Ezinne Okechukwu | Data Analyst | ezdavoo@gmail.com

Quantum Job Simulation || Forage Virtual Internship

```
# import numpy as np
import pandas as pd
# import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
```

```
# load files
df_cust_comm = pd.read_csv('cust')
df_calendar = pd.read_csv('calendar')
df_transactions = pd.read_csv('transactions')
df_newcustomerlist = pd.read_csv('newcustomerlist')
df_customerdemographic = pd.read_csv('customerdemographic')
```

```
# head files
df_cust_comm.head()
df_calendar.head()
df_transactions.head()
df_newcustomerlist.head()
df_customerdemographic.head()
```

```
df_cust_comm.shape
```

```
((127,5,14400,800) DATE: STORE_NBR: TRN_ID: PROD_NBR: PROD_NAME: PROD_QTY: TOT_SALES: PACK_SIZE: BRAND: LIFESTAGE: PREMIUM_C
0  2018/10/17  1  1000  1  Natural Chip Compny SeaSalt175g  2  6.0  175  NATURAL  YOUNG SINGLES/COUPLES  Mainstream
1  2018/10/16  8  1307  348  Red Rock Deli Chikn&Garlic Aioli 150g  2  2.7  150  RRD  YOUNG SINGLES/COUPLES  Mainstream
2  2018/10/01  2  1343  383  Grain Waves Sour Cream&Chives 210G  2  3.6  210  GRAINWAVES  YOUNG FAMILIES  Budget
3  2018/10/07  3  5  32  Cobram Estate EVOO 250ml  1  11.0  250  COBRAM  YOUNG SINGLES/COUPLES  Premium
4  2018/10/08  4  1004  1  Natural Chip Compny SeaSalt175g  2  6.0  175  NATURAL  YOUNG SINGLES/COUPLES  Mainstream
```

```
df_calendar.shape
```

```
(364,2)
```

```
df_transactions.shape
```

```
(264836,8)
```

```
df_newcustomerlist.shape
```

```
(2000,10)
```

```
df_customerdemographic.shape
```

```
(1000,5)
```

