Experimentation & Uplift Analysis

Loading the necessary packages and modules

```
# Importing important modules and packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
from statistics import stdev
from scipy.stats import t
```

Loading the dataset, completed Exploratory Data Analysis

```
1 data = pd.read_csv('C:/Users/LENOVO/Desktop/Task1.csv')
```

EDA has already been performed on this dataset.

```
1 data.shape
```

```
(88827, 12)
```

Let's check the dataset

	DATE	STORE_NBR	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	BRAND	PACK_SIZE
0	2018-10-17	1	1000	YOUNG SINGLES/COUPLES	Premium	1	5	Natural Chip Compny SeaSalt175g	2	Natural Chip Company	175g
1	2019-05-14	1	1307	MIDAGE SINGLES/COUPLES	Budget	348	66	CCs Nacho Cheese 175g	3	CCs	175g
2	2019-05-20	1	1343	MIDAGE SINGLES/COUPLES	Budget	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	Smiths	170g
3	2018-08-17	2	2373	MIDAGE SINGLES/COUPLES	Budget	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	Smiths	175g
4	2018-08-18	2	2426	MIDAGE SINGLES/COUPLES	Budget	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	Kettle	150g
5	2019-05-16	4	4149	MIDAGE SINGLES/COUPLES	Budget	3333	16	Smiths Crinkle Chips Salt & Vinegar 330g	1	Smiths	330g
6	2018-08-20	5	5026	MIDAGE SINGLES/COUPLES	Budget	4525	42	Doritos Corn Chip Mexican Jalapeno 150g	1	Doritos	150g

Extracting month from data & removing stores which don't have sales in all months

```
df['DATE'] = pd.to_datetime(df['DATE'])
df['MONTH_YEAR'] = pd.to_datetime(df['DATE']).dt.strftime('%Y-%m')
```

```
df['MONTH_YEAR'].head()
```

```
0    2018-10
1    2019-05
2    2019-05
3    2018-08
4    2018-08
Name: MONTH_YEAR, dtype: object
```

```
df['MONTH_YEAR'].dtypes
```

```
dtype('O')
```

```
df1 = df.groupby('STORE_NBR')['MONTH_YEAR'].nunique()

df1 = df1.to_frame()
```

```
df2 = df1[df1['MONTH_YEAR'] != 12]
```

```
df2.index.values
```

```
array([ 11,  14,  31,  42,  44,  52,  92,  99, 117, 127, 132, 139, 140,
       146, 158, 159, 161, 177, 192, 198, 204, 206, 218, 224, 244, 258,
       263, 267], dtype=int64)
```

```
for index in df2.index.values:
    df = df[df['STORE_NBR'] != index]
```

Now the dataset only consists of stores which made sales in all the months from June 2018 to July 2019

Now we will create the metrics that will be used for firther analysis of the trial and control stores.

The metrics include:

1. Total Sales for each store in each month
2. Number of customers in each store
3. Transactions per Customer in each store for every month
4. Chips bought per customer
5. Average unit Price

```
metric = df.groupby(['STORE_NBR', 'MONTH_YEAR'])['TOT_SALES', 'PROD_QTY'].sum()
```

```
j = df.groupby(['STORE_NBR', 'MONTH_YEAR'])['LYLTY_CARD_NBR'].nunique()

j = j.to_frame()
```

```
metric = pd.merge(left=metric, right=j, on=['MONTH_YEAR', 'STORE_NBR'])
```

```
metric['Transaction_per_customer'] = metric['TOT_SALES']/metric['LYLTY_CARD_NBR']
metric['Chips_per_customer'] = metric['PROD_QTY']/metric['LYLTY_CARD_NBR']

metric['Averagr Price'] = metric['TOT_SALES']/metric['PROD_QTY']
```

```
metric.rename(columns={'LYLTY_CARD_NBR':'Customer_per_store', 'TOT_SALES':'Total Sales', 'PROD_QTY':'Quantity'},
              inplace=True)
```

Let's make a table for the pre-trial period

```
metric_data = metric_data.loc[(['2018-07', '2018-08', '2018-09', '2018-10', '2018-11',
                                '2018-12', '2019-01'], slice(None)), :]
```

Creating a function to find the correlation between potential control stores and trial stores based on metrics

```
def calcorr(inputTable, metrics, trial_store):

    output = pd.DataFrame({'Store1':[], 'Store2':[], 'Correlation':[]})

    a = inputTable.loc[inputTable['STORE_NBR'] == trial_store, metrics]
    a.reset_index(drop=True, inplace=True)

    Store_Numbers = inputTable['STORE_NBR'].unique()

    for i in Store_Numbers:
        b = inputTable.loc[inputTable['STORE_NBR'] == i, metrics]
        b.reset_index(drop=True, inplace=True)

        output = output.append({'Store1':trial_store, 'Store2':i, 'Correlation':b.corr(a)}, ignore_index=True)

    return output
```

This function will take in the complete data for all the stores, the metric for which the correlation has to be calculated and, the trial store number among 77, 86 and, 88. It calculates the correlation between every pair of trial and control stores for the specified trial store and returns the output

Creating a function to find the magnitude of distance between the trial and control stores

```
def caldist(inputTable, metrics, trial_store):

    output = pd.DataFrame({'Store1':[], 'Store2':[], 'Magnitude':[]})

    a = inputTable.loc[inputTable['STORE_NBR'] == trial_store, metrics]
    a.reset_index(drop=True, inplace=True)

    Store_Numbers = inputTable['STORE_NBR'].unique()

    for i in Store_Numbers:
        b = inputTable.loc[inputTable['STORE_NBR'] == i, metrics]
        b.reset_index(drop=True, inplace=True)

        c = abs(a-b)
        d = np.mean(1-(c-min(c))/(max(c)-min(c)))

        output = output.append({'Store1':trial_store, 'Store2':i, 'Magnitude':d}, ignore_index=True)

    return output
```

This function calculates the magnitude of difference between the trial store and control store for the given metrics and characteristics, makes a dataset giving out the absolute difference.

Experimentation on Trial Store 77

Determining the Control Store for Store 77

```
trial_store = 77

corr_sales = calcorr(metric_data, 'Total Sales', trial_store)
corr_customer = calcorr(metric_data, 'Customer_per_store', trial_store)
corr_avg_price = calcorr(metric_data, 'Averagr Price', trial_store)

mag_sales = caldist(metric_data, 'Total Sales', trial_store)
mag_customer = caldist(metric_data, 'Customer_per_store', trial_store)
mag_avg_price = caldist(metric_data, 'Averagr Price', trial_store)

score_sales = pd.concat([corr_sales, mag_sales['Magnitude']], axis=1)

# Adding an additional column in 'score_sales' which calculates the weighted average

corr_weight = 0.5

score_sales['Sales Score'] = corr_weight*score_sales['Correlation'] + (1-corr_weight)*score_sales['Magnitude']
score_sales.head(10)
```

This will create a dataset having the magnitude of difference and correlation for “Total Sales” between the given trial store and all the potential control stores.

Similarly, we will create a dataset for other metrics such as “Number of Customers per Store” and “Average Price per Customer”.

First, we will create a final weighted score using a **correlation weight of 0.5** for each metric and then combine them.

