|  |  |
| --- | --- |
| **Name** | Gisanrin Oluwaseun |
| **Email** | [seungisanrin@gmail.com](mailto:seungisanrin@gmail.com) |
| **GitHub** | github.com/seungisanrin |

# \*\*Quantium Data Analytics Internship - Task Two (2)\*\*

## Loading Libraries and Datasets for the Analysis

# Loading the necessary libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from statistics import stdev

# Loading the cleaned and merged dataaset

df = pd.read\_csv('QVI\_data.csv')

df.shape

df.dtypes

df.describe()

df.head()

# Converting the date column to a datetime object

df['DATE'] = pd.to\_datetime(df['DATE'])

## Extracting Stores with Transactions in every Month

len(df['STORE\_NBR'].unique())

df['MONTH'] = df['DATE'].dt.strftime('%B')

df['YEAR\_MONTH'] = df['DATE'].dt.strftime('%Y-%m')

# Number of months with transactions per store

n\_months\_perstore = df.groupby('STORE\_NBR')['YEAR\_MONTH'].nunique().reset\_index()

# Filtering for stores that have transactions recorded for every month

full\_obs = n\_months\_perstore[n\_months\_perstore['YEAR\_MONTH'] == 12]

full\_obs

The stores who don't have transactions for at least one month have been filtered, now to reflect on the dataset.

# Storing the eligible stores for the analysis to a variable

store\_df = df

# Extracting the relevant informmation from the

monthly\_revenue = store\_df.groupby(['STORE\_NBR','YEAR\_MONTH'])['TOT\_SALES'].sum().reset\_index()

monthly\_customers = store\_df.groupby(['STORE\_NBR','YEAR\_MONTH'])['LYLTY\_CARD\_NBR'].nunique().reset\_index()

monthly\_txn = store\_df.groupby(['STORE\_NBR','YEAR\_MONTH'])['TXN\_ID'].count().reset\_index()

monthly\_nchips = store\_df.groupby(['STORE\_NBR','YEAR\_MONTH'])['PROD\_QTY'].sum().reset\_index()

# Aggregating all the necessary metrics for further analysis

store\_agg = pd.concat([monthly\_revenue, monthly\_customers['LYLTY\_CARD\_NBR'],

monthly\_txn['TXN\_ID'], monthly\_nchips['PROD\_QTY']],

axis=1,

ignore\_index=True)

store\_agg.columns = ['store\_nbr','year\_month','tot\_sales','n\_customers',

'n\_txns','n\_chips']

# Computing more of the necessary metrics

store\_agg['txn\_per\_customer'] = store\_agg['n\_txns']/store\_agg['n\_customers']

store\_agg['chips\_per\_customer'] = store\_agg['n\_chips']/store\_agg['n\_customers']

store\_agg['avgprice\_per\_unit'] = store\_agg['tot\_sales']/store\_agg['n\_chips']

# Dropping rows that are not necessary for the analysis

store\_agg.drop(columns=['n\_txns','n\_chips'], inplace=True)

store\_agg.head()

store\_agg.info()

# Filtering the store aggregate data for pretrial metrics, before February 2019

pretrial\_metric = store\_agg[(store\_agg['year\_month'] < '2019-02') &

(store\_agg['store\_nbr'].isin(full\_obs['STORE\_NBR']))]

pretrial\_metric.shape

# Function to calculate the correlation between a trial store and control stores

def calcCorrelation(metric\_col,trial\_store,df\_x):

'''

Calculates the correlation coefficient between all the control stores and the

specified trial store.

Args:

metric\_col (str): The column metric on which the correlation coefficient should be

calculated

trial\_store (int): The trial store number that is being compared with other control stores.

Options are: 'tot\_sales', 'n\_customers', 'txn\_per\_customer',

'chips\_per\_customer', 'avgprice\_per\_unit'.

df (pd.DataFrame): The dataframe containing all metrics for both trial and control stores

Returns:

Pandas DataFrama Object: Dataframe with three columns the trial store number,

control store number, and the magnitude distance.

