Text

Description automatically generated

**Activity based**

**Project Report on**

**DWDM**

**Submitted to Vishwakarma University, Pune**

**Under the Initiative of**

**By**

**Arun Adhikari**

**SRN No : 202100406**

**Roll No : 38**

**Div : G**

**Third Year Engineering**

**Department of Computer Engineering**

**Faculty of Science and Technology**

**Academic Year**

**2023-2024**

**Problem Statement**

**Implement the Apriori Algorithm.**

**Business Requirements**

**Stakeholders: Executive leadership, customers,**

**Data Sources: Customer database, content database**

import numpy as np

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules, fpgrowth

# Loading the Data

data = pd.read\_excel('Online\_Retail.xlsx')

data.head()

# Exploring the columns of the data

data.columns

# Exploring the different regions of transactions

data.Country.unique()

# Stripping extra spaces in the description

data['Description'] = data['Description'].str.strip()

# Dropping the rows without any invoice number

data.dropna(axis=0, subset=['InvoiceNo'], inplace=True)

data['InvoiceNo'] = data['InvoiceNo'].astype('str')

# Dropping all transactions which were done on credit

data = data[~data['InvoiceNo'].str.contains('C')]

# Transactions done in France

basket\_France = (data[data['Country'] == "France"]

.groupby(['InvoiceNo', 'Description'])['Quantity']

.sum().unstack().reset\_index().fillna(0)

.set\_index('InvoiceNo'))

# Transactions done in the United Kingdom

basket\_UK = (data[data['Country'] == "United Kingdom"]

.groupby(['InvoiceNo', 'Description'])['Quantity']

.sum().unstack().reset\_index().fillna(0)

.set\_index('InvoiceNo'))

# Transactions done in Portugal

basket\_Por = (data[data['Country'] == "Portugal"]

.groupby(['InvoiceNo', 'Description'])['Quantity']

.sum().unstack().reset\_index().fillna(0)

.set\_index('InvoiceNo'))

# Transactions done in Sweden

basket\_Sweden = (data[data['Country'] == "Sweden"]

.groupby(['InvoiceNo', 'Description'])['Quantity']

.sum().unstack().reset\_index().fillna(0)

.set\_index('InvoiceNo'))

# Defining the hot encoding function to make the data suitable for the concerned libraries

def hot\_encode(x):

if x <= 0:

return 0

if x >= 1:

return 1

# Encoding the datasets with map (since applymap is deprecated)

basket\_France = basket\_France.map(hot\_encode)

basket\_UK = basket\_UK.map(hot\_encode)

basket\_Por = basket\_Por.map(hot\_encode)

basket\_Sweden = basket\_Sweden.map(hot\_encode)

# --------------------

# Building models with FP-Growth instead of Apriori

# --------------------

# France

frq\_items = fpgrowth(basket\_France, min\_support=0.05, use\_colnames=True)

rules = association\_rules(frq\_items, metric="lift", min\_threshold=1)

rules = rules.sort\_values(['confidence', 'lift'], ascending=[False, False])

print("France Rules")

print(rules.head())

# UK (using higher min\_support to avoid memory issues)

frq\_items = fpgrowth(basket\_UK, min\_support=0.02, use\_colnames=True)

rules = association\_rules(frq\_items, metric="lift", min\_threshold=1)

rules = rules.sort\_values(['confidence', 'lift'], ascending=[False, False])

print("UK Rules")

print(rules.head())

# Portugal

frq\_items = fpgrowth(basket\_Por, min\_support=0.05, use\_colnames=True)

rules = association\_rules(frq\_items, metric="lift", min\_threshold=1)

rules = rules.sort\_values(['confidence', 'lift'], ascending=[False, False])

print("Portugal Rules")

print(rules.head())