Data Inspection and Preparation

```
# check columns
df.columns
```

```
Index(['LYLTY_CARD_NBR', 'DATE', 'STORE_NBR', 'TXN_ID', 'PROD_NBR',
       'PROD_NAME', 'PROD_QTY', 'TOT_SALES', 'PACK_SIZE', 'BRAND',
       'LIFESTAGE', 'PREMIUM_CUSTOMER'],
      dtype='object')
```

```
# check columns in newcustomerlist
df_newcustomerlist.columns, df_customerdemographic.columns
```

```
(Index(['LYLTY_CARD_NBR', 'TXN_DATE', 'PROD_NBR', 'PROD_QTY', 'TOT_SALES',
       'PROMOCODE', 'PREMIUM_CUSTOMER', 'Unnamed: 7', 'Unnamed: 8',
       'Unnamed: 9'],
      dtype='object'),
 Index(['LYLTY_CARD_NBR', 'LIFEMAP', 'FRAMEWORK'],
      dtype='object'))
```

```
# check data types
df.dtypes
```

```
LYLTY_CARD_NBR    int64
DATE             object
STORE_NBR         int64
TXN_ID            int64
PROD_NBR          int64
PROD_NAME        object
PROD_QTY          int64
TOT_SALES       float64
PACK_SIZE         int64
BRAND            object
LIFESTAGE        object
PREMIUM_CUSTOMER ...
```

```
# check for null values
df.isnull().sum()
```

```
LYLTY_CARD_NBR    0
DATE              0
STORE_NBR         0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
PACK_SIZE         0
BRAND             0
LIFESTAGE         0
PREMIUM_CUSTOMER  0
dtype: int64
```

```
df.isnull().sum()
```

```
LYLTY_CARD_NBR    0
DATE              0
STORE_NBR         0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
PACK_SIZE         0
BRAND             0
LIFESTAGE         0
PREMIUM_CUSTOMER  0
dtype: int64
```

```
df_newcustomerlist.isnull().sum()
```

```
LYLTY_CARD_NBR    0
TXN_DATE          0
PROD_NBR          0
PROD_QTY          0
TOT_SALES         0
PROMOCODE         0
PREMIUM_CUSTOMER  0
Unnamed: 7        0
Unnamed: 8        0
Unnamed: 9        0
dtype: int64
```

```
df_customerdemographic.isnull().sum()
```

```
LYLTY_CARD_NBR    0
LIFEMAP           0
FRAMEWORK         0
dtype: int64
```

```
# check duplicate
df.duplicated().sum()
```

```
0
```

```
df_newcustomerlist.duplicated().sum()
```

```
0
```

```
df_customerdemographic.duplicated().sum()
```

```
0
```

```
# change date to datetime
df['DATE'] = pd.to_datetime(df['DATE'])
df['DATE'].dtypes
```

```
dtype('datetime64[ns]')
```

```
# change date to datetime
df['TXN_DATE'] = pd.to_datetime(df['TXN_DATE'])
df['TXN_DATE'].dtypes
```

```
dtype('datetime64[ns]')
```

```
# change date to month
df['MONTH'] = df['DATE'].dt.month
df['MONTH'].head()
```

```
0    10
1    10
2    10
3    10
4    10
Name: MONTH, dtype: int64
```

```
# change date to month
df['TXN_MONTH'] = df['TXN_DATE'].dt.month
df['TXN_MONTH'].head()
```

```
0    10
1    10
2    10
3    10
4    10
Name: TXN_MONTH, dtype: int64
```

```
# change date to year
df['YEAR'] = df['DATE'].dt.year
df['YEAR'].head()
```

```
0    2018
1    2018
2    2018
3    2018
4    2018
Name: YEAR, dtype: int64
```

```
# change date to year
df['TXN_YEAR'] = df['TXN_DATE'].dt.year
df['TXN_YEAR'].head()
```

```
0    2018
1    2018
2    2018
3    2018
4    2018
Name: TXN_YEAR, dtype: int64
```

```
# create a new column
df['PROMOCODE'] = np.nan
df['PROMOCODE'].head(5)
```

```
0    NaN
1    NaN
2    NaN
3    NaN
4    NaN
Name: PROMOCODE, dtype: float64
```

```
# create a new column
df['PREMIUM_CUSTOMER'] = np.nan
df['PREMIUM_CUSTOMER'].head(5)
```

```
0    NaN
1    NaN
2    NaN
3    NaN
4    NaN
Name: PREMIUM_CUSTOMER, dtype: float64
```

```
df_cust_comm.head(5)
df_calendar.head(5)
df_transactions.head(5)
df_newcustomerlist.head(5)
df_customerdemographic.head(5)
```

```
df_cust_comm.shape
df_calendar.shape
df_transactions.shape
df_newcustomerlist.shape
df_customerdemographic.shape
```

```
(127,5,14400,800)
(364,2)
(264836,8)
(2000,10)
(1000,5)
```

Before Trial: Finding Control Stores for each Trial Store

Trial began in February 2019 and ended 30 April, 2019.
To get control stores for each trial stores, get the store with highest premium combination monthly sales with each trial store.

```
# create a new sales dataset for transactions before trial period
df_sales = df[df['DATE'] < '2019-02']
df_sales.head(5)
df_sales.shape
# create a new sales dataset for transactions after trial period
# filter out transactions after trial period
df_sales = df[df['DATE'] >= '2019-02']
df_sales.head(5)
df_sales.shape
```

```
(0,0)
```

```
# merge sales dataset
df_sales = df[['LYLTY_CARD_NBR', 'DATE', 'STORE_NBR', 'TXN_ID', 'PROD_NBR',
               'PROD_NAME', 'PROD_QTY', 'TOT_SALES', 'PACK_SIZE', 'BRAND',
               'LIFESTAGE', 'PREMIUM_CUSTOMER']]
df_sales.head(5)
```

```
# create a new column
df_sales['MONTH'] = df_sales['DATE'].dt.month
df_sales['MONTH'].head(5)
```

```
Series([], Name: MONTH, dtype: int64)
```

```
# find the total sales of store by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['TOT_SALES'].sum().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# find the total number of customers by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['LYLTY_CARD_NBR'].nunique().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	LYLTY_CARD_NBR
1	2018/07	20
1	2018/08	19
1	2018/09	44
1	2018/10	45
1	2019/02	45