'''

# Creating an output dictionary to store the values generated

output = {'trial\_store': [],'control\_store':[],'corr\_coef':[]}

# Extracting the specified metric column for the trial stores

trial\_val = df\_x[df\_x['store\_nbr'] == trial\_store][metric\_col].reset\_index(drop=True)

# Creating a series for the store numbers to loop through

store\_num = df\_x['store\_nbr'].unique()

# Looping through all the stores to generate the coefficient metric

for i in store\_num:

control\_val = df\_x[df\_x['store\_nbr'] == i][metric\_col].reset\_index(drop=True)

output['trial\_store'].append(trial\_store)

output['control\_store'].append(i)

output['corr\_coef'].append(control\_val.corr(trial\_val))

# Saving the populated dictionary to a dataframe

output\_df = pd.DataFrame(output)

return output\_df

\*\*Calculating Magnitude Distance\*\*

1 - (Observed distance – minimum distance)/(Maximum distance – minimum distance)

# Function to calculate the magnitude distance between a trial store and control stores

def calcMagnitudeDistance(metric\_col,trial\_store,df\_x):

'''

Calculates the magnitude distance between all the control stores and the

specified trial store

Args:

metric\_col (str): The column metric on which the magnitude distance should be calculated

trial\_store (int): The trial store number that is being compared with other control stores.

Options are: 'tot\_sales', 'n\_customers', 'txn\_per\_customer',

'chips\_per\_customer', 'avgprice\_per\_unit'.

df (pd.DataFrame): The dataframe containing all metrics for both trial and control stores

Returns:

Pandas DataFrama Object: Dataframe with three columns the trial store number,

control store number, and the magnitude distance.

'''

# Creating an output dictionary to store the values generated

output = {'trial\_store': [],'control\_store':[],'mag\_distance':[]}

# Extracting the specified metric column for the trial stores

trial\_val = df\_x[df\_x['store\_nbr'] == trial\_store][metric\_col].reset\_index(drop=True)

# Creating a series for the store numbers to loop through

store\_num = df\_x['store\_nbr'].unique()

# Looping through all the stores to generate the coefficient metric

for i in store\_num:

control\_val = df\_x[df\_x['store\_nbr'] == i][metric\_col].reset\_index(drop=True)

# Defining the standard distance value

standard = abs(control\_val-trial\_val)

# Calculating the magnitude by using the provided distance formula and taking the mean

# of the resulting series

magnitude = (1 - ((standard-min(standard))/(max(standard)-min(standard)))).mean()

# Saving to the dictionary variable

output['trial\_store'].append(trial\_store)

output['control\_store'].append(i)

output['mag\_distance'].append(magnitude)

# Saving the populated dictionary to a dataframe

output\_df = pd.DataFrame(output)

return output\_df

\*\*Store 77: Total Sales and Number of Customers Correlations and Magnitude Distance\*\*

trial\_store = 77

corr\_tot\_sales = calcCorrelation('tot\_sales',trial\_store,pretrial\_metric)

corr\_n\_customers = calcCorrelation('n\_customers',trial\_store,pretrial\_metric)

mag\_tot\_sales = calcMagnitudeDistance('tot\_sales',trial\_store,pretrial\_metric)

mag\_n\_customers = calcMagnitudeDistance('n\_customers',trial\_store,pretrial\_metric)

sales77 = pd.concat([corr\_tot\_sales,mag\_tot\_sales['mag\_distance']], axis=1)

sales77['sales\_score'] = (sales77['corr\_coef']\*0.5) + (sales77['mag\_distance']\*0.5)

customer77 = pd.concat([corr\_n\_customers,mag\_n\_customers['mag\_distance']], axis=1)

customer77['customer\_score'] = (customer77['corr\_coef']\*0.5) + (customer77['mag\_distance']\*0.5)

customer77.head()

sales77.head()

# Merging sales and customer scores to one dataframe

score\_control\_77 = sales77.merge(customer77,on=['trial\_store','control\_store'])

# Cleaning up the column name for the merged dataframe

score\_control\_77.rename(columns={'corr\_coef\_x': 'sales\_corr\_coef', 'mag\_distance\_x': 'sales\_mag\_distance',

'corr\_coef\_y': 'cust\_corr\_coef', 'mag\_distance\_y': 'cust\_mag\_distance'}, inplace=True)

score\_control\_77.head()

score\_control\_77['final\_control\_score'] = (score\_control\_77['sales\_score'] \* 0.5) + (score\_control\_77['customer\_score'] \* 0.5)

