sum','mean','std']).reset_index()
print(category_stats)
print("\nStatistical Analysis:")
print(f"Mean Units Sold: {mean_sales}")
print(f"Meadian Units Sold: {median_sales}")
print(f"Mode Units Sold: {mode_sales}")
print(f"Variance Units Sold: {variance_sales}")
print(f"Standard Deviation Units Sold: {std_deviation_sales}")
```

```
Descriptive statistic for Units Sold:
count    20.000000
mean     18.800000
std       3.302312
min      13.000000
25%      17.000000
50%      18.500000
75%      21.000000
max      25.000000
Name: units_sold, dtype: float64

Category Statistics
     category  sum       mean       std
```

```
0    Clothing   21  21.000000       NaN
1  Electronic   73  18.250000  2.217356
2        Home  181  20.111111  3.723051
3      Sports  101  16.833333  2.714160

Statistical Analysis:
Mean Units Sold: 18.8
Meadian Units Sold: 18.5
Mode Units Sold: 17
Variance Units Sold: 10.905263157894737
Standard Deviation Units Sold: 3.302311789927586
```

Inferential Statistics

```
confidence_level = 0.95
degrees_freedom = len(sales_data['units_sold'])-1
sample_mean = mean_sales
sample_standard_error = std_deviation_sales / 
np.sqrt(len(sales_data['units_sold']))

#T-score

t_score = stats.t.ppf((1 + confidence_level) /2, degrees_freedom)
margin_of_error = t_score * sample_standard_error