	Store1	Store2	Correlation	Magnitude	Sales Score
0	77.0	1.0	-0.341275	0.403486	0.031106
1	77.0	2.0	-0.110906	0.410989	0.150041
2	77.0	3.0	-0.134088	0.485765	0.175839
3	77.0	4.0	-0.364167	0.403795	0.019814
4	77.0	5.0	-0.530441	0.581823	0.025691
5	77.0	6.0	-0.083126	0.494748	0.205811
6	77.0	7.0	-0.187441	0.459983	0.136271
7	77.0	8.0	0.479138	0.493050	0.486094
8	77.0	9.0	-0.343593	0.567736	0.112071
9	77.0	10.0	-0.322208	0.530270	0.104031

	Store1	Store2	Correlation	Magnitude	Customer Score
0	77.0	1.0	0.205764	0.587302	0.396533
1	77.0	2.0	-0.224065	0.482143	0.129039
2	77.0	3.0	-0.081832	0.428571	0.173370
3	77.0	4.0	-0.176079	0.450000	0.136960
4	77.0	5.0	-0.278693	0.511278	0.116292
5	77.0	6.0	0.304114	0.547619	0.425867
6	77.0	7.0	-0.072336	0.321429	0.124546
7	77.0	8.0	0.143385	0.547619	0.345502
8	77.0	9.0	-0.378962	0.632653	0.126846
9	77.0	10.0	-0.620336	0.470899	-0.074718

```
score_customer = pd.concat([corr_customer, mag_customer['Magnitude']], axis=1)

score_customer['Customer Score'] = corr_weight*score_customer['Correlation'] + (1-corr_weight)*score_customer['Magnitude']
```

```
score_avg_price = pd.concat([corr_avg_price, mag_avg_price['Magnitude']], axis=1)

score_avg_price['Avg Price Score'] = corr_weight*score_avg_price['Correlation']
                                   + (1-corr_weight)*score_avg_price['Magnitude']
```

	Store1	Store2	Correlation	Magnitude	Avg Price Score
0	77.0	1.0	-0.799493	0.539282	-0.130106
1	77.0	2.0	-0.514532	0.743444	0.114456
2	77.0	3.0	-0.168485	0.636928	0.234222
3	77.0	4.0	-0.027489	0.593611	0.283061
4	77.0	5.0	-0.045324	0.739516	0.347096
5	77.0	6.0	0.513189	0.657470	0.585330
6	77.0	7.0	-0.304087	0.499314	0.097614
7	77.0	8.0	0.748430	0.465636	0.607033
8	77.0	9.0	0.033687	0.571109	0.302398
9	77.0	10.0	0.565272	0.534494	0.549883

Now, we will combine all the weighted scores for each metric.

```
score_sales.set_index(['Store1', 'Store2'], inplace=True)
score_customer.set_index(['Store1', 'Store2'], inplace=True)
score_avg_price.set_index(['Store1', 'Store2'], inplace=True)

score_control = pd.concat([score_sales['Sales Score'], score_customer['Customer Score'],
                           score_avg_price['Avg Price Score']],
                          axis=1)
```

Let's see a part of the data

Store1	Store2	Sales Score	Customer Score	Avg Price Score
	1.0	0.031106	0.396533	-0.130106
	2.0	0.150041	0.129039	0.114456
	3.0	0.175839	0.173370	0.234222
	4.0	0.019814	0.136960	0.283061
	5.0	0.025691	0.116292	0.347096
	6.0	0.205811	0.425867	0.585330
	7.0	0.136271	0.124546	0.097614
77.0	8.0	0.486094	0.345502	0.607033
	9.0	0.112071	0.126846	0.302398
	10.0	0.104031	-0.074718	0.549883
	12.0	0.108118	0.083492	0.229378
	13.0	0.235170	0.472673	0.276563
	15.0	-0.026031	-0.030218	0.029103
	16.0	-0.039767	-0.060113	0.317294
	17.0	0.462985	0.411187	0.209071

During the analysis, we found out that "Average Price Score" doesn't play much of a role in defining the final score between control scores and trial store

Let's devise the control store

```
score_control['Final'] = 0.5 * (score_control['Sales Score'] + score_control['Customer Score'])
```

```
score_control.sort_values(by='Final', ascending=False).head()
```

Store1	Store2	Sales Score	Customer Score	Avg Price Score	Final
	38.0	0.714492	0.825870	0.339815	0.770181
	187.0	0.767113	0.728812	0.220831	0.747962
77.0	265.0	0.757670	0.680050	0.386710	0.718860
	94.0	0.692070	0.701354	0.583919	0.696712
	88.0	0.743402	0.581507	0.107780	0.662454

We can see that the control store for trial Store 77 is **Store 38**, has the highest calculated final score of **0.77**.

```
control_sore = 38

past_Sales = metric_data.copy()

store_type = []

for i in past_Sales['STORE_NBR']:
    if i == trial_store:
        store_type.append('Trial Store')
    elif i == control_sore:
        store_type.append('Control Store')
    else:
        store_type.append('Other Stores')

past_Sales['Store_Type'] = store_type
past_Sales.head()
```

```
STORE_NBR                    int64
MONTH_YEAR                  object
Total Sales                float64
Quantity                     int64
Customer_per_store           int64
Transaction_per_customer   float64
Chips_per_customer         float64
Averagr Price              float64
Store_Type                  object
MONTH                datetime64[ns]
dtype: object
```

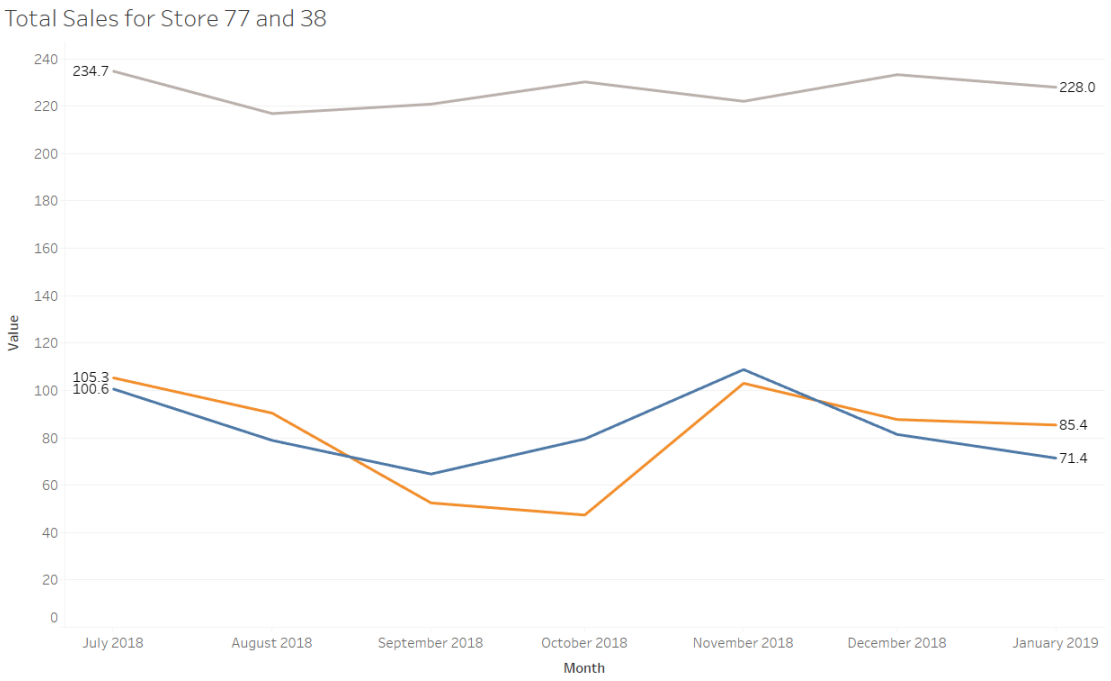
```
past_Sales['MONTH'] = pd.to_datetime(past_Sales['MONTH_YEAR'])
```