# Extracting the control store ussing the score metric

control\_store = score\_control\_77.sort\_values(by='final\_control\_score',

ascending=False).head(1)['control\_store'].reset\_index(drop=True).item()

# Assigning values to the initialised `store\_type` column

store\_agg['store\_type'] = (

store\_agg['store\_nbr']

.map({trial\_store: 'Trial', control\_store: 'Control'})

.fillna('Other Stores')

)

store\_agg.head()

store\_agg.columns

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='tot\_sales',

hue='store\_type',

err\_style=None)

plt.title('Total Sales by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_77\_monthly\_sales.png')

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='n\_customers',

hue='store\_type',

err\_style=None)

plt.title('Number of Customers by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_77\_n\_customers.png')

From the visualisation above we can observe the similarity in sales performance and customer volume of stores 77 and 233 during the pretrial period, hence store 233 is serves as a control store suitable for the comparative analysis.

scaling\_factor\_sales = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['tot\_sales'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['tot\_sales'].sum())

scaling\_factor\_sales

scaled\_control\_sales = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_sales = scaled\_control\_sales.assign(control\_sales = (

scaled\_control\_sales['tot\_sales'] \* scaling\_factor\_sales

))

percentDiff = scaled\_control\_sales[['year\_month','control\_sales']].reset\_index(drop=True)

percentDiff = percentDiff.merge(right=store\_agg[store\_agg['store\_nbr'] == trial\_store][['year\_month','tot\_sales']].reset\_index(drop=True).rename(columns={'tot\_sales': 'trial\_sales'}),

on='year\_month')

percentDiff['percentage\_diff'] = (

abs(percentDiff['control\_sales'] - percentDiff['trial\_sales'])/percentDiff['control\_sales']

)

stdDev = stdev(percentDiff[(percentDiff['year\_month'] < '2019-02')]['percentage\_diff'])

degreesOfFreedom = 7

percentDiff['tValue'] = (percentDiff['percentage\_diff'] - 0)/stdDev

percentDiff.head()

pastSales = (

percentDiff[['year\_month', 'trial\_sales']] # Extract columns

.rename(columns={'trial\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Trial') # Add new column

)

pastSales.head()

pastControlSales = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.rename(columns={'control\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Control') # Add new column

)

pastControlSales.head()

pastSales\_Control95 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 + (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 95th % confidence interval') # Add Store\_type column

)

pastSales\_Control95.head()

pastSales\_Control5 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 - (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 5th % confidence interval') # Add Store\_type column

)

pastSales\_Control5.head()

trialAssessment = (pd.concat([pastSales,pastControlSales,pastSales\_Control5,pastSales\_Control95], axis=0)

.sort\_values('year\_month')

.reset\_index(drop=True))

trialAssessment.head()

plt.figure(figsize=(10,6))

sns.lineplot(trialAssessment,

x='year\_month',

y='totSales',

hue='Store\_type')

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.title('Total Sales by Month')

plt.ylim(bottom = 0)

plt.legend(title = 'Store Type')

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('sales\_77.png')

plt.show()

The results show that the trial in store 77 is evidently different from the control store during the trial period, as the trial store performance is significantly better than that of the control store in two out of the three month trial period.

scaling\_factor\_customer = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['n\_customers'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['n\_customers'].sum())

scaling\_factor\_customer

scaled\_control\_customer = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_customer = scaled\_control\_customer.assign(

control\_customer = (scaled\_control\_customer['n\_customers'] \* scaling\_factor\_customer)

)

scaled\_control\_customer.head()

trial\_customer = store\_agg[['year\_month', 'store\_nbr', 'n\_customers']]

trial\_customer = trial\_customer.loc[trial\_customer['store\_nbr'] == trial\_store]

trial\_customer.drop(columns=['store\_nbr'], inplace=True)

percentageDiff = (

scaled\_control\_customer[['year\_month','control\_customer']]

.merge(right = trial\_customer, on = 'year\_month')

.rename(columns = {'n\_customers' : 'trial\_customers'})

.assign(percDiff = lambda df: abs(df['control\_customer'] - df['trial\_customers'])/df['control\_customer'])