```
# create a new dataset
monthly_sales = pd.merge(df_total_sales[['STORE_NBR', 'MONTH']], df_total_sales, on=['STORE_NBR', 'MONTH'])
monthly_sales.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# create a new dataset for transactions before trial period
df_sales = df[df['DATE'] < '2019-02']
df_sales.head(5)
df_sales.shape
# create a new dataset for transactions after trial period
# filter out transactions after trial period
df_sales = df[df['DATE'] >= '2019-02']
df_sales.head(5)
df_sales.shape
```

```
(0,0)
```

```
# create a new column
df_sales['MONTH'] = df_sales['DATE'].dt.month
df_sales['MONTH'].head(5)
```

```
Series([], Name: MONTH, dtype: int64)
```

```
# find the total sales of store by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['TOT_SALES'].sum().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# find the total number of customers by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['LYLTY_CARD_NBR'].nunique().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	LYLTY_CARD_NBR
1	2018/07	20
1	2018/08	19
1	2018/09	44
1	2018/10	45
1	2019/02	45

```
# create a new dataset
monthly_sales = pd.merge(df_total_sales[['STORE_NBR', 'MONTH']], df_total_sales, on=['STORE_NBR', 'MONTH'])
monthly_sales.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# create a new dataset
df_sales = df[df['DATE'] < '2019-02']
df_sales.head(5)
df_sales.shape
# create a new dataset for transactions after trial period
# filter out transactions after trial period
df_sales = df[df['DATE'] >= '2019-02']
df_sales.head(5)
df_sales.shape
```

```
(0,0)
```

```
# create a new column
df_sales['MONTH'] = df_sales['DATE'].dt.month
df_sales['MONTH'].head(5)
```

```
Series([], Name: MONTH, dtype: int64)
```

```
# find the total sales of store by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['TOT_SALES'].sum().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# find the total number of customers by month
monthly_sales_by_store = df_sales.groupby(['STORE_NBR', 'MONTH'])['LYLTY_CARD_NBR'].nunique().reset_index()
monthly_sales_by_store.head(5)
```

STORE_NBR	MONTH	LYLTY_CARD_NBR
1	2018/07	20
1	2018/08	19
1	2018/09	44
1	2018/10	45
1	2019/02	45

```
# create a new dataset
monthly_sales = pd.merge(df_total_sales[['STORE_NBR', 'MONTH']], df_total_sales, on=['STORE_NBR', 'MONTH'])
monthly_sales.head(5)
```

STORE_NBR	MONTH	TOT_SALES
1	2018/07	206.9
1	2018/08	150.8
1	2018/09	299.8
1	2018/10	308.8
1	2019/02	268.8

```
# create a function to calculate correlation
def find_highest_corr(df, trial_store):
    # create an empty dataset
    corr_df = pd.DataFrame(columns=['STORE_NBR', 'MONTH', 'TOT_SALES', 'TOT_CUSTOMERS'])
    # loop through each store
    for store in df['STORE_NBR'].unique():
        # find the total sales and total customers for this store
        store_sales = df[df['STORE_NBR'] == store]['TOT_SALES'].sum()
        store_customers = df[df['STORE_NBR'] == store]['TOT_CUSTOMERS'].sum()
        # find the total sales and total customers for the trial store
        trial_sales = df[df['STORE_NBR'] == trial_store]['TOT_SALES'].sum()
        trial_customers = df[df['STORE_NBR'] == trial_store]['TOT_CUSTOMERS'].sum()
        # calculate the correlation
        corr = store_sales.corr(trial_sales)
        # add the correlation to the dataset
        corr_df.loc[len(corr_df)] = [store, corr, trial_sales, trial_customers]
    # return the dataset
    return corr_df
```

```
# find the highest correlation for trial store 77
corr_df = find_highest_corr(monthly_sales, 77)
corr_df.head(5)
```

STORE_NBR	MONTH	TOT_SALES	TOT_CUSTOMERS
1	2018/07	206.9	20
1	2018/08	150.8	19
1	2018/09	299.8	44
1	2018/10	308.8	45
1	2019/02	268.8	45

```
# find the highest correlation for trial store 77
corr_df.sort_values(by='TOT_SALES', ascending=False).head(5)
```

STORE_NBR	MONTH	TOT_SALES	TOT_CUSTOMERS
1	2018/07	206.9	20
1	2018/08	150.8	19
1	2018/09	299.8	44
1	2018/10	308.8	45
1	2019/02	268.8	45

```
# find the highest correlation for trial store 77
corr_df.sort_values(by='TOT_CUSTOMERS', ascending=False).
```