Now, we will analyze “Total Sales” & “Total Customers” for Store 77 and Store 38

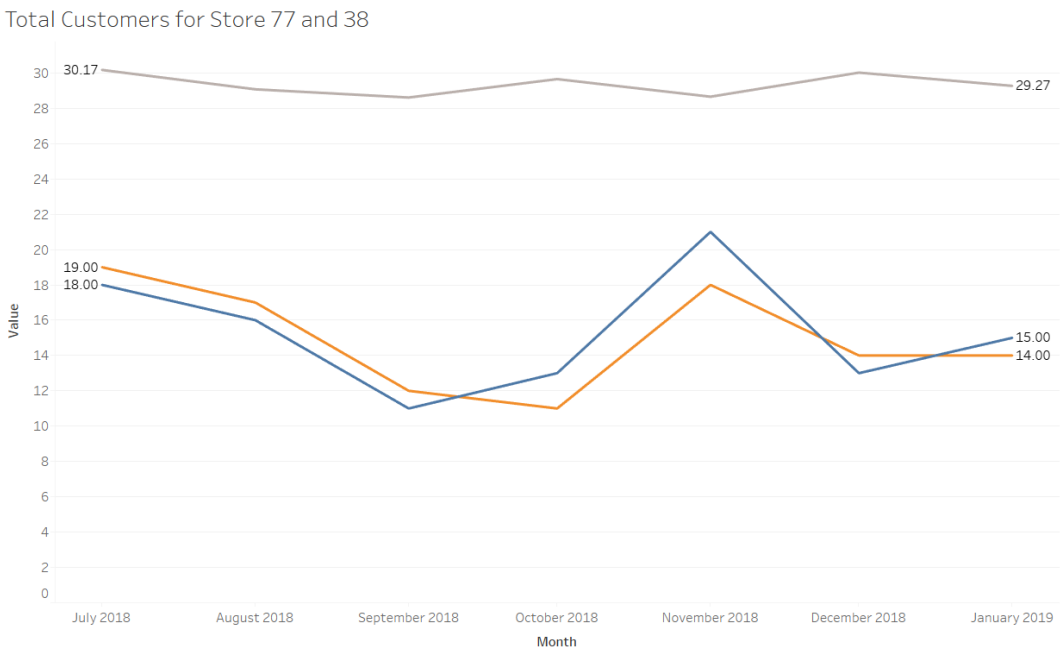
```
trial_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Trial Store', ['MONTH', 'Total Sales']]
trial_77.set_index('MONTH', inplace=True)
trial_77.rename(columns={'Total Sales':'Trial Store'}, inplace=True)

control_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Control Store', ['MONTH', 'Total Sales']]
control_77.set_index('MONTH', inplace=True)
control_77.rename(columns={'Total Sales':'Control Store'}, inplace=True)

others_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Other Stores', ['MONTH', 'Total Sales']]
others_77 = pd.DataFrame(others_77.groupby('MONTH')['Total Sales'].mean())
others_77.rename(columns={'Total Sales':'Other Stores'}, inplace=True)

combine_sales = pd.concat([control_77, trial_77, others_77], axis=1)

combine_sales.to_csv('Combines_Sales_77.csv', index=True)
```

```
trial_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Trial Store', ['MONTH', 'Customer_per_store']]
trial_77.set_index('MONTH', inplace=True)
trial_77.rename(columns={'Customer_per_store':'Trial Store'}, inplace=True)

control_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Control Store', ['MONTH', 'Customer_per_store']]
control_77.set_index('MONTH', inplace=True)
control_77.rename(columns={'Customer_per_store':'Control Store'}, inplace=True)

others_77 = past_Sales.loc[past_Sales['Store_Type'] == 'Other Stores', ['MONTH', 'Customer_per_store']]
others_77 = pd.DataFrame(others_77.groupby('MONTH')['Customer_per_store'].mean())
others_77.rename(columns={'Customer_per_store':'Other Stores'}, inplace=True)

combine_customer = pd.concat([control_77, trial_77, others_77], axis=1)

combine_customer.to_csv('Combines_Cusotmers_77.csv', index=True)
```

Let's visualize both metrics separately



We can see that Store 38 and 77 are closely related to each other *(77 – "Orange", 38 – "Blue")*. Their trends of sales are almost equal for each month except **September - October 2018** when **Store 38 gives total sales of 64.5 and 79.5** respectively **higher than that of Store 77**



We can see that Store 38 and 77 are closely related to each other *(77 – "Orange", 38 – "Blue")*. Their trends of **total customers are almost equal for each month except November 2018** when Store 38 has 21 total customers.

Now, we have to test our hypothesis whether our conclusions were not by chance but logical and are giving satisfactory results in **the trial period of February 2019 – April 2019.**

To test whether the results are satisfactory, we will create **confidence intervals of 5% and 95%.**

Assessment of Store 77

Total Sales

```
trial_sum = past_Sales.loc[past_Sales['Store_Type'] == 'Trial Store', 'Total Sales'].sum()
control_sum = past_Sales.loc[past_Sales['Store_Type'] == 'Control Store', 'Total Sales'].sum()
scaling_sales = trial_sum/control_sum
scaling_sales
```

```
0.9767640526225868
```

```
scaled_control_stores = metric.copy()

scaled_control_stores = scaled_control_stores.loc[scaled_control_stores['STORE_NBR'] == control_sore]


scaled_control_stores['Control Sales'] = scaled_control_stores['Total Sales']*scaling_sales
scaled_control_stores.head()
```

	STORE_NBR	MONTH_YEAR	Total Sales	Quantity	Customer_per_store	Transaction_per_customer	Chips_per_customer	Averagr Price
408	38	2018-07	100.6	27	18	5.588889	1.500000	3.725926
409	38	2018-08	78.9	23	16	4.931250	1.437500	3.430435
410	38	2018-09	64.7	19	11	5.881818	1.727273	3.405263
411	38	2018-10	79.5	20	13	6.115385	1.538462	3.975000
412	38	2018-11	108.8	34	21	5.180952	1.619048	3.200000
413	38	2018-12	81.4	23	13	6.261538	1.769231	3.539130
414	38	2019-01	71.4	23	15	4.760000	1.533333	3.104348
415	38	2019-02	81.8	25	16	5.112500	1.562500	3.272000
416	38	2019-03	64.6	20	12	5.383333	1.666667	3.230000
417	38	2019-04	74.9	22	13	5.761538	1.692308	3.404545
418	38	2019-05	55.0	16	12	4.583333	1.333333	3.437500
419	38	2019-06	94.1	29	17	5.535294	1.705882	3.244828

We calculate the percentage difference of the Sales of Store 38 and 77 for each month.

```
percent_off = scaled_control_stores[['MONTH_YEAR', 'Control Sales']]
percent_off.reset_index(drop=True, inplace=True)

trial_sales = metric.loc[metric['STORE_NBR'] == trial_store, 'Total Sales']
trial_sales.reset_index(drop=True, inplace=True)

percent_off = pd.concat([percent_off, trial_sales], axis=1)
percent_off.rename(columns={'Total Sales':'Trial Sales'}, inplace=True)

percent_off['Percentage_diff'] = abs(percent_off['Control Sales'] - percent_off['Trial Sales'])/percent_off['Control Sales']
percent_off
```

Our Null hypothesis is such that the trial period is the same as the pre-trial period

We will now calculate the standard deviation on the percentage for the pre-trial period

```
std_dev = stdev(percent_off.loc[percent_off['MONTH_YEAR'] < '2019-02', 'Percentage_diff'])
std_dev
```

```
0.11876720305646335
```

The **degree of freedom comes to be 6**. As there are a total of **7 entries** before Feb 2019, the degree of freedom will be (7 - 1 = 6) six.

```
percent_off['tvalue'] = percent_off['Percentage_diff']/std_dev
percent_off.loc[(percent_off['MONTH_YEAR'] > '2019-01') & (percent_off['MONTH_YEAR'] < '2019-05'), 'tvalue']
```

```
7    0.431986
8    1.214430
9    4.216886
Name: tvalue, dtype: float64
```