)

percentageDiff.head()

stdDev = stdev(percentageDiff[percentageDiff['year\_month'] < '2019-02']['percDiff'])

degreesOfFreedom = 7 # 8 months under observation, hence the degree of freedom is (8 - 1 = 7)

pastCustomers = store\_agg[['year\_month','n\_customers','store\_type']]

pastCustomers = pastCustomers.loc[pastCustomers['store\_type'].isin(['Trial','Control'])].reset\_index(drop=True)

pastCustomers.head()

pastCustomers\_Control95 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 + stdDev \* 2))

.assign(store\_type = 'Control 95th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control95.head()

pastCustomers\_Control5 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 - stdDev \* 2))

.assign(store\_type = 'Control 5th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control5.head()

trialAssessment = (

pd.concat([pastCustomers,pastCustomers\_Control95,pastCustomers\_Control5], axis=0)

.sort\_values('year\_month', ascending=True)

.reset\_index(drop=True)

)

trialAssessment.head()

plt.figure(figsize=(10,6))

sns.lineplot(trialAssessment,

x='year\_month',

y='n\_customers',

hue='store\_type')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.title('Number of Customers by Month')

plt.ylim(bottom = 0)

plt.legend(title = 'Store Type')

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('customer\_77.png')

plt.show()

This customer volume analysis between stores 77 and 233 yielded similar results to the sales analysis carried out earlier. The trial store (77) performs significantly better than the control store in the last two months of the three month trial period.

\*\*Store 86: Total Sales and Number of Customers Correlations and Magnitude Distance\*\*

trial\_store = 86

corr\_tot\_sales = calcCorrelation('tot\_sales',trial\_store,pretrial\_metric)

corr\_n\_customers = calcCorrelation('n\_customers',trial\_store,pretrial\_metric)

mag\_tot\_sales = calcMagnitudeDistance('tot\_sales',trial\_store,pretrial\_metric)

mag\_n\_customers = calcMagnitudeDistance('n\_customers',trial\_store,pretrial\_metric)

sales86 = pd.concat([corr\_tot\_sales,mag\_tot\_sales['mag\_distance']], axis=1)

sales86['sales\_score'] = (sales86['corr\_coef']\*0.5) + (sales86['mag\_distance']\*0.5)

sales86.head()

customer86 = pd.concat([corr\_n\_customers,mag\_n\_customers['mag\_distance']], axis=1)

customer86['customer\_score'] = (customer86['corr\_coef']\*0.5) + (customer86['mag\_distance']\*0.5)

customer86.head()

score\_control\_86 = sales86.merge(customer86,on=['trial\_store','control\_store'])

score\_control\_86.rename(columns={'corr\_coef\_x': 'sales\_corr\_coef', 'mag\_distance\_x': 'sales\_mag\_distance',

'corr\_coef\_y': 'cust\_corr\_coef', 'mag\_distance\_y': 'cust\_mag\_distance'}, inplace=True)

score\_control\_86.head()

score\_control\_86['final\_control\_score'] = (score\_control\_86['sales\_score'] \* 0.5) + (score\_control\_86['customer\_score'] \* 0.5)

control\_store = score\_control\_86.sort\_values(by='final\_control\_score',

ascending=False).head(1)['control\_store'].reset\_index(drop=True).item()

control\_store

store\_agg['store\_type'] = (store\_agg['store\_nbr']

.map({trial\_store: 'Trial', control\_store: 'Control'})

.fillna('Other Stores'))

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='tot\_sales',

hue='store\_type',

err\_style=None)

plt.title('Total Sales by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_86\_monthly\_sales.png')

plt.show()

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='n\_customers',

hue='store\_type',

err\_style=None)

plt.title('Number of Customers by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_86\_n\_customers.png')

plt.show()

The visualisations above shows the similarity in sales performance and customer volume between the trial store (86) and the control store (155). Hence, the consequent suitability of store 155 as the control store for the comparative analysis with the trial store 86.

scaling\_factor\_sales = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['tot\_sales'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['tot\_sales'].sum())

scaling\_factor\_sales

scaled\_control\_sales = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_sales = scaled\_control\_sales.assign(

control\_sales = (scaled\_control\_sales['tot\_sales'] \* scaling\_factor\_sales)