# Sweden

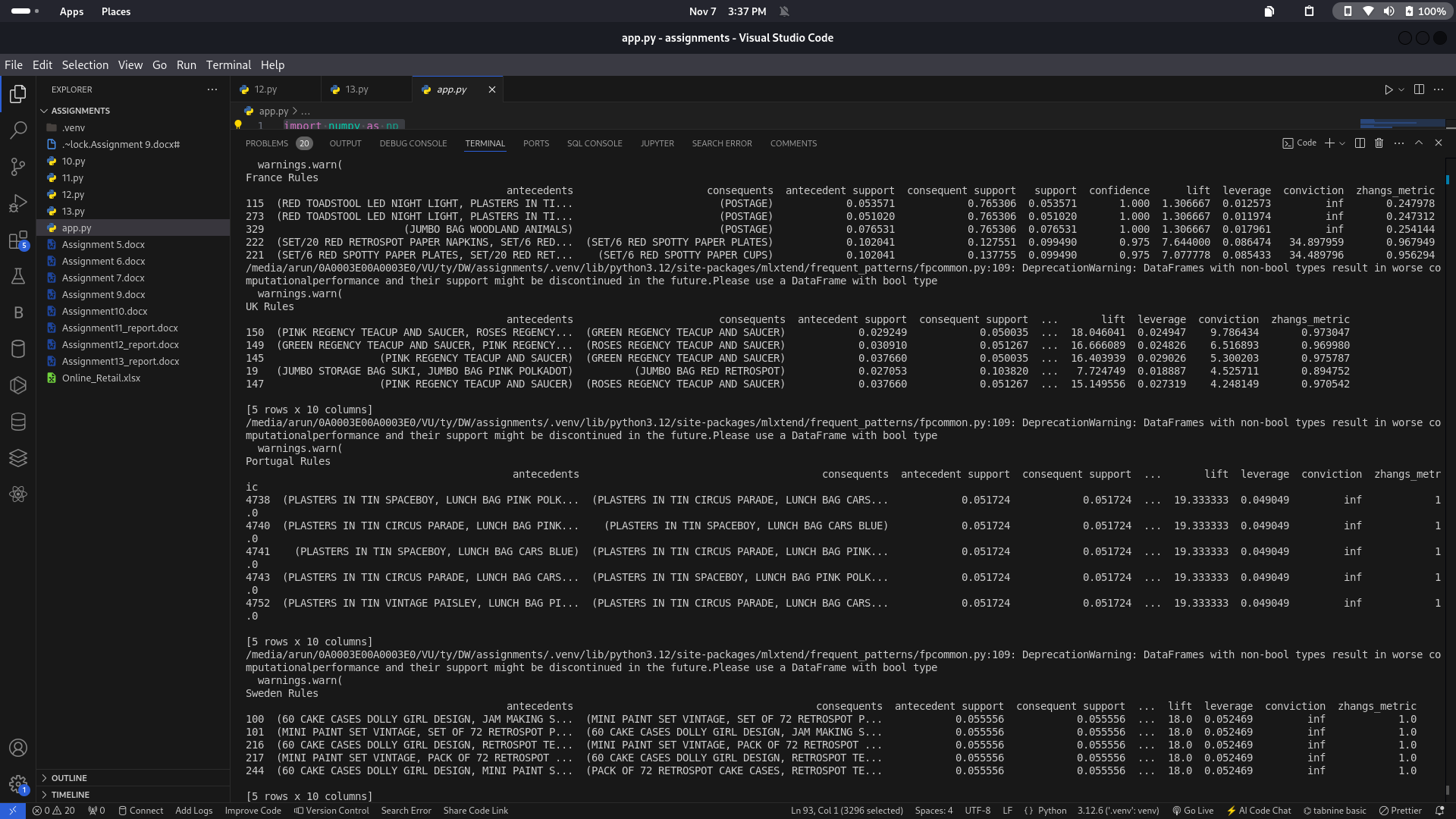
frq\_items = fpgrowth(basket\_Sweden, min\_support=0.05, use\_colnames=True)

rules = association\_rules(frq\_items, metric="lift", min\_threshold=1)

rules = rules.sort\_values(['confidence', 'lift'], ascending=[False, False])

print("Sweden Rules")

print(rules.head())

****