We calculate the T-Value for the Percentage difference for Sales

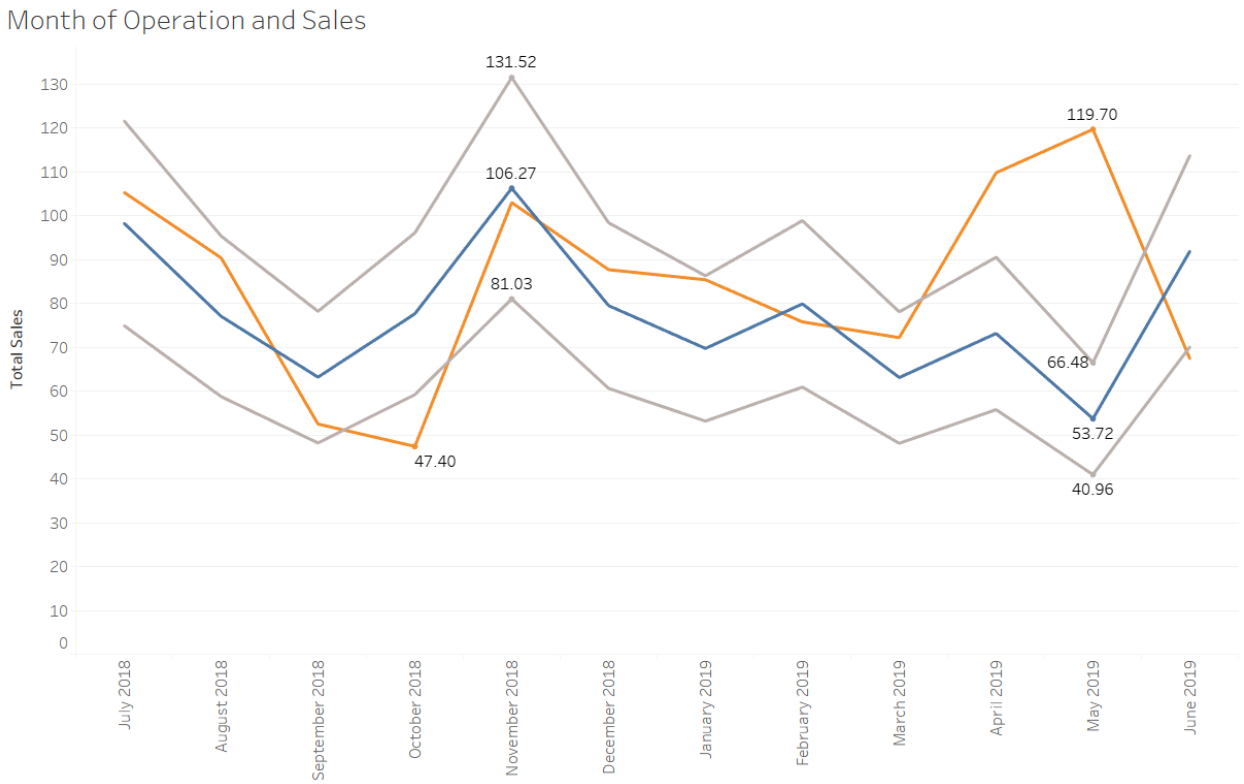
- The **standard deviation** comes out to be **0.11876**
- For **February 2019**, the **T-Value** is **0.431986**, **March 2019 – 1.214430, April 2019 – 4.216886**
- The **95th percentile for T-distribution** with a degree of freedom 6, comes out to be **1.94318**

We will now extract the control store's sales for all months and also evaluate the confidence intervals

```
controlSales = scaled_control_stores.loc[:, ['MONTH', 'Control Sales']]
controlSales.set_index('MONTH', inplace=True)

# Let's calculate the 5% and 95% confidence interval
controlSales['Control 5% Confidence Interval'] = controlSales['Control Sales'] * (1 - std_dev*2)
controlSales['Control 95% Confidence Interval'] = controlSales['Control Sales'] * (1 + std_dev*2)
controlSales
```

and combine it with trial scores

MONTH	Control Sales	Control 5% Confidence Interval	Control 95% Confidence Interval	Trial Sales
2018-07-01	98.262464	74.921748	121.603180	105.3
2018-08-01	77.066684	58.760695	95.372673	90.4
2018-09-01	63.196634	48.185259	78.208009	52.5
2018-10-01	77.652742	59.207544	96.097940	47.4
2018-11-01	106.271929	81.028689	131.515168	103.0
2018-12-01	79.508594	60.622567	98.394621	87.7
2019-01-01	69.740953	53.175077	86.306829	85.4
2019-02-01	79.899300	60.920467	98.878132	75.8
2019-03-01	63.098958	48.110784	78.087131	72.2
2019-04-01	73.159628	55.781699	90.537556	109.8
2019-05-01	53.722023	40.961194	66.482852	119.7
2019-06-01	91.913497	70.080879	113.746115	67.4



We can see the sales of Store 77 lies under the confidence interval of sales of Store 38 in the pre-trial period. For the trial-period, the sales are well correlated with that of Store 38 except for the month of **April 2019**, when the **sales in Store 77 boom to 119.70** and that of Store 38 is 53.72.

Total Number of Customers

Performing the same operation for "Number of Customers" each month.

```
std_devc = stdev(percent_offc.loc[percent_offc['MONTH_YEAR'] < '2019-02', 'Percentage_diff'])
std_devc
```

```
0.030875714127836965
```

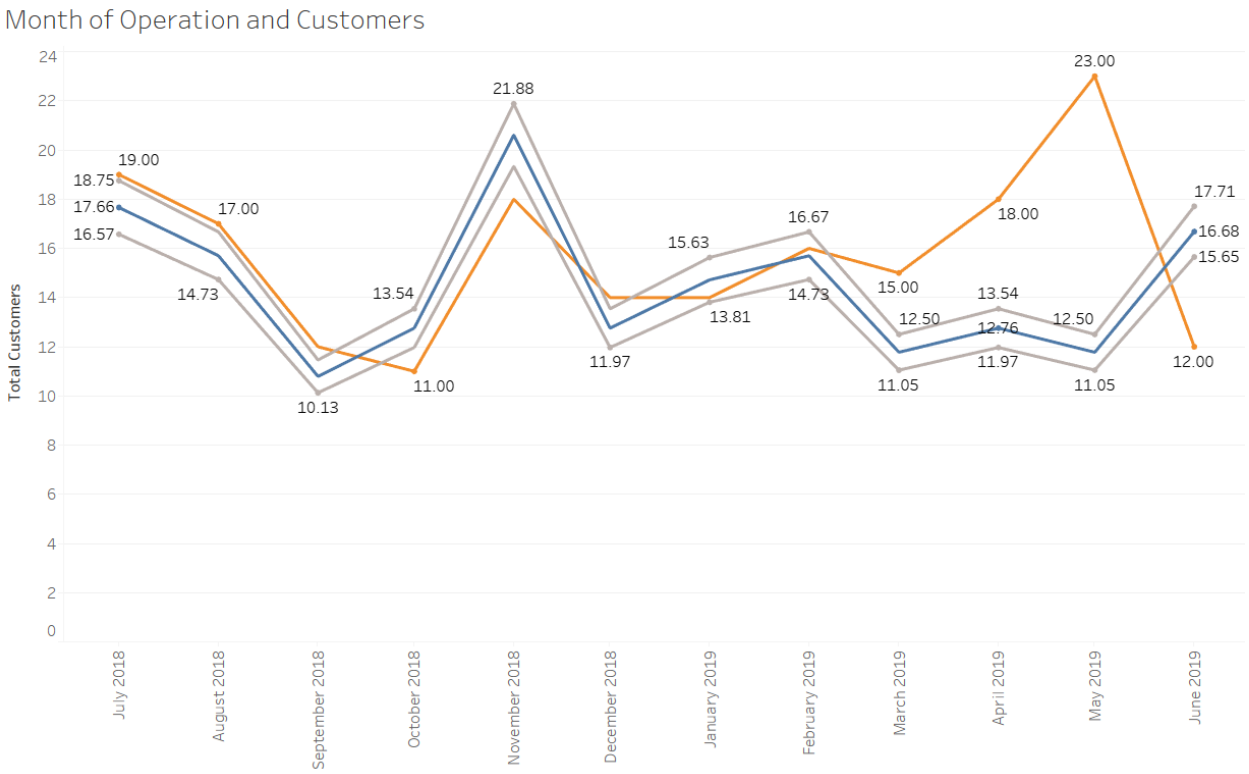
```
percent_offc['tvalue'] = percent_offc['Percentage_diff']/std_devc
percent_offc.loc[(percent_offc['MONTH_YEAR'] > '2019-01') & (percent_offc['MONTH_YEAR'] < '2019-05'), 'tvalue']
```

```
7     0.616913
8     8.868120
9    13.311077
Name: tvalue, dtype: float64
```

