

## Task 2-Experimentation and Uplift testing

Julia has asked us to evaluate the performance of a store trial which was performed in stores 77, 88.

This can be broken down by:

1. total sales revenue
2. total number of customers
3. average number of transactions per customer

Create a measure to compare different control stores to each of the trial stores to do this write a metric to reduce having to re-do the analysis for each trial store. Consider using Pearson correlations or a metric such as a magnitude distance e.g.  $1 - (\text{Observed distance} - \text{minimum distance}) / (\text{Maximum distance} - \text{minimum distance})$  as a measure.

Once you have selected your control stores, compare each trial and control pair during the trial period. You want to test if total sales are significantly different in the trial period and if so, check if the driver of change is more purchasing customers or more purchases per customers etc.

Main areas of Focus are :

1. Select control stores – Explore data, define metrics, visualize graphs
2. Assessment of the trial – insights/trends by comparing trial stores with control stores
3. Collate findings – summarize and provide recommendations

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
```

```
In [2]: qvi = pd.read_csv("/content/QVI_data.csv")
qvi.head()
```

```
Out[2]:
```

	LYLTY_CARD_NBR	DATE	STORE_NBR	TXN_ID	PROD_NBR	PROD_NAME	PRC
0	1000	2018-10-17	1	1	5	Natural Chip Compny SeaSalt175g	
1	1002	2018-09-16	1	2	58	Red Rock Deli Chikn&Garlic Aioli 150g	
2	1003	2019-03-07	1	3	52	Grain Waves Sour Cream&Chives 210G	
3	1003	2019-03-08	1	4	106	Natural ChipCo Hony Soy Chckn175g	
4	1004	2018-11-02	1	5	96	WW Original Stacked Chips 160g	

### Checking for nulls

```
In [3]: qvi.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264834 entries, 0 to 264833
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   LYLTY_CARD_NBR        264834 non-null int64
1   DATE                  264834 non-null object
2   STORE_NBR             264834 non-null int64
3   TXN_ID                264834 non-null int64
4   PROD_NBR              264834 non-null int64
5   PROD_NAME             264834 non-null object
6   PROD_QTY              264834 non-null int64
7   TOT_SALES             264834 non-null float64
8   PACK_SIZE             264834 non-null int64
9   BRAND                 264834 non-null object
10  LIFESTAGE             264834 non-null object
11  PREMIUM_CUSTOMER      264834 non-null object
dtypes: float64(1), int64(6), object(5)
memory usage: 24.2+ MB
```

```
In [4]: qvi["DATE"] = pd.to_datetime(qvi["DATE"])
qvi["YEARMONTH"] = qvi["DATE"].dt.strftime("%Y%m").astype("int")
```

Compile each store's monthly:

1. Total sales
2. Number of customers,
3. Average transactions per customer
4. Average chips per customer
5. Average price per unit

```
In [5]: def monthly_store_metrics():
        store_yrmo_group = qvi.groupby(["STORE_NBR", "YEARMONTH"])
        total = store_yrmo_group["TOT_SALES"].sum()
        num_cust = store_yrmo_group["LYLTY_CARD_NBR"].nunique()
        trans_per_cust = store_yrmo_group.size() / num_cust
        avg_chips_per_cust = store_yrmo_group["PROD_QTY"].sum() / num_cust
        avg_chips_price = total / store_yrmo_group["PROD_QTY"].sum()
        aggregates = [total, num_cust, trans_per_cust, avg_chips_per_cust, avg_chips_price]
        metrics = pd.concat(aggregates, axis=1)
        metrics.columns = ["TOT_SALES", "nCustomers", "nTxnPerCust", "nChipsPerTxn", "avgPricePerUnit"]
        return metrics
```

```
In [6]: qvi_monthly_metrics = monthly_store_metrics().reset_index()
        qvi_monthly_metrics.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3169 entries, 0 to 3168
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   STORE_NBR             3169 non-null  int64
1   YEARMONTH             3169 non-null  int64
2   TOT_SALES             3169 non-null  float64
3   nCustomers            3169 non-null  int64
4   nTxnPerCust           3169 non-null  float64
5   nChipsPerTxn          3169 non-null  float64
6   avgPricePerUnit       3169 non-null  float64
dtypes: float64(4), int64(3)
memory usage: 173.4 KB
```

**Pre-Trial Observation as this filter only stores with full 12 months observation**