confidence_interval = (sample_mean - margin_of_error,sample_mean + 
margin_of_error)
print("\nConfidence Interval for Mean of Units Sold")
print(confidence_interval)
```

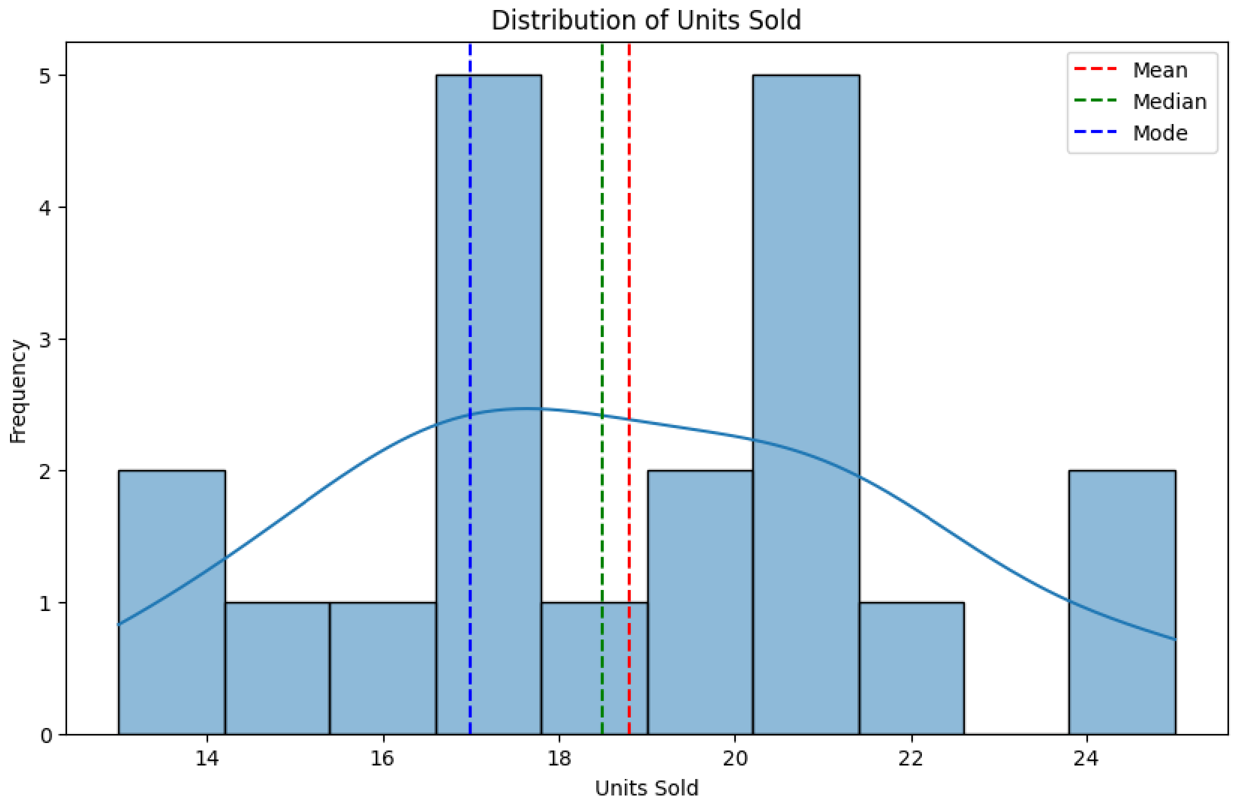
```
Confidence Interval for Mean of Units Sold
(np.float64(17.254470507823573), np.float64(20.34552949217643))
```

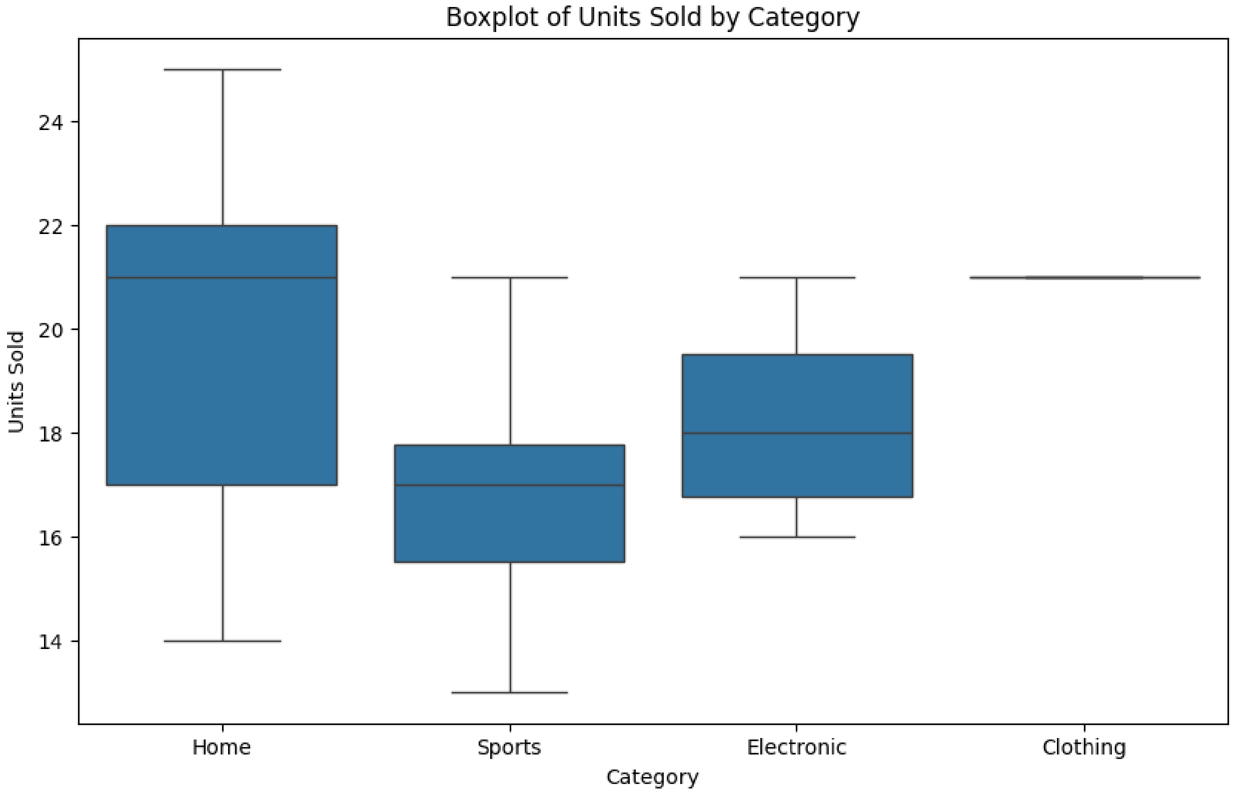
```
t_statistic, p_value = stats.ttest_1samp(sales_data['units_sold'],20)

print('\n Hypothesis Testing (t-test):')
print(f'T-statistic: {t_statistic},P-value:{p_value}')

