Untitled

November 27, 2023

[35]:
```
import numpy as np
import pandas as pd
df=pd.read_csv('1673872777_ausapparalsales4thqrt2020.csv')
print(df)
df.head()
```

```
           Date        Time State     Group  Unit  Sales
0      1-Oct-20     Morning    WA      Kids     8  20000
1      1-Oct-20     Morning    WA       Men     8  20000
2      1-Oct-20     Morning    WA     Women     4  10000
3      1-Oct-20     Morning    WA   Seniors    15  37500
4      1-Oct-20   Afternoon    WA      Kids     3   7500
…           …           …   …       …   …     …
7555  30-Dec-20   Afternoon   TAS   Seniors    14  35000
7556  30-Dec-20     Evening   TAS      Kids    15  37500
7557  30-Dec-20     Evening   TAS       Men    15  37500
7558  30-Dec-20     Evening   TAS     Women    11  27500
7559  30-Dec-20     Evening   TAS   Seniors    13  32500

[7560 rows x 6 columns]
```

[35]:

	Date	Time	State	Group	Unit	Sales
0	1-Oct-20	Morning	WA	Kids	8	20000
1	1-Oct-20	Morning	WA	Men	8	20000
2	1-Oct-20	Morning	WA	Women	4	10000
3	1-Oct-20	Morning	WA	Seniors	15	37500
4	1-Oct-20	Afternoon	WA	Kids	3	7500

[9]:
```
df.isna()
```

[9]:

	Date	Time	State	Group	Unit	Sales
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
…	…	…	…	…	…	…
7555	False	False	False	False	False	False

```
7556  False  False  False  False  False  False
7557  False  False  False  False  False  False
7558  False  False  False  False  False  False
7559  False  False  False  False  False  False

[7560 rows x 6 columns]
```

[8]:
```
df.notna()
```

[8]:
```
      Date  Time  State  Group  Unit  Sales
0     True  True   True   True  True   True
1     True  True   True   True  True   True
2     True  True   True   True  True   True
3     True  True   True   True  True   True
4     True  True   True   True  True   True
…      …     …      …      …     …      …
7555  True  True   True   True  True   True
7556  True  True   True   True  True   True
7557  True  True   True   True  True   True
7558  True  True   True   True  True   True
7559  True  True   True   True  True   True

[7560 rows x 6 columns]
```

[10]:
```
df.isna().sum()
```

[10]:
```
Date     0
Time     0
State    0
Group    0
Unit     0
Sales    0
dtype: int64
```

[5]:
```
df.mean()
```

```
/tmp/ipykernel_71/3698961737.py:1: FutureWarning: The default value of
numeric_only in DataFrame.mean is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  df.mean()
```

[5]:
```
Unit        18.005423
Sales    45013.558201
dtype: float64
```

0.0.1 Perform descriptive statistical analysis on the data (Sales and Unit columns) (Techniques such as mean, median, mode and standard deviation can be used.)

[11]:
```
df_Sales= df['Sales'].mean()
print(df_Sales)
```

```
45013.5582010582
```

[12]:
```
df_Sales= df['Sales'].median()
print(df_Sales)
```

```
35000.0
```

[13]:
```
df_Sales= df['Sales'].mode()
print(df_Sales)
```

```
0    22500
Name: Sales, dtype: int64
```

[14]:
```
df_Unit= df['Unit'].mean()
print(df_Unit)
```

```
18.00542328042328
```

[15]:
```
df_Unit= df['Unit'].median()
print(df_Unit)
```

```
14.0
```

[16]:
```
df_Unit= df['Unit'].mode()
print(df_Unit)
```

```
0    9
Name: Unit, dtype: int64
```

[43]:
```
df_Sales = df['Sales'].std()
print(df_Sales)
```

```
32253.506943966073
```

0.0.2 Determine which group is generating the highest sales, and which group is generating the lowest sales

[42]:
```
Group_Sales = df.groupby('Group')['Sales'].sum()
max_group = Group_Sales.max()
min_group = Group_Sales.min()
print(f"Group with the highest sales: {max_group}")
print(f"Group with the lowest sales: {min_group}")
```

```
Group with the highest sales: 85750000
Group with the lowest sales: 84037500
```