```
In [7]: observ_counts = qvi_monthly_metrics["STORE_NBR"].value_counts()
        full_observ_index = observ_counts[observ_counts == 12].index
        full_observ = qvi_monthly_metrics[qvi_monthly_metrics["STORE_NBR"].isin(full_observ_index)]
        pretrial_full_observ = full_observ[full_observ["YEARMONTH"] < 201902]

        pretrial_full_observ.head(8)
```

```
Out[7]:
```

	STORE_NBR	YEARMONTH	TOT_SALES	nCustomers	nTxnPerCust	nChipsPerTxn	avgPricePerUnit
0	1	201807	206.9	49	1.061224	1.265306	1.061224
1	1	201808	176.1	42	1.023810	1.285714	1.023810
2	1	201809	278.8	59	1.050847	1.271186	1.050847
3	1	201810	188.1	44	1.022727	1.318182	1.022727
4	1	201811	192.6	46	1.021739	1.239130	1.021739
5	1	201812	189.6	42	1.119048	1.357143	1.119048
6	1	201901	154.8	35	1.028571	1.200000	1.028571
12	2	201807	150.8	39	1.051282	1.179487	1.051282

```
In [11]: def calcCorrTable(metricCol, storeComparison, inputTable=pretrial_full_ob:
control_store_nbrs = inputTable[~inputTable["STORE_NBR"].isin([77, 86, 88])
corrs = pd.DataFrame(columns = ["YEARMONTH", "Trial_Str", "Ctrl_Str", "Corr_Score"])
trial_store = inputTable[inputTable["STORE_NBR"] == storeComparison][metricCol]
for control in control_store_nbrs:
    concat_df = pd.DataFrame(columns = ["YEARMONTH", "Trial_Str", "Ctrl_Str", "Corr_Score"])
    control_store = inputTable[inputTable["STORE_NBR"] == control][metricCol]
    concat_df["Corr_Score"] = trial_store.corrwith(control_store, axis=1)
    concat_df["Trial_Str"] = storeComparison
    concat_df["Ctrl_Str"] = control
    concat_df["YEARMONTH"] = list(inputTable[inputTable["STORE_NBR"] == storeComparison]["YEARMONTH"])
    corrs = pd.concat([corrs, concat_df])
return corrs
```

```
In [12]: corr_table = pd.DataFrame()
for trial_num in [77, 86, 88]:
    corr_table = pd.concat([corr_table, calcCorrTable(["TOT_SALES", "nCusts"], trial_num, inputTable)])

corr_table.head(8)
```

```
Out[12]:
```

	YEARMONTH	Trial_Str	Ctrl_Str	Corr_Score
0	201807	77	1	0.070414
1	201808	77	1	0.027276
2	201809	77	1	0.002389
3	201810	77	1	-0.020045
4	201811	77	1	0.030024
5	201812	77	1	0.063946
6	201901	77	1	0.001470
0	201807	77	2	0.142957

```
In [13]: def calculateMagnitudeDistance(metricCol, storeComparison, inputTable=pretrial_full_ob):
control_store_nbrs = inputTable[~inputTable["STORE_NBR"].isin([77, 86, 88])]
dists = pd.DataFrame()
trial_store = inputTable[inputTable["STORE_NBR"] == storeComparison][metricCol]
for control in control_store_nbrs:
    concat_df = abs(inputTable[inputTable["STORE_NBR"] == control][metricCol] - trial_store)
    concat_df["YEARMONTH"] = list(inputTable[inputTable["STORE_NBR"] == control]["YEARMONTH"])
    concat_df["Trial_Str"] = storeComparison
    concat_df["Ctrl_Str"] = control
    dists = pd.concat([dists, concat_df])
for col in metricCol:
    dists[col] = 1 - ((dists[col] - dists[col].min()) / (dists[col].max() - dists[col].min()))
    dists["magnitude"] = dists[metricCol].mean(axis=1)
return dists
```

```
In [14]: dist_table = pd.DataFrame()
for trial_num in [77, 86, 88]:
    dist_table = pd.concat([dist_table, calculateMagnitudeDistance(["TOT_SALES", "nCusts"], trial_num, inputTable)])

dist_table.head(8)
dist_table
```

```
Out[14]:
```

	TOT_SALES	nCustomers	nTxnPerCust	nChipsPerTxn	avgPricePerUnit	YEARMONTH
0	0.935431	0.980769	0.958035	0.739412	0.883569	201807
1	0.942972	0.951923	0.993823	0.802894	0.886328	201808
2	0.961503	0.836538	0.992126	0.730041	0.703027	201809
3	0.988221	0.932692	0.989514	0.940460	0.590528	201810
4	0.962149	0.951923	0.874566	0.730358	0.832481	201811
...	...	...	...	...	...	...
2	0.207554	0.286822	0.462846	0.779879	0.923887	201809
3	0.346797	0.387597	0.571497	0.796875	0.971133	201810
4	0.286706	0.310078	0.623883	0.813241	0.966999	201811
5	0.347151	0.387597	0.376456	0.699748	0.962198	201812
6	0.402353	0.449612	0.450378	0.739714	0.971335	201901

5397 rows × 9 columns

We'll select control stores based on how similar monthly total sales in dollar amounts and monthly number of customers are to the trial stores by using correlation and magnitude distance.