)

percentDiff = scaled\_control\_sales[['year\_month','control\_sales']].reset\_index(drop=True)

percentDiff = percentDiff.merge(right=store\_agg[store\_agg['store\_nbr'] == trial\_store][['year\_month','tot\_sales']].reset\_index(drop=True).rename(columns={'tot\_sales': 'trial\_sales'}),

on='year\_month')

percentDiff['percentage\_diff'] = abs(percentDiff['control\_sales'] - percentDiff['trial\_sales'])/percentDiff['control\_sales']

stdDev = stdev(percentDiff[(percentDiff['year\_month'] < '2019-02')]['percentage\_diff'])

degreesOfFreedom = 7

percentDiff['tValue'] = (percentDiff['percentage\_diff'] - 0)/stdDev

percentDiff.head()

pastSales = (

percentDiff[['year\_month', 'trial\_sales']] # Extract columns

.rename(columns={'trial\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Trial') # Add new column

)

pastSales.head()

pastControlSales = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.rename(columns={'control\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Control') # Add new column

)

pastControlSales.head()

pastSales\_Control95 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 + (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 95th % confidence interval') # Add Store\_type column

)

pastSales\_Control95.head()

pastSales\_Control5 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 - (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 5th % confidence interval') # Add Store\_type column

)

pastSales\_Control5.head()

trialAssessment = (pd.concat([pastSales,pastControlSales,pastSales\_Control5,pastSales\_Control95], axis=0)

.sort\_values('year\_month')

.reset\_index(drop=True))

trialAssessment.head()

plt.figure(figsize=(13,8))

sns.lineplot(trialAssessment,

x='year\_month',

y='totSales',

hue='Store\_type')

plt.ylim(bottom = 0)

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.title('Total Sales by Month')

plt.legend(title = 'Store Type', bbox\_to\_anchor = (1,1))

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('sales\_86.png')

plt.show()

The results show that the trial in store 86 is not significantly different to its

control store in the trial period as the trial store performance lies inside the 5%

to 95% confidence interval of the control store in two of the three trial months.

scaling\_factor\_customer = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['n\_customers'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['n\_customers'].sum())

scaled\_control\_customer = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_customer = scaled\_control\_customer.assign(

control\_customer = (scaled\_control\_customer['n\_customers'] \* scaling\_factor\_customer)

)

trial\_customer = store\_agg[['year\_month', 'store\_nbr', 'n\_customers']]

trial\_customer = trial\_customer.loc[trial\_customer['store\_nbr'] == trial\_store]

trial\_customer.drop(columns=['store\_nbr'], inplace=True)

percentageDiff = (

scaled\_control\_customer[['year\_month','control\_customer']]

.merge(right = trial\_customer, on = 'year\_month')

.rename(columns = {'n\_customers' : 'trial\_customers'})

.assign(percDiff = lambda df: abs(df['control\_customer'] - df['trial\_customers'])/df['control\_customer'])

)

percentageDiff.head()

stdDev = stdev(percentageDiff[percentageDiff['year\_month'] < '2019-02']['percDiff'])

degreesOfFreedom = 7 # 8 months under observation, hence the degree of freedom is (8 - 1 = 7)

pastCustomers = store\_agg[['year\_month','n\_customers','store\_type']]

pastCustomers = pastCustomers.loc[pastCustomers['store\_type'].isin(['Trial','Control'])].reset\_index(drop=True)

pastCustomers.head()

pastCustomers\_Control95 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 + stdDev \* 2))

.assign(store\_type = 'Control 95th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control95.head()

pastCustomers\_Control5 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 - stdDev \* 2))

.assign(store\_type = 'Control 5th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control5.head()

trialAssessment = (

pd.concat([pastCustomers,pastCustomers\_Control95,pastCustomers\_Control5], axis=0)

.sort\_values('year\_month', ascending=True)

.reset\_index(drop=True)

)

trialAssessment.head()

plt.figure(figsize=(10,6))

sns.lineplot(trialAssessment,

x='year\_month',

y='n\_customers',

hue='store\_type')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.title('Number of Customers by Month')

plt.ylim(bottom = 0)

plt.legend(title = 'Store Type')

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('customer\_86.png')

plt.show()

The result shows that there is a significant difference in the customer volume during the trial months, however since this not reflected in the sales analysis. We should check with the category manager if there were special deals on chips products resulting in lower prices, consequently less sales.