We calculate the T-Value for the Percentage difference for Sales and displaying the T-Values for the months in the trial-period.

- The **standard deviation** comes out to be **0.030875**
- For **February 2019**, the **T-Value** is **0.616913**, **March 2019** – **8.868120**, **April 2019** – **13.311077**

MONTH	Control Customers	Control 5% Confidence Interval	Control 95% Confidence Interval	Trial Customers
2018-07-01	17.663551	16.572802	18.754301	19
2018-08-01	15.700935	14.731379	16.670490	17
2018-09-01	10.794393	10.127823	11.460962	12
2018-10-01	12.757009	11.969246	13.544773	11
2018-11-01	20.607477	19.334936	21.880018	18
2018-12-01	12.757009	11.969246	13.544773	14
2019-01-01	14.719626	13.810668	15.628584	14
2019-02-01	15.700935	14.731379	16.670490	16
2019-03-01	11.775701	11.048535	12.502867	15
2019-04-01	12.757009	11.969246	13.544773	18
2019-05-01	11.775701	11.048535	12.502867	23
2019-06-01	16.682243	15.652091	17.712395	12



We can see the number of customers in Store 77 lies under the confidence interval of that of Store 38 in the pre-trial period. For the **trial-period, the "total customers" are well correlated** with that of Store 38 **except for the months of March 2019 – April 2019**, when the number of customers in **Store 77 drastically increase to 15-23** and that of Store 38 is 13.

**To analyze and assess different aspects of Store 86 and Store 8, we will be performing the same operations.*

Experimentation on Trial Store 86

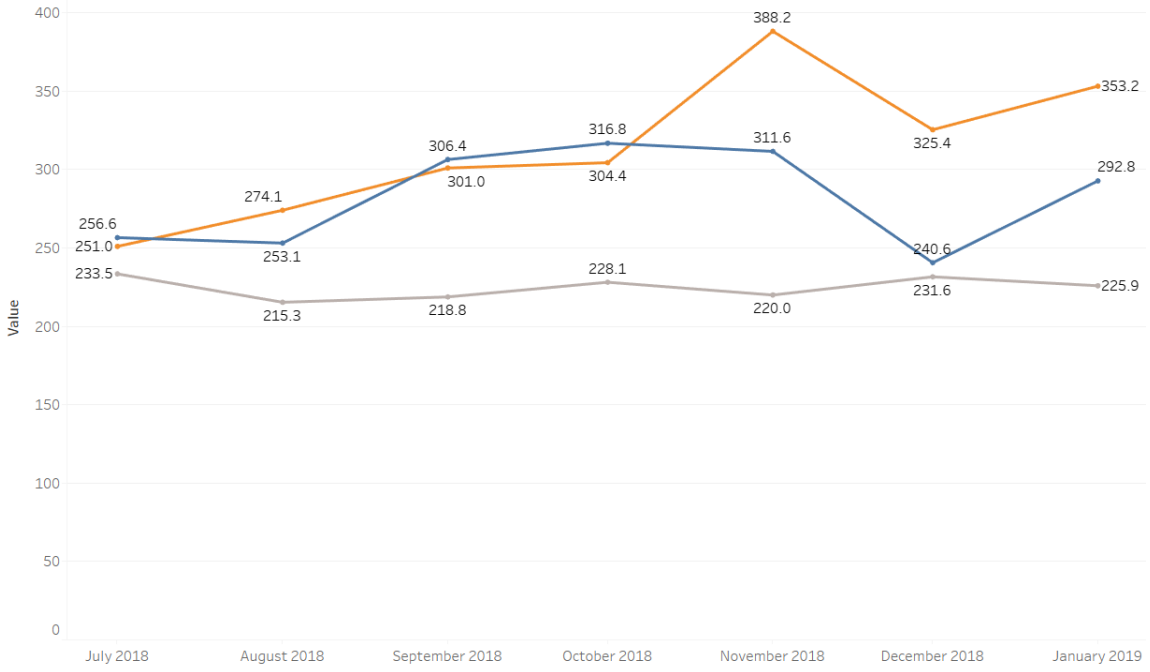
Determining the Control Store for Store 86

Store1	Store2	Sales Score	Customer Score	Avg Price Score	Final
86.0	1.0	0.196399	-0.011379	0.087782	0.092510
	2.0	0.169018	0.324887	0.572277	0.246953
	3.0	0.064209	0.259962	-0.077748	0.162086
	4.0	-0.091960	-0.147859	0.094755	-0.119909
	5.0	0.500972	0.467152	0.079934	0.484062

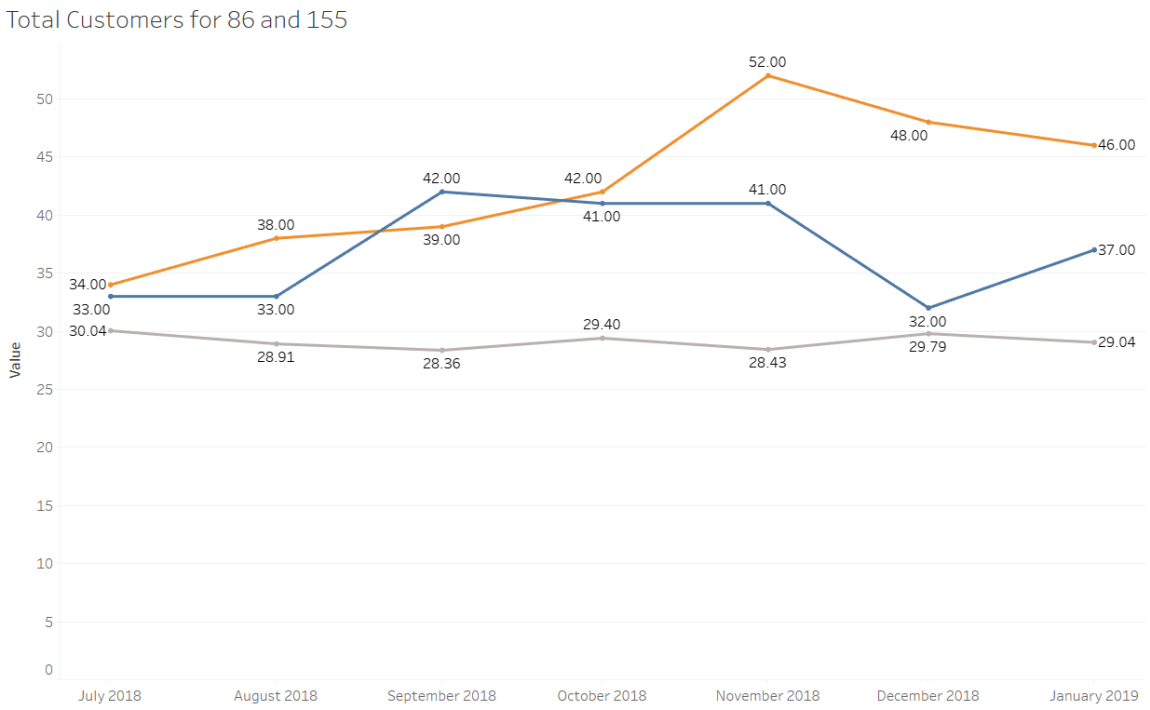
We get the control store as Store 155.