0.0.3 Determine which state is generating the highest sales, and which state is generating the lowest sales.

[45]:
```
State_Sales = df.groupby('State')['Sales'].sum()
max_State = State_Sales.max()
min_State = State_Sales.min()
print(f"State with the highest sales: {max_State}")
print(f"State with the lowest sales: {min_State}")
```

```
State with the highest sales: 105565000
State with the lowest sales: 22152500
```

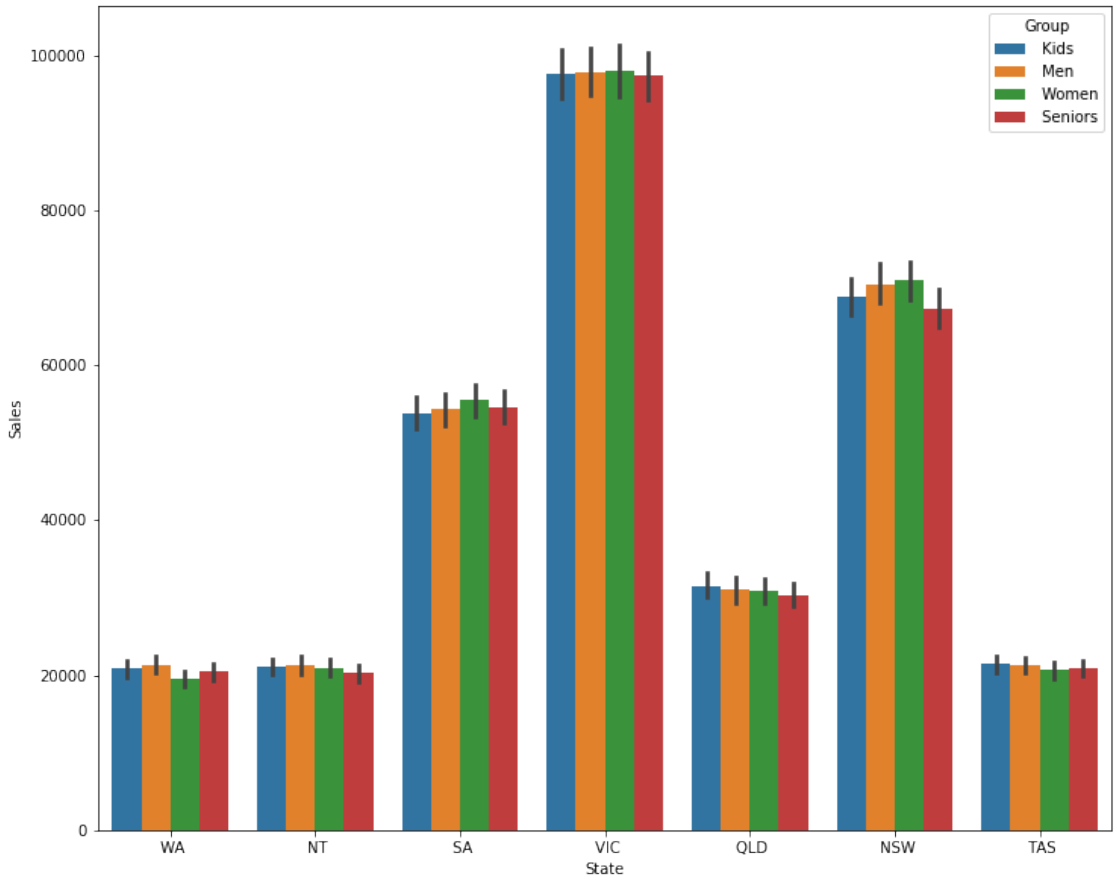
0.0.4 Generate weekly, monthly and quarterly reports for the analysis made.