```
In [15]: def combine_corr_dist(metricCol, storeComparison, inputTable=pretrial_full):
    corrs = calcCorrTable(metricCol, storeComparison, inputTable)
    dists = calculateMagnitudeDistance(metricCol, storeComparison, inputTable)
    dists = dists.drop(metricCol, axis=1)
    combine = pd.merge(corrs, dists, on=["YEARMONTH", "Trial_Str", "Ctrl_Str"])
    return combine
```

```
In [16]: compare_metrics_table1 = pd.DataFrame()
for trial_num in [77, 86, 88]:
    compare_metrics_table1 = pd.concat([compare_metrics_table1, combine_corr_dist(trial_num, "Trial_Str", "Ctrl_Str")])
```

```
In [17]: corr_weight = 0.5
dist_weight = 1 - corr_weight
```

Determining the top five highest composite score for each trial based on Total sales

```
In [18]: grouped_comparison_table1 = compare_metrics_table1.groupby(["Trial_Str", "Ctrl_Str"])
grouped_comparison_table1["CompScore"] = (corr_weight * grouped_comparison_table1["Corr"]) + (dist_weight * grouped_comparison_table1["Dist"])
for trial_num in compare_metrics_table1["Trial_Str"].unique():
    print(grouped_comparison_table1[grouped_comparison_table1["Trial_Str"] == trial_num].sort_values("CompScore", ascending=False).head(5))
```

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
218	77	233	1.0	0.986477	0.993238
239	77	255	1.0	0.979479	0.989739
177	77	188	1.0	0.977663	0.988831
49	77	53	1.0	0.976678	0.988339
120	77	131	1.0	0.976267	0.988134

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
356	86	109	1.0	0.966783	0.983391
401	86	155	1.0	0.965876	0.982938
464	86	222	1.0	0.962280	0.981140
467	86	225	1.0	0.960512	0.980256
471	86	229	1.0	0.951704	0.975852

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
551	88	40	1.0	0.941165	0.970582
538	88	26	1.0	0.904377	0.952189
582	88	72	1.0	0.903800	0.951900
517	88	4	1.0	0.903466	0.951733
568	88	58	1.0	0.891678	0.945839

```
In [19]: compare_metrics_table2 = pd.DataFrame()
for trial_num in [77, 86, 88]:
    compare_metrics_table2 = pd.concat([compare_metrics_table2, combine_c
```

Determining the top five highest composite score for each trial based on no. of customers

```
In [20]: grouped_comparison_table2 = compare_metrics_table2.groupby(["Trial_Str", '
grouped_comparison_table2["CompScore"] = (corr_weight * grouped_comparison
for trial_num in compare_metrics_table2["Trial_Str"].unique():
    print(grouped_comparison_table2[grouped_comparison_table2["Trial_Str"]
```

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
218	77	233	1.0	0.993132	0.996566
38	77	41	1.0	0.976648	0.988324
101	77	111	1.0	0.968407	0.984203
105	77	115	1.0	0.967033	0.983516
15	77	17	1.0	0.965659	0.982830

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
401	86	155	1.0	0.986772	0.993386
467	86	225	1.0	0.969577	0.984788
356	86	109	1.0	0.969577	0.984788
471	86	229	1.0	0.964286	0.982143
293	86	39	1.0	0.961640	0.980820

	Trial_Str	Ctrl_Str	Corr_Score	magnitude	CompScore
736	88	237	1.0	0.987818	0.993909
705	88	203	1.0	0.944629	0.972315
551	88	40	1.0	0.942414	0.971207
668	88	165	1.0	0.935770	0.967885
701	88	199	1.0	0.932447	0.966224

```
In [21]: for trial_num in compare_metrics_table2["Trial_Str"].unique():
a = grouped_comparison_table1[grouped_comparison_table1["Trial_Str"] :
b = grouped_comparison_table2[grouped_comparison_table2["Trial_Str"] :
print((pd.concat([a,b], axis=1).sum(axis=1)/2).sort_values(ascending=)
```

Trial_Str	Ctrl_Str	
77	233	0.994902
	41	0.986020
	46	0.984762

dtype: float64

Trial_Str	Ctrl_Str	
86	155	0.988162
	109	0.984090
	225	0.982522

dtype: float64

Trial_Str	Ctrl_Str	
88	40	0.970895
	26	0.958929
	72	0.954079

dtype: float64

Similarities based on total sales:

1. Trial store 77: Store 233, 255, 188
2. Trial store 86: Store 109, 155, 222
3. Trial store 88: Store 40, 26, 72