if p_value < 0.05:
    print('Reject The Null Hyptohesis: The  mean units sold is 
different from 20')
else:
    print('Fail to Reject the Null Hypothesis: The mean units sold is 
not different from 20')
```

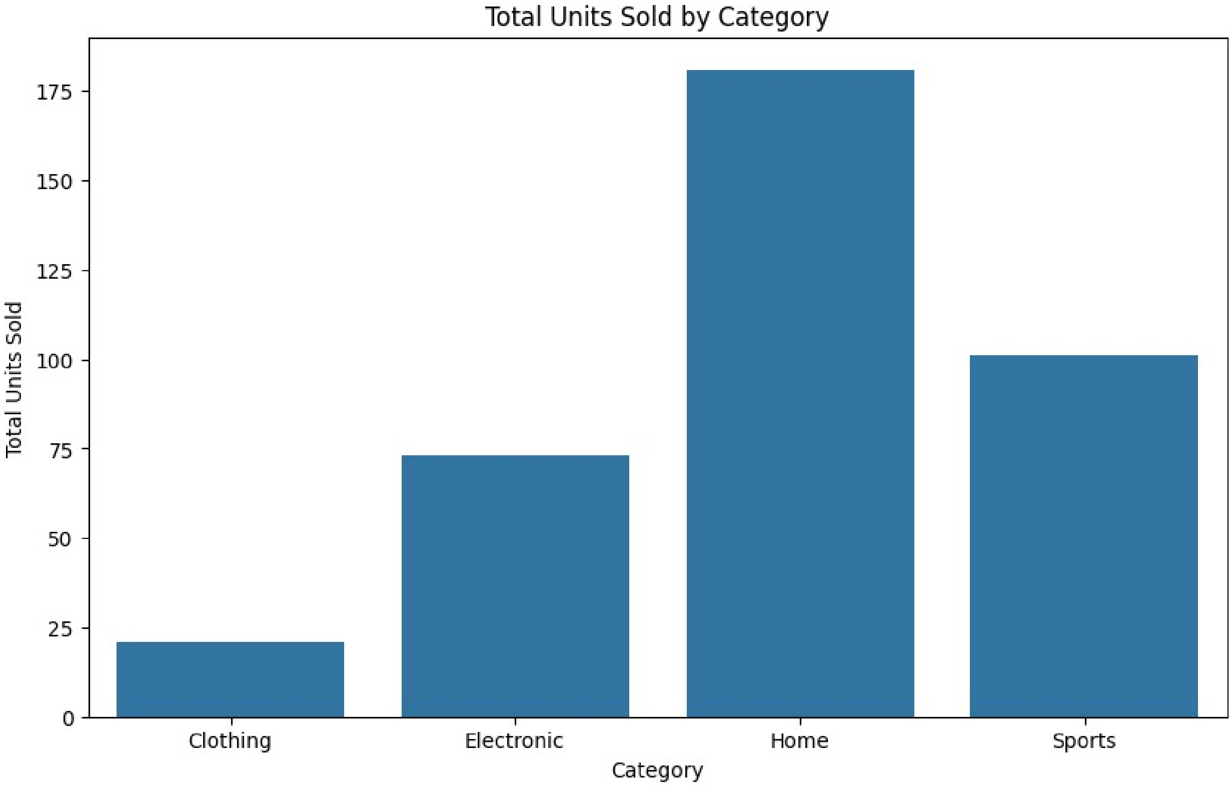
```
 Hypothesis Testing (t-test):
T-statistic: -1.6250928099424466,P-value:0.12061572226781002
Fail to Reject the Null Hypothesis: The mean units sold is not different from 20
```

Visualization

```python
# Plot distribution of units sold
plt.figure(figsize=(10, 6))
sns.histplot(sales_data['units_sold'], bins=10, kde=True)
plt.title('Distribution of Units Sold')
plt.xlabel('Units Sold')
plt.ylabel('Frequency')

# Add lines for mean, median, and mode
plt.axvline(mean_sales, color='red', linestyle='--', label='Mean')
plt.axvline(median_sales, color='green', linestyle='--', label='Median')
plt.axvline(mode_sales, color='blue', linestyle='--', label='Mode')  # Corrected duplicate entry
plt.legend()
plt.show()

# Boxplot for units sold by category
plt.figure(figsize=(10, 6))
sns.boxplot(x='category', y='units_sold', data=sales_data)
plt.title('Boxplot of Units Sold by Category')
plt.xlabel('Category')
plt.ylabel('Units Sold')
plt.show()

# Bar plot (Corrected column names)
plt.figure(figsize=(10, 6))
sns.barplot(x='category', y='sum', data=category_stats)  # Changed 'TotaL Units Sold' to 'sum'
plt.title('Total Units Sold by Category')
plt.xlabel('Category')
plt.ylabel('Total Units Sold')
plt.show()
```


Distribution of Units Sold
Mean
Median
Mode
Frequency
0
1
2
3
4
5
14
16
18
20
22
24
Units Sold


Boxplot of Units Sold by Category
Units Sold
14
16
18
20
22
24
Home
Sports
Electronic
Clothing
Category


Total Units Sold by Category
Total Units Sold
0
25
50
75
100
125
150
175
Clothing
Electronic
Home
Sports
Category