PHASE 3

Project Title: Product sales Analysis

To perform data preprocessing for product sales analysis on Kaggle, you can follow these steps:

- Data Import: Upload your dataset to Kaggle. You can either use datasets available on Kaggle or upload your own.
- Jupyter Notebook: Create a Jupyter Notebook on Kaggle, which is a popular platform for data analysis and machine learning. You can do this from the "Notebooks" section.
- Data Loading: In your Jupyter Notebook, load the dataset using Python libraries like Pandas. For example:
  - Python
  - import pandas as pd
  - data = pd.read_csv('your_dataset.csv')
- Data Exploration: Explore the dataset to understand its structure, check for missing values, and gain insights into the data. Use functions like `info()`, `head()`, and `describe()`.
- Data Cleaning:
  - Handle missing values using techniques such as imputation or removal.
  - Remove duplicates using the `drop duplicates()` function.
  - Correct any inconsistencies in the data.
- Data Transformation:
  - Convert data types if needed.
  - Normalize or scale numerical data.
  - Encode categorical variables.
- Data Visualization: Use libraries like Matplotlib and Seaborn to create visualizations to better understand the data.
- Data Analysis: Perform the desired sales analysis using statistical and machine learning techniques. This could involve trend analysis, forecasting, clustering, or classification, depending on your objectives.
- Data Export: Save the preprocessed data and analysis results if needed.

PREPROCESSING DATASET FOR SALES AND ANALYSIS :

```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib.style as style
from datetime import timedelta
import datetime as dt
import time
import os


for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/retail-case-study-data/prod_cat_info.csv
```

```
/kaggle/input/retail-case-study-data/Transactions.csv
/kaggle/input/retail-case-study-data/Customer.csv

customer = pd.read_csv("/kaggle/input/retail-case-study-data/Customer.csv")
prod_cat= pd.read_csv("/kaggle/input/retail-case-study-data/prod_cat_info.csv")
transactions = pd.read_csv("/kaggle/input/retail-case-study-data/Transactions.csv")
```

DATA PREPROCESSING:

```
customer.isnull().sum()
"""Both Gender and city_code columns have null values"""

#To fix this, I applied ffill (fill forward) to the null cells
customers = customer.fillna({
    'Gender': customer['Gender'].ffill(),
    'city_code': customer['city_code'].ffill()
})

#Splitting transaction date into year, month and day of week


transactions['tran_date'] = pd.to_datetime(transactions['tran_date'], errors='coerce')


linkcode
transactions.insert(loc=3, column='year', value= transactions.tran_date.dt.year)
transactions.insert(loc=4, column='month', value= transactions.tran_date.dt.month)
transactions.insert(loc=5, column='day', value=(transactions.tran_date.dt.weekday_name))

transactions.head()
```

OUTPUT

	transaction_id	cust_id	tran_date	year	month	day	prod_subcat_code	prod_cat_code	Qty	Rate	Tax	total_amt	Store_type
0	80712190438	270351	2014-02-28	2014	2	Friday	1	1	-5	-772	405.300	-4265.300	e-Shop
1	29258453508	270384	2014-02-27	2014	2	Thursday	5	3	-5	-1497	785.925	-8270.925	e-Shop
2	51750724947	273420	2014-02-24	2014	2	Monday	6	5	-2	-791	166.110	-1748.110	TeleShop
3	93274880719	271509	2014-02-24	2014	2	Monday	11	6	-3	-1363	429.345	-4518.345	e-Shop
4	51750724947	273420	2014-02-23	2014	2	Sunday	6	5	-2	-791	166.110	-1748.110	TeleShop

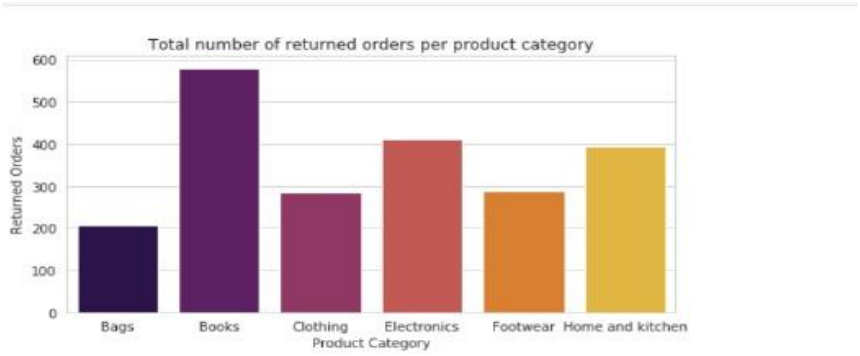
```
df = pd.merge(left = customers, right = transactions,
left_on = 'customer_Id', right_on = 'cust_id').drop('cust_id', axis =1)
```

```
#This joins the customers and transactions dataset on customer_Id and cus
t_id. The duplicate column (cust_id)
#is dropped.
df.duplicated().sum()

#There are 13 duplicate cells in the df dataframe. Next step: drop duplic
ates.
df.drop_duplicates(inplace = True)
df_new = pd.merge(df, prod_cat, left_on = ('prod_subcat_code', "prod_ca
t_code"), right_on = ('prod_sub_cat_code', "prod_cat_code")).drop('prod
_sub_cat_code', axis =1)
df_new.shape
#Columns from the prod_cat dataset have been added to the df dataframe
df_new.describe()

#showing basic statistical details
customer_city=df_new[['city_code','customer_Id']]
customer_city.groupby(['city_code'])['customer_Id'].aggregate('count').
reset_index().sort_values('customer_Id', ascending=False)
```

	city_code	customer_Id
3	4.0	2430
2	3.0	2410
4	5.0	2357
6	7.0	2356
9	10.0	2333
7	8.0	2328
1	2.0	2268
0	1.0	2255
8	9.0	2176
5	6.0	2127

RETAIL SALES AND ANALYSIS:-

```
"""Books, Electronics and Home & Kitchen were the most returned product c
ategories."""
category = rdf.groupby(by=['prod_cat'], as_index = False)['Qty'].count(
)
plt.figure(figsize=(8,4))
sns.set_style('whitegrid')
sns.barplot(x = "prod_cat", y = 'Qty', data = category,  palette= "infe
rno")
plt.xlabel('Product Category')
plt.ylabel('Returned Orders')
plt.title('Total number of returned orders per product category')
plt.show()
```



PURCHASE BY AGE CATEGORY :-

```
"""Customers aged betweeen 40-50 purchased the most products and 24-30 custo
mers purchased the least"""
plt.figure(figsize=(8,6))
sns.countplot(x = 'prod_cat', hue = 'age_category', data = sdf, palette= "i
nferno")

"""Pivot chart representation"""
spend_per_category = sdf.groupby(['age_category','prod_cat'])['total_amt'].
sum().reset_index()
spend_per_category.pivot(index = "age_category", columns = "prod_cat", valu
es = 'total_amt').round(0)
```

prod_cat	Bags	Books	Clothing	Electronics	Footwear	Home and kitchen
age_category						
24-29	841516.0	2530348.0	1357383.0	2071557.0	1271120.0	1747640.0
30-39	1814498.0	5620636.0	2832031.0	4676507.0	2729131.0	3699736.0
40-50	2027516.0	6233818.0	2885083.0	5095412.0	2974368.0	4045584.0