\*\*Store 88: Total Sales and Number of Customers Correlations and Magnitude Distance\*\*

trial\_store = 88

corr\_tot\_sales = calcCorrelation('tot\_sales',trial\_store,pretrial\_metric)

corr\_n\_customers = calcCorrelation('n\_customers',trial\_store,pretrial\_metric)

mag\_tot\_sales = calcMagnitudeDistance('tot\_sales',trial\_store,pretrial\_metric)

mag\_n\_customers = calcMagnitudeDistance('n\_customers',trial\_store,pretrial\_metric)

sales88 = pd.concat([corr\_tot\_sales,mag\_tot\_sales['mag\_distance']], axis=1)

sales88['sales\_score'] = (sales88['corr\_coef']\*0.5) + (sales88['mag\_distance']\*0.5)

sales88.head()

customer88 = pd.concat([corr\_n\_customers,mag\_n\_customers['mag\_distance']], axis=1)

customer88['customer\_score'] = (customer88['corr\_coef']\*0.5) + (customer88['mag\_distance']\*0.5)

customer88.head()

score\_control\_88 = sales88.merge(customer88,on=['trial\_store','control\_store'])

score\_control\_88.rename(columns={'corr\_coef\_x': 'sales\_corr\_coef', 'mag\_distance\_x': 'sales\_mag\_distance',

'corr\_coef\_y': 'cust\_corr\_coef', 'mag\_distance\_y': 'cust\_mag\_distance'}, inplace=True)

score\_control\_88.head()

score\_control\_88['final\_control\_score'] = (score\_control\_88['sales\_score'] \* 0.5) + (score\_control\_88['customer\_score'] \* 0.5)

score\_control\_88.sort\_values('final\_control\_score', ascending=False).head()

After checking the significance of visualisations, store 237 was determined to be the most compatible and relevant to the trial store 88

control\_store = 237

store\_agg['store\_type'] = store\_agg['store\_nbr'].map({trial\_store: 'Trial', control\_store: 'Control'}).fillna('Other Stores')

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='tot\_sales',

hue='store\_type',

err\_style=None)

plt.title('Total Sales by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_88\_monthly\_sales.png')

plt.show()

plt.figure(figsize=(7,5))

sns.lineplot(store\_agg.loc[store\_agg['year\_month'] < '2019-02'],

x='year\_month',

y='n\_customers',

hue='store\_type',

err\_style=None)

plt.title('Number of Customers by Month')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.legend(title = 'Store Type', loc='upper left', bbox\_to\_anchor=(1, 1), ncol=1)

plt.tight\_layout()

plt.savefig('trial\_88\_n\_customers.png')

plt.show()

It can be visually observed the significance of the similarity between the trial store (88) and the control store (237) in both sales and customer volume metrics. Hence, store 237 serves as a suitable store for the comparative analysis with the trial store (88)

scaling\_factor\_sales = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['tot\_sales'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['tot\_sales'].sum())

scaling\_factor\_sales

scaled\_control\_sales = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_sales = scaled\_control\_sales.assign(

control\_sales = (scaled\_control\_sales['tot\_sales'] \* scaling\_factor\_sales)

)

percentDiff = scaled\_control\_sales[['year\_month','control\_sales']].reset\_index(drop=True)

percentDiff = percentDiff.merge(right=store\_agg[store\_agg['store\_nbr'] == trial\_store][['year\_month','tot\_sales']].reset\_index(drop=True).rename(columns={'tot\_sales': 'trial\_sales'}),

on='year\_month')

percentDiff['percentage\_diff'] = abs(percentDiff['control\_sales'] - percentDiff['trial\_sales'])/percentDiff['control\_sales']

stdDev = stdev(percentDiff[(percentDiff['year\_month'] < '2019-02')]['percentage\_diff'])

degreesOfFreedom = 7

percentDiff['tValue'] = (percentDiff['percentage\_diff'] - 0)/stdDev

percentDiff.head()

pastSales = (

percentDiff[['year\_month', 'trial\_sales']] # Extract columns

.rename(columns={'trial\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Trial') # Add new column

)

pastSales.head()

pastControlSales = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.rename(columns={'control\_sales': 'totSales'}) # Rename column

.assign(Store\_type='Control') # Add new column

)

pastControlSales.head()

pastSales\_Control95 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 + (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 95th % confidence interval') # Add Store\_type column

)

pastSales\_Control95.head()

pastSales\_Control5 = (

percentDiff[['year\_month', 'control\_sales']] # Extract columns

.assign(totSales=lambda df: df['control\_sales'] \* (1 - (stdDev \* 2))) # Compute totSales