[ ]:
```

```

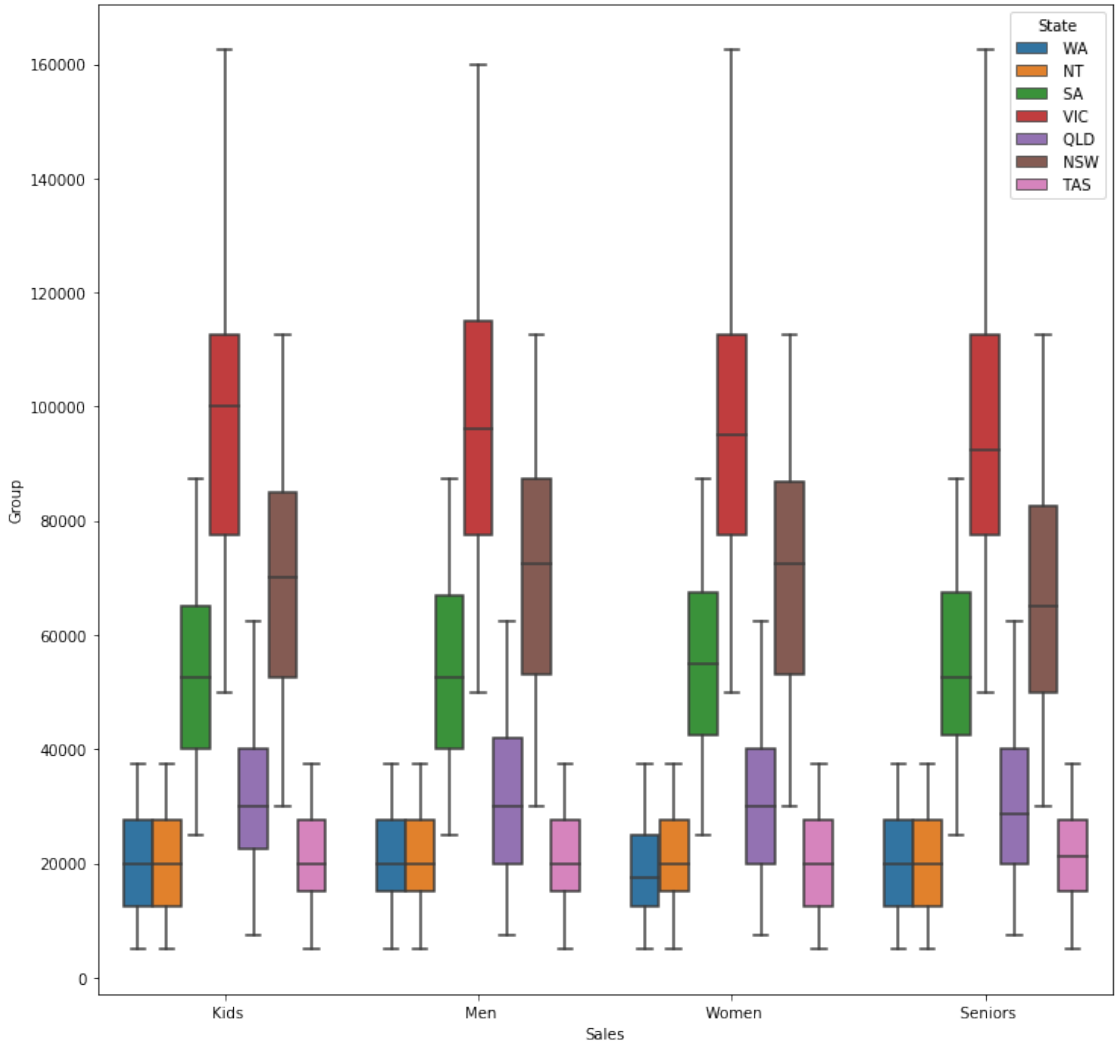
0.0.5 State-wise sales analysis for different groups (kids, women, men, and seniors)

