**Christmas Sales Data Project**

**#1- import python libraries**

import numpy as np #for numeric Computing, array

import pandas as pd #for Data Cleaning

import matplotlib.pyplot as plt # visualizing data%matplotlib inline

import seaborn as sns #data visualization library based on Matplotlib.

**#pyplot** module provides a collection of functions that make it easy to create various types of plots, such as line plots, scatter plots, bar plots, histograms, and more.

**#2-import csv file**

**#encoding='unicode\_escape'** parameter is used to handle any potential encoding issues in the CSV file.

df = pd.read\_csv('Christmas Sales Data.csv', encoding= 'unicode\_escape')

**#3-df.shape** attribute in pandas is used to get the dimensions of a DataFrame. It returns a tuple representing the number of rows and columns in the DataFrame.

df.shape

**#4-You get 5 rows if you use head() function.**

df.head()

**#5-info():**DataFrame, including information about the data types, non-null values, and memory usage. It provides an overview of the structure of the DataFrame.

df.info()

**#6-drop** unrelated/blank columns

# inplace is Use for save changes in file.

df.drop(['Status', 'unnamed1'], axis=1, inplace=True)

**#7-df.isnull()** method in pandas is used to check for missing values (NaN) in a DataFrame.

df.isnull()

**#8-check for null values**

pd.isnull(df).sum()

**#9-drop null values**

**df.dropna(inplace=True)** method in pandas is used to remove any rows from the DataFrame (**df**) that contain at least one missing value (NaN).

df.dropna(inplace=True)

**#10-change data type**

**astype** method in pandas is used to change the data type of a column in a DataFrame.

df['Amount'] = df['Amount'].astype('int')

**#11-dtype is used for check Datatype in column.**

df['Amount'].dtypes

**#12-df.columns is used for check columns in table.**

df.columns

**#13-rename column.**

df.rename(columns= {'Marital\_Status':'Shaadi'})

**#14- describe() method returns description of the data in the DataFrame (i.e. count, mean, std, etc)**

df.describe()

**#15- use describe() for specific columns**

df[['Age', 'Orders', 'Amount']].describe()

**Exploratory Data Analysis**

**#1- plotting a bar chart for Gender and it's count**

ax = sns.countplot(x = 'Gender',data = df)

for bars in ax.containers:

ax.bar\_label(bars)

* **countplot** is used to create a bar plot for categorical data.
* **containers** are the individual bars in the plot.
* **bar\_label** is used to add labels to the bars, displaying the count of observations for each category.

**#2- plotting a bar chart for gender vs total amount.**

sales\_gen = df.groupby(['Gender'], as\_index=False)['Amount'].sum().sort\_values(by='Amount', ascending=False)

sns.barplot(x = 'Gender',y= 'Amount' ,data = sales\_gen)

**#3-Sales count Age group by and Gender.**

ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

for bars in ax.containers:

ax.bar\_label(bars)

**#4-Total Amount vs Age Group**

sales\_age = df.groupby(['Age Group'], as\_index=False)['Amount'].sum().sort\_values(by='Amount', ascending=False)

sns.barplot(x = 'Age Group',y= 'Amount' ,data = sales\_age)

**#5-total number of orders from top 10 states**

sales\_state = df.groupby(['State'], as\_index=False)['Orders'].sum().sort\_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(15,5)})

sns.barplot(data = sales\_state, x = 'State',y= 'Orders')

**#6- total amount/sales from top 10 states**

sales\_state = df.groupby(['State'], as\_index=False)['Amount'].sum().sort\_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(15,5)})

sns.barplot(data = sales\_state, x = 'State',y= 'Amount')

**#7-Marital Status**

ax = sns.countplot(data = df, x = 'Marital\_Status')

sns.set(rc={'figure.figsize':(7,5)})

for bars in ax.containers:

ax.bar\_label(bars)

**#8-Marital Status On Gender**

sales\_state = df.groupby(['Marital\_Status', 'Gender'], as\_index=False)['Amount'].sum().sort\_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(6,5)})

sns.barplot(data = sales\_state, x = 'Marital\_Status',y= 'Amount', hue='Gender')

**#9-occupation**

sns.set(rc={'figure.figsize':(20,5)})

ax = sns.countplot(data = df, x = 'Occupation')

for bars in ax.containers:

ax.bar\_label(bars)

**#10-Product Categories**

sns.set(rc={'figure.figsize':(20,5)})

ax = sns.countplot(data = df, x = 'Product\_Category')

for bars in ax.containers:

ax.bar\_label(bars)

**11-Product Categories**

sales\_state = df.groupby(['Product\_Category'], as\_index=False)['Amount'].sum().sort\_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})

sns.barplot(data = sales\_state, x = 'Product\_Category',y= 'Amount')

**12-Product Id**

sales\_state = df.groupby(['Product\_ID'], as\_index=False)['Orders'].sum().sort\_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})

sns.barplot(data = sales\_state, x = 'Product\_ID',y= 'Orders')

**#13- top 10 most sold products (same thing as above)**

fig1, ax1 = plt.subplots(figsize=(12,7))

df.groupby('Product\_ID')['Orders'].sum().nlargest(10).sort\_values(ascending=False).plot(kind='bar')