.drop(columns=['control\_sales']) # Drop the control\_sales column

.assign(Store\_type='Control 5th % confidence interval') # Add Store\_type column

)

pastSales\_Control5.head()

trialAssessment = (pd.concat([pastSales,pastControlSales,pastSales\_Control5,pastSales\_Control95], axis=0)

.sort\_values('year\_month')

.reset\_index(drop=True))

trialAssessment.head()

plt.figure(figsize=(13,8))

sns.lineplot(trialAssessment,

x='year\_month',

y='totSales',

hue='Store\_type')

#plt.ylim(bottom = 0)

plt.xlabel('Month of Operation')

plt.ylabel('Total Sales')

plt.title('Total Sales by Month')

plt.ylim(bottom = 0)

plt.legend(title = 'Store Type', bbox\_to\_anchor = (1,1))

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('sales\_88.png')

plt.show()

The results show that there is significant difference in the sales perfomance between the trial and the control stores. The trial store performed evidently better than the control store in two months from the three month trial period.

scaling\_factor\_customer = (pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == trial\_store]['n\_customers'].sum()/

pretrial\_metric.loc[pretrial\_metric['store\_nbr'] == control\_store]['n\_customers'].sum())

scaled\_control\_customer = store\_agg.loc[store\_agg['store\_nbr'] == control\_store]

scaled\_control\_customer = scaled\_control\_customer.assign(

control\_customer = (scaled\_control\_customer['n\_customers'] \* scaling\_factor\_customer)

)

trial\_customer = store\_agg[['year\_month', 'store\_nbr', 'n\_customers']]

trial\_customer = trial\_customer.loc[trial\_customer['store\_nbr'] == trial\_store]

trial\_customer.drop(columns=['store\_nbr'], inplace=True)

percentageDiff = (

scaled\_control\_customer[['year\_month','control\_customer']]

.merge(right = trial\_customer, on = 'year\_month')

.rename(columns = {'n\_customers' : 'trial\_customers'})

.assign(percDiff = lambda df: abs(df['control\_customer'] - df['trial\_customers'])/df['control\_customer'])

)

percentageDiff.head()

stdDev = stdev(percentageDiff[percentageDiff['year\_month'] < '2019-02']['percDiff'])

degreesOfFreedom = 7 # 8 months under observation, hence the degree of freedom is (8 - 1 = 7)

pastCustomers = store\_agg[['year\_month','n\_customers','store\_type']]

pastCustomers = pastCustomers.loc[pastCustomers['store\_type'].isin(['Trial','Control'])].reset\_index(drop=True)

pastCustomers.head()

pastCustomers\_Control95 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 + stdDev \* 2))

.assign(store\_type = 'Control 95th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control95.head()

pastCustomers\_Control5 = (

pastCustomers[pastCustomers['store\_type'] == 'Control']

.assign(n\_customers = lambda df: df['n\_customers'] \* (1 - stdDev \* 2))

.assign(store\_type = 'Control 5th % confidence interval')

.reset\_index(drop=True)

)

pastCustomers\_Control5.head()

trialAssessment = (

pd.concat([pastCustomers,pastCustomers\_Control95,pastCustomers\_Control5], axis=0)

.sort\_values('year\_month', ascending=True)

.reset\_index(drop=True)

)

trialAssessment.head()

plt.figure(figsize=(10,6))

sns.lineplot(trialAssessment,

x='year\_month',

y='n\_customers',

hue='store\_type')

plt.xlabel('Month of Operation')

plt.ylabel('Number of Customers')

plt.title('Number of Customers by Month')

plt.ylim(bottom = 0)

plt.legend(title = 'Store Type')

plt.axvspan(xmin='2019-02', xmax='2019-04', color = '#d65f5f', alpha = 0.1)

plt.tight\_layout()

plt.savefig('customers\_88.png')

plt.show()

Total number of customers in the trial period for the trial store is significantly

higher than the control store for two out of three months, which indicates a

positive trial effect.

\*\*Conclusion\*\*

Based on the exploratory data analysis, trial stores 77 and 86 demonstrated a significant increase in the number of customers for at least two of the three months during the trial period. In contrast, while trial store 88 did show a notable rise in customer numbers, this increase was not statistically significant according to the analysis.