[45]:
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(12,10))
sns.barplot(x="State", y="Sales", hue="Group", data=df)
plt.xlabel('State')
plt.ylabel('Sales')
plt.show()
```



0.0.6 Group-wise sales analysis (kids, women, men, and seniors) across different states.

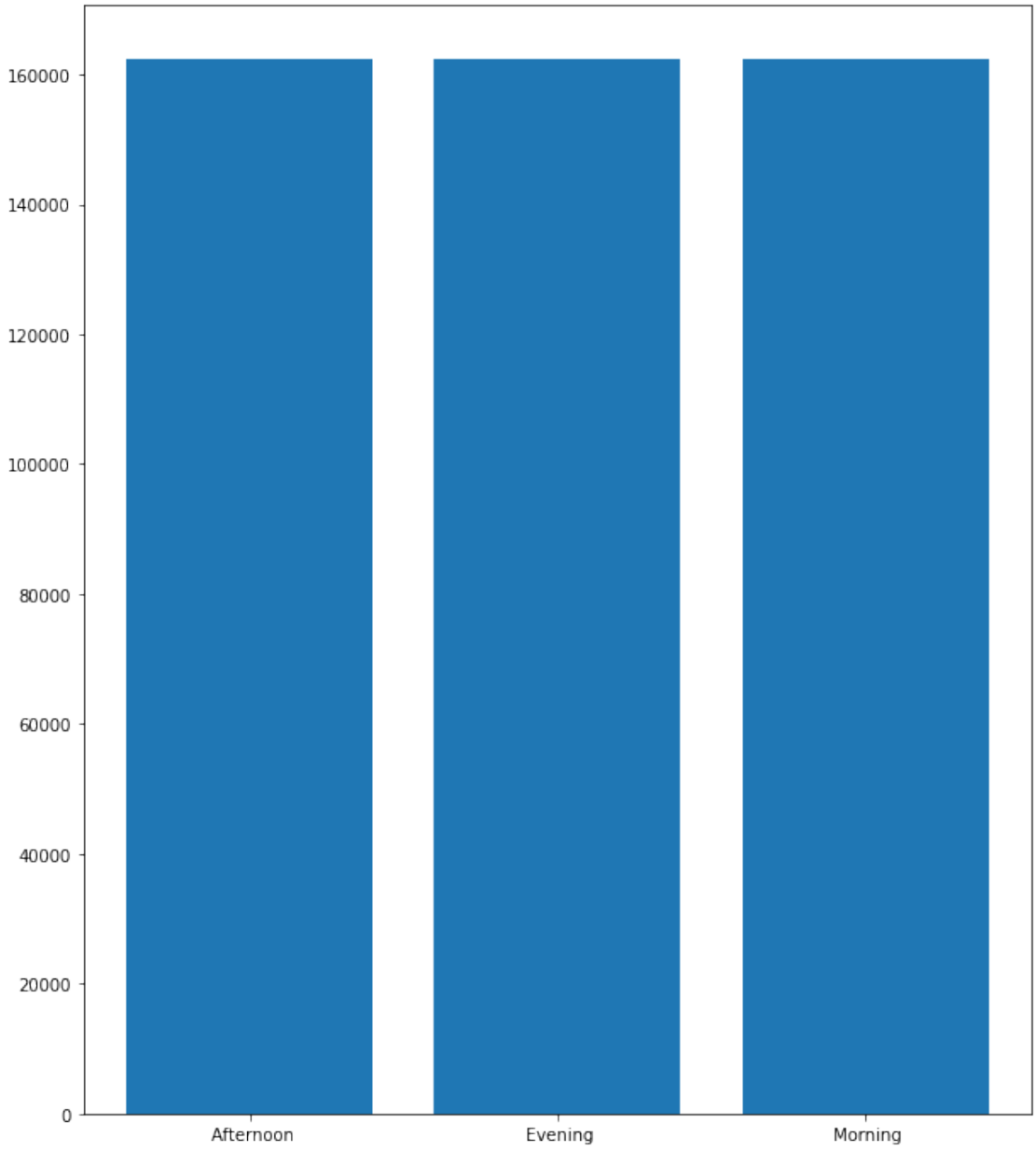
[55]:
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(12,12))
sns.boxplot(x='Group',y='Sales',hue='State',data=df)
plt.xlabel('Sales')
plt.ylabel('Group')
plt.show()
```



0.0.7 Time-of-the-day analysis: during which time of the day are sales the highest, and during which time are sales the lowest?

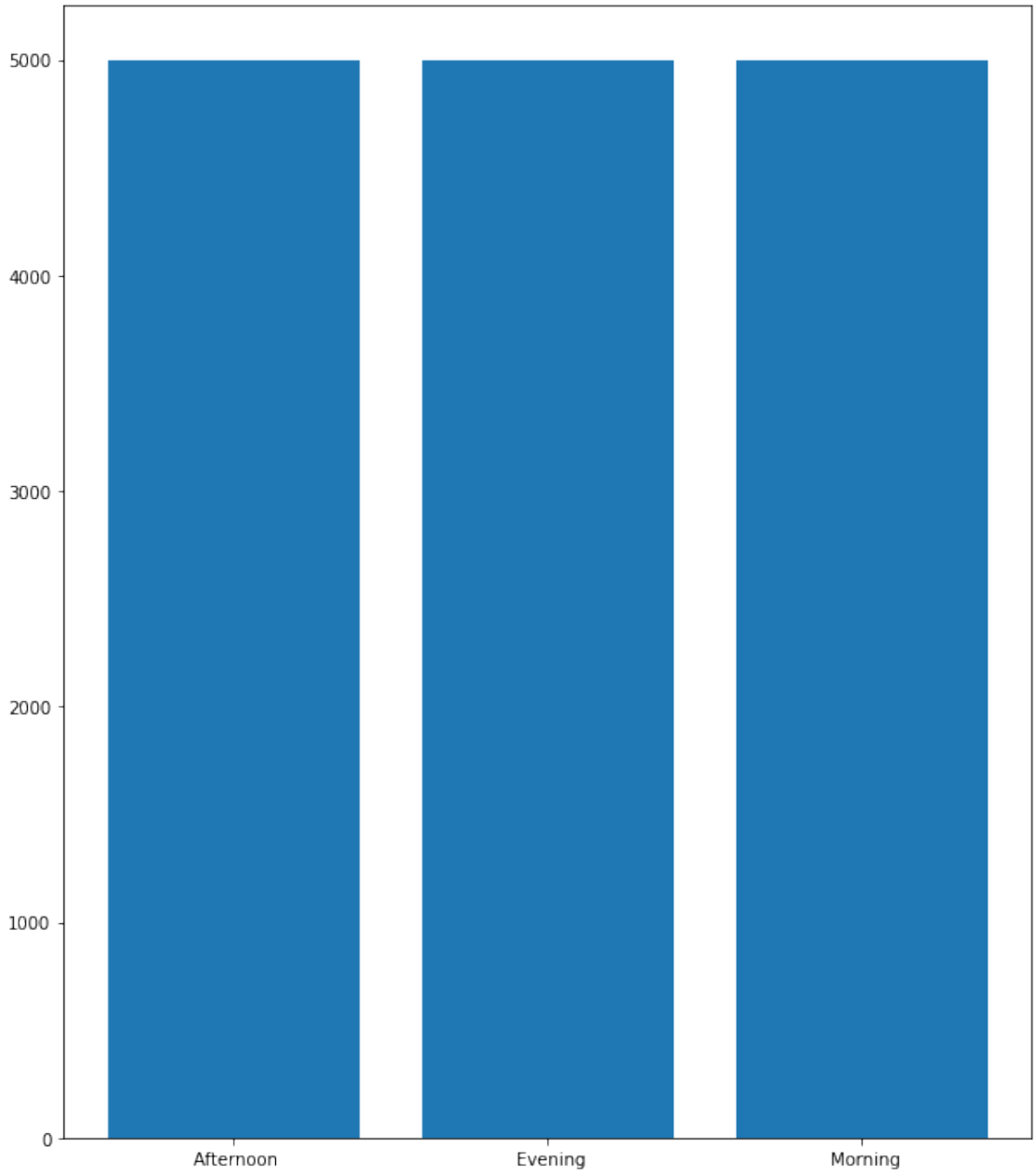
[71]:
```
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(10,12))
Time_Sales = df.groupby('Time')['Sales'].max()
plt.bar(Time_Sales.index, Time_Sales.values)
```

[71]: <BarContainer object of 3 artists>



[70]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,12))
Time_Sales = df.groupby('Time')['Sales'].min()
plt.bar(Time_Sales.index, Time_Sales.values)
```

[70]: <BarContainer object of 3 artists>



0.0.8 Include your recommendation, and indicate why you are choosing the recommended visualization package.

I had taken matplotlib and seaborn for visualization because that packages are easy to visualize the given content