R, F, M Analysis - Week 2 Assignment

⚠️ **Online Retail II data analysis on UCI Machine Learning Repository**

20229007 Ryu, Sungyoung (BTM, K-School MEI's Program)

```
import pandas as pd
from google.colab import drive #Mounting Google Drive
drive.mount('/content/drive') #Setting a file path to data directory
%cd '/content/drive/MyDrive/Colab Notebooks/data/'
df0910 = pd.read_excel('online_retail_II.xlsx', sheet_name = 0) #yeardata sheets from 2009 to 2010
df1011 = pd.read_excel('online_retail_II.xlsx', sheet_name = 1) #yeardata sheets from 2010 to 2011
```

```
df0911 = pd.concat([df0910, df1011])
df0911.to_csv('btm539_rfm_analysis.csv', index=False)  # index=False prevents pandas to write row index
```

```
import pandas as pd
from google.colab import drive #Mounting Google Drive
drive.mount('/content/drive') #Setting a file path to data directory
%cd '/content/drive/MyDrive/Colab Notebooks/data/'
df = pd.read_csv('btm539_rfm_analysis.csv')
df
```

```
df.dtypes
```

```
df.isna().sum()
nan_values = df[df.isnull().any(axis=1)]
print (nan_values)
```

```
df = df.dropna()
df
```

```
df.astype({'Customer ID': 'int32'}).dtypes
```

```
# Spaghetti code and hashed comment is making loops with 'for' iteration

df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
df['Year'] = df['InvoiceDate'].dt.year.astype('int32')
df['Month'] = df['InvoiceDate'].dt.month_name()
df['Day'] = df['InvoiceDate'].dt.day.astype('int32')
df['Hour'] = df['InvoiceDate'].dt.hour.astype('int32')
df['Minute'] = df['InvoiceDate'].dt.minute.astype('int32')
df['Second'] = df['InvoiceDate'].dt.second.astype('int32')
df['Weekday'] = df['InvoiceDate'].dt.strftime('%A')
ymdhmsw = ['Year', 'Month', 'Day', 'Hour', 'Minute', 'Second', 'Weekday']
# for col in ymdhmsw:
#     df[col] = df['InvoiceDate'].dt.__getattribute__(col.lower())
#     df.insert(5, col, df.pop(col))
df.insert(loc=5, column='Weekday', value=df.pop('Weekday'))
df.insert(loc=5, column='Second', value=df.pop('Second'))
df.insert(loc=5, column='Minute', value=df.pop('Minute'))
df.insert(loc=5, column='Hour', value=df.pop('Hour'))
df.insert(loc=5, column='Day', value=df.pop('Day'))
df.insert(loc=5, column='Month', value=df.pop('Month'))
df.insert(loc=5, column='Year', value=df.pop('Year'))
df
```

```
# Regarding calculating recency (advise from prof. Kwon):

import datetime as dt
from datetime import datetime, timedelta
customer_retention = df.groupby('Customer ID')['InvoiceDate'].max().reset_index() # - The last transaction data will have 1 for recency.
customer_retention['CustomerRetention'] = customer_retention['InvoiceDate'] + dt.timedelta(days=1) # - last_day = the last transaction day + use 'timedelta' function to
customer_retention['Recency'] = (customer_retention['CustomerRetention'] - df['InvoiceDate']).dt.days # - Then, recency = (last_day - InvoiceDate).dt.days, which will g
print(customer_retention)
```

```
snapshot_date = df['InvoiceDate'].max() + dt.timedelta(days=1) # creating timestamp from invoice date
df['TotalSum'] = df['Price'] * df['Quantity']
rfm = df.groupby(['Customer ID']).agg({'InvoiceDate': lambda x : (snapshot_date - x.max()).days, 'Invoice':'count', 'TotalSum': 'sum'})
rfm = rfm.rename(columns={'InvoiceDate':'Recency','InvoiceNo':'Frequency','TotalSum':'MonetaryValue'})
rfm
```

Customer ID	Recency	Invoice	MonetaryValue
12346.0	326	48	-64.68
12347.0	2	253	5633.32
12348.0	75	51	2019.40
12349.0	19	180	4404.54
12350.0	310	17	334.40
...	...	...	...
18283.0	4	986	2736.65
18284.0	430	29	436.68
18285.0	661	12	427.00
18286.0	477	70	1188.43
18287.0	43	156	4177.89

5942 rows × 3 columns

```
df = df[(df['Quantity'] > 0) & (df['Price'] = 0)] # Negative 'Q' (Return goods) and Zero 'P' (Free gift) has been cleared.
df
```

```
print("Number of transactions: ", df['Invoice'].nunique())
print("Number of products: ",df['StockCode'].nunique())
print("Number of customers:", df['Customer ID'].nunique())
groupby_invoice = pd.DataFrame(df.groupby('Invoice')['StockCode'].nunique()) # Build the composition of baskets, group by 'Invoice'
groupby_invoice['CustomersNo'] = df.groupby('Invoice')['Customer ID'].nunique()
groupby_invoice
```

```
import seaborn as sns
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
fig.set_size_inches(12, 10)
sns.distplot(groupby_invoice,ax=ax)
plt.show()
```