Let's visualize both metrics separately

Total Sales for 86 and 155



We can see that **Store 86 & 155 are closely related** to each other *(86 – "Orange", 155 – "Blue")*. Their trends of sales are almost equal for each month **except November 2018 when Store 86 gives total sales of 388.2 as compared to Store 155 which gives 311.6.**



We can see that Store 86 & 155 are related to each other *(86 – "Orange", 155 – "Blue")* for the months before October 2018. **Their trends of total customers get a little distorted as they reach the end of the year 2018**. This could be one of the reasons for the sales booming in the same months.

Assessment of Store 86

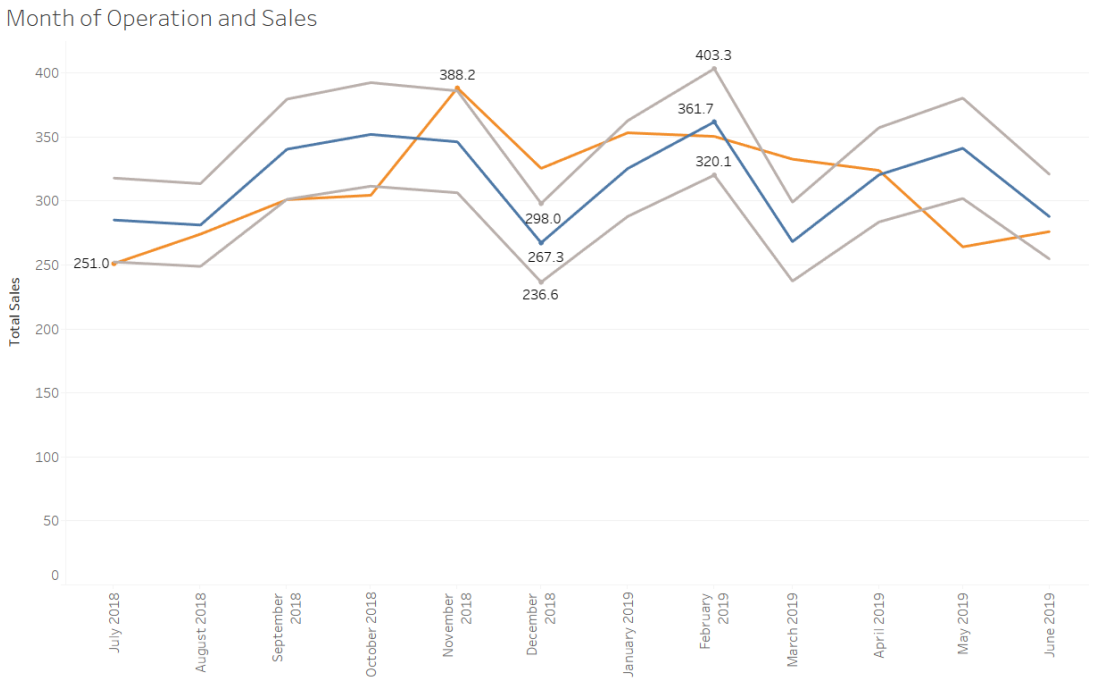
Total Sales

```
percent_off['Percentage_diff'] = abs(percent_off['Control Sales'] - percent_off['Trial Sales'])/percent_off['Control Sales']


std_dev = stdev(percent_off.loc[percent_off['MONTH_YEAR'] < '2019-02', 'Percentage_diff'])

percent_off['tvalue'] = percent_off['Percentage_diff']/std_dev
percent_off.loc[(percent_off['MONTH_YEAR'] > '2019-01') & (percent_off['MONTH_YEAR'] < '2019-05'), 'tvalue']
```

```
7    0.544044
8    4.180504
9    0.185545
Name: tvalue, dtype: float64
```

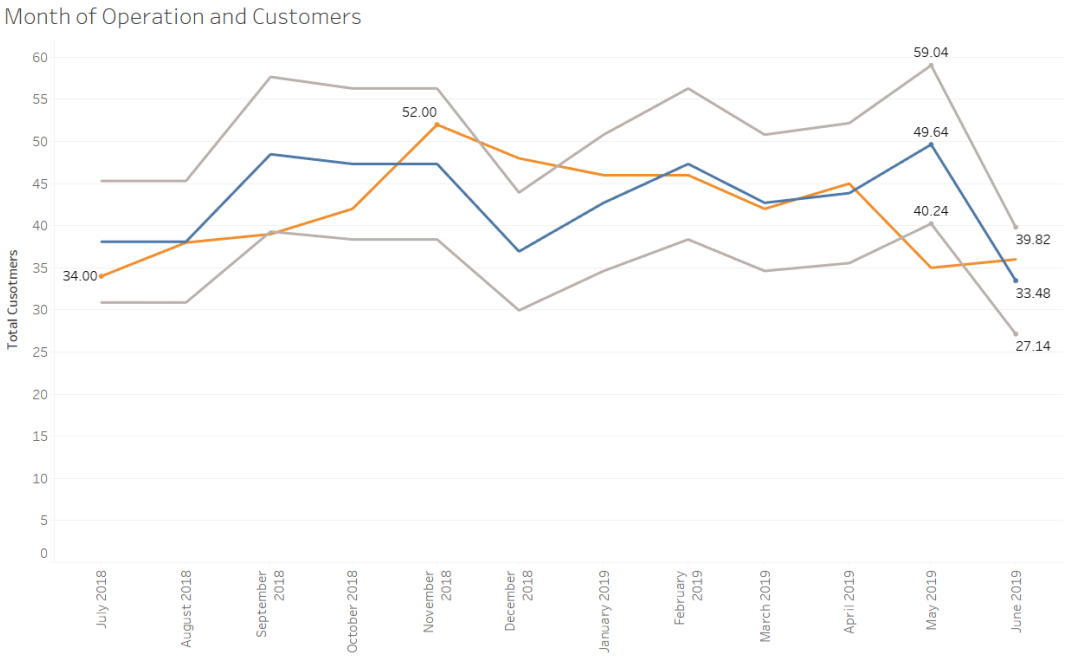
We calculate the T-Value for the Percentage difference for Sales

- The **standard deviation** comes out to be **0.057469**
- For **February 2019**, the **T-Value** is **0.544044**, **March 2019 – 4.180504, April 2019 – 0.185545**
- The 95th percentile for T-distribution with the degree of freedom 6, comes out to be 1.94318



We can see the sales of Store 86 lies under the confidence interval of sales of Store 155 in the pre-trial period. For the trial-period, the sales are well correlated with that of Store 155 **except for the month of March 2019** with slight distortion, when the sales in Store 86 remain average.

Total Number of Customers



We can see the sales of Store 86 lies under the confidence interval of sales of Store 155 for the whole year and **shows trends similar to Store 155** in 2019 and especially in the trial period.

Experimentation on Store 88

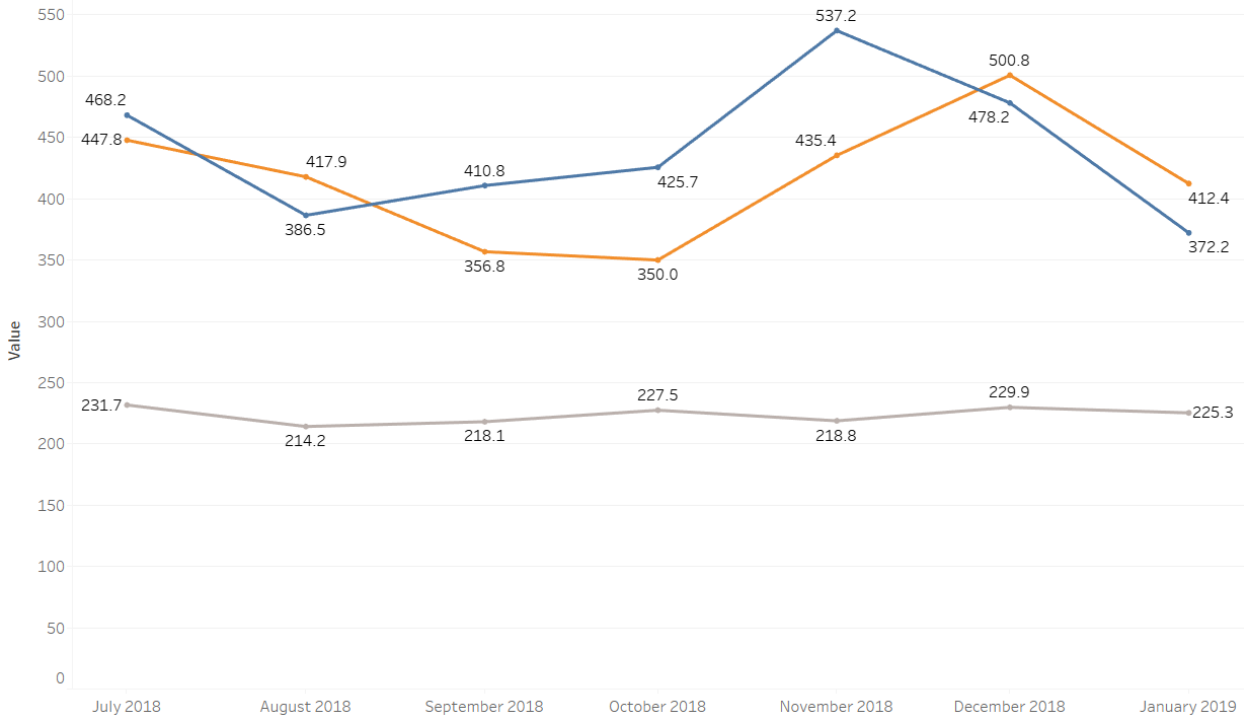
Determining the Control Store for Store 88

Store1	Store2	Sales Score	Customer Score	Avg Price Score	Final
88.0	1.0	-0.008900	0.038584	0.219070	0.014842
	2.0	0.023862	0.132533	0.550978	0.078198
	3.0	0.439119	0.429472	0.196824	0.434295
	4.0	0.011417	-0.044971	0.217053	-0.016777
	5.0	0.006044	0.030745	0.085577	0.018394

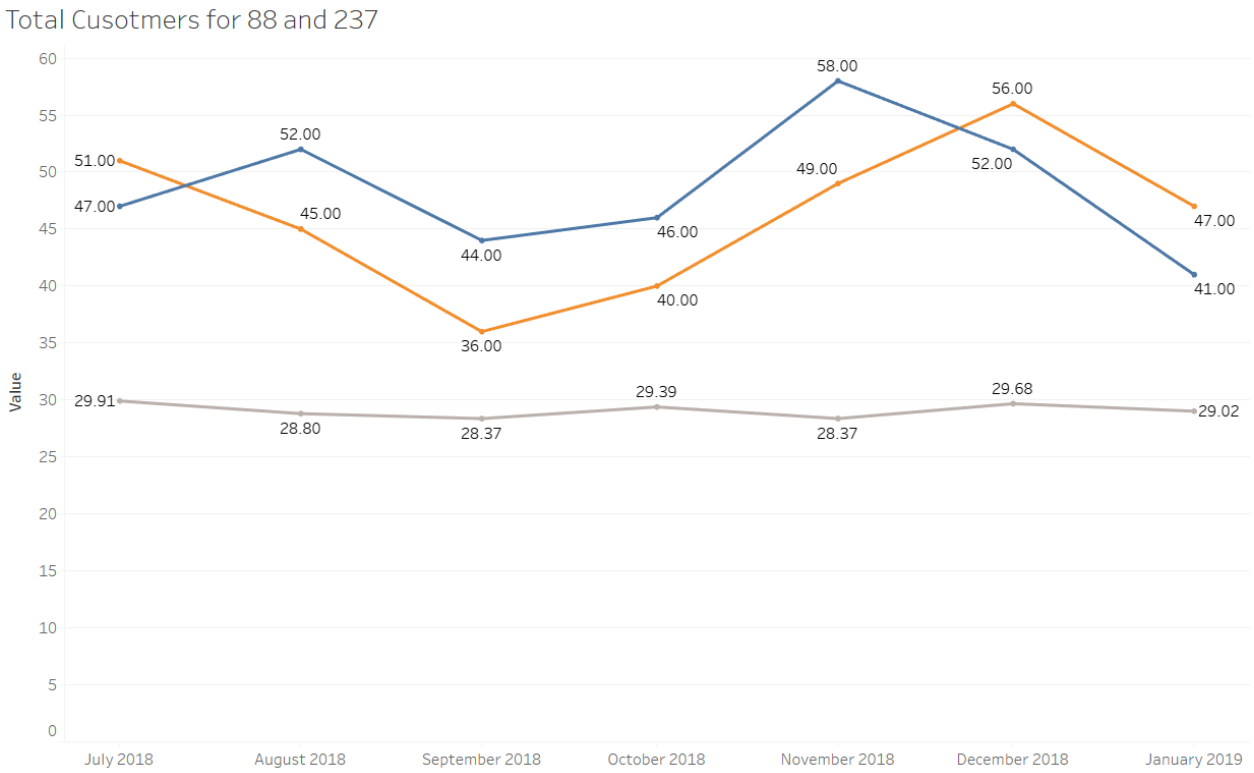
The Control Store comes out to be Store 237.

Let's visualize both metrics separately

Total Sales for 88 and 237



We can see that **Store 88 follows the trend closely to Store 237 at the end of 2018 and January 2019** as the **total sales** are not too far from each other being at **412.4 and 372.2** respectively. In the months of September to October 2018, we can see a similar trend in slopes but don't follow each other closely.



Both the stores 88 and 237 follow very similar trends from August 2018 to January 2019. Store 88 seems to get closely correlated to Store 237 in **December 2018 to January 2019** with a **difference of 4-6 customers**.

Assessment of Store 88

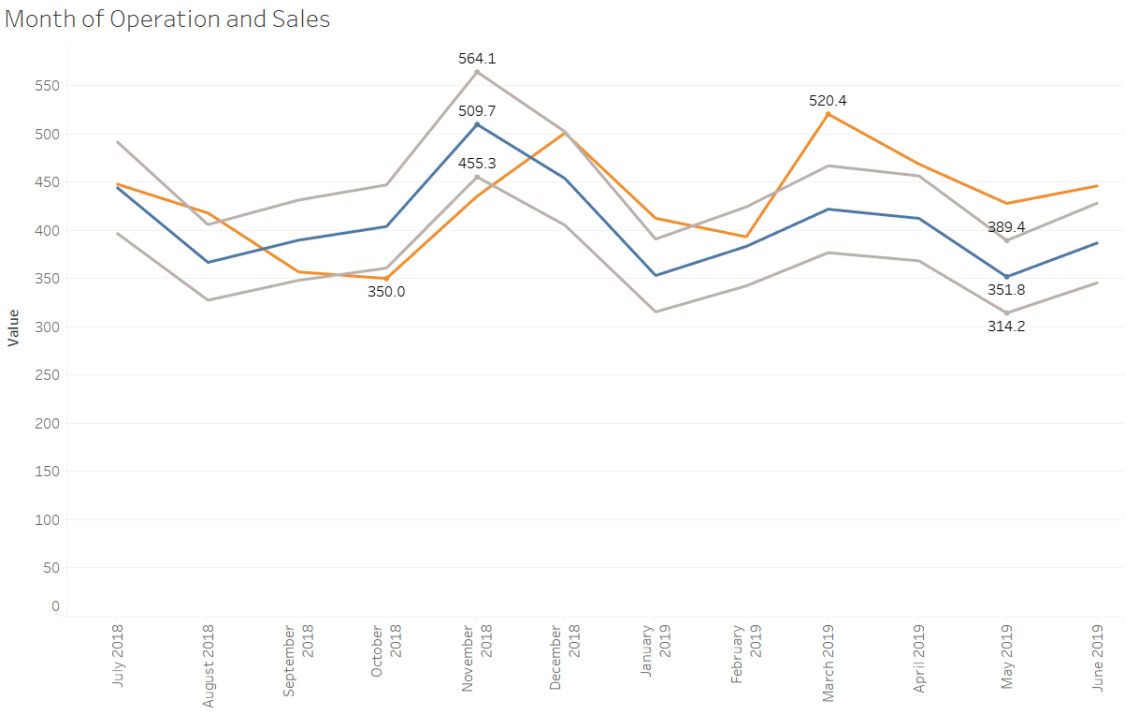
Total Sales

```
percent_off['Percentage_diff'] = abs(percent_off['Control Sales'] - percent_off['Trial Sales'])/percent_off['Control Sales']


std_dev = stdev(percent_off.loc[percent_off['MONTH_YEAR'] < '2019-02', 'Percentage_diff'])


percent_off['tvalue'] = percent_off['Percentage_diff']/std_dev
percent_off.loc[(percent_off['MONTH_YEAR'] > '2019-01') & (percent_off['MONTH_YEAR'] < '2019-05'), 'tvalue']
```

```
7    0.483476
8    4.377233
9    2.555802
Name: tvalue, dtype: float64
```

We calculate the T-Value for the Percentage difference for Sales

- The **standard deviation** comes out to be **0.053342**
- For **February 2019**, the **T-Value** is **0.483476**, **March 2019 – 4.377233**, **April 2019 – 2.555802**

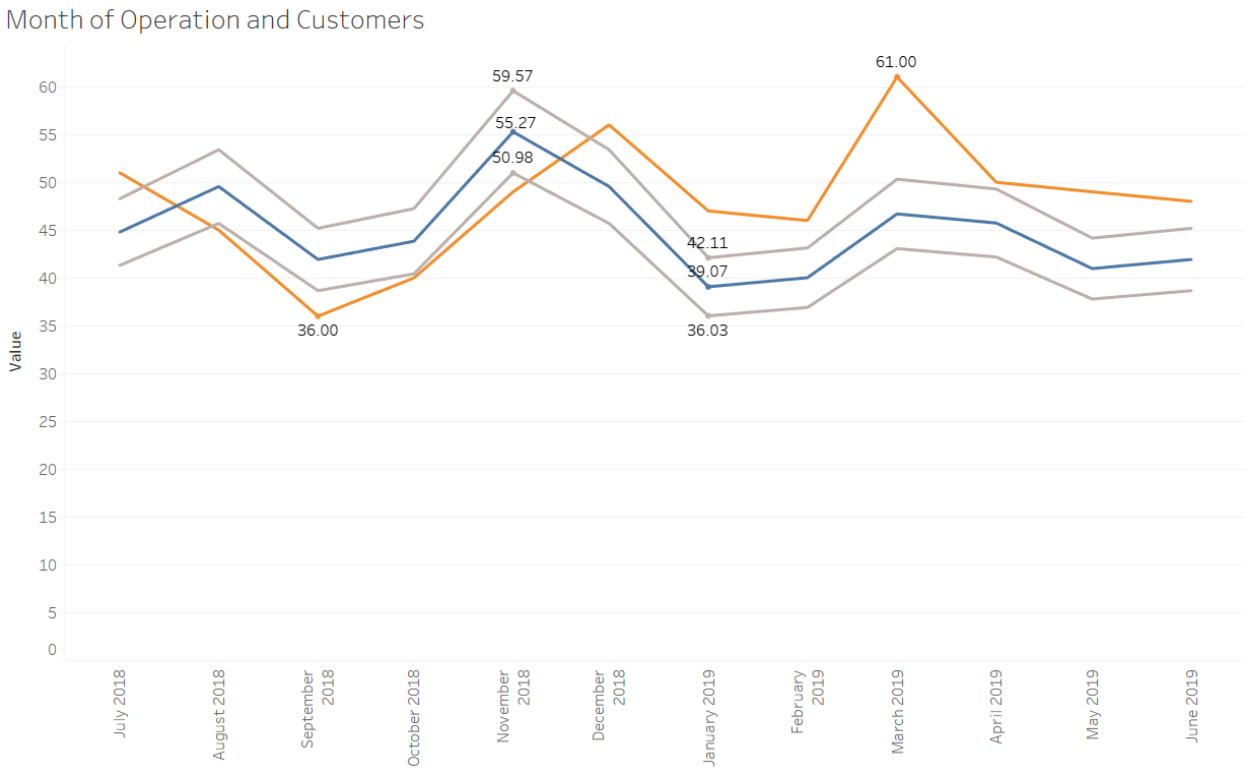


We can see the sales of Store 88 lies well under the confidence interval of sales of Store 237 in the pre-trial period. For the trial-period, the sales of store 88 show similar trends being outside the confidence interval of Store 237. The **sales go high in the month of March 2019 to 520.4** in the mid of the trial period.

Total Number of Customers

MONTH	Control Customers	Control 5% Confidence Interval	Control 95% Confidence Interval	Trial Customers
2018-07-01	44.788235	41.307576	48.268894	51
2018-08-01	49.552941	45.701999	53.403883	45
2018-09-01	41.929412	38.670922	45.187901	36
2018-10-01	43.835294	40.428692	47.241897	40
2018-11-01	55.270588	50.975307	59.565870	49
2018-12-01	49.552941	45.701999	53.403883	56
2019-01-01	39.070588	36.034269	42.106908	47
2019-02-01	40.023529	36.913153	43.133906	46
2019-03-01	46.694118	43.065345	50.322890	61
2019-04-01	45.741176	42.186461	49.295892	50
2019-05-01	40.976471	37.792038	44.160903	49
2019-06-01	41.929412	38.670922	45.187901	48

The **Standard deviation** is evaluated to be equal to **0.038856**.



We can observe that Store 88 shows similar trends to Store 237 regarding the number of customers coming to its store per month **except in the month of March 2019**, where the **customers are more than ever, 61** which can be the sole **cause of high sales** during the same month. **Store 88 most of the time is outside the confidence intervals** but closely correlated.

Conclusion

- We've found control stores **38, 155, 237** for trial stores **77, 86 and, 88** respectively.
- The results for trial stores 77 and 88 during the trial period show a significant difference in at least one of the three trial months but this is not the case for trial store 86. We can check with the client if the implementation of the trial was different in trial store 86 but overall, the trial shows a significant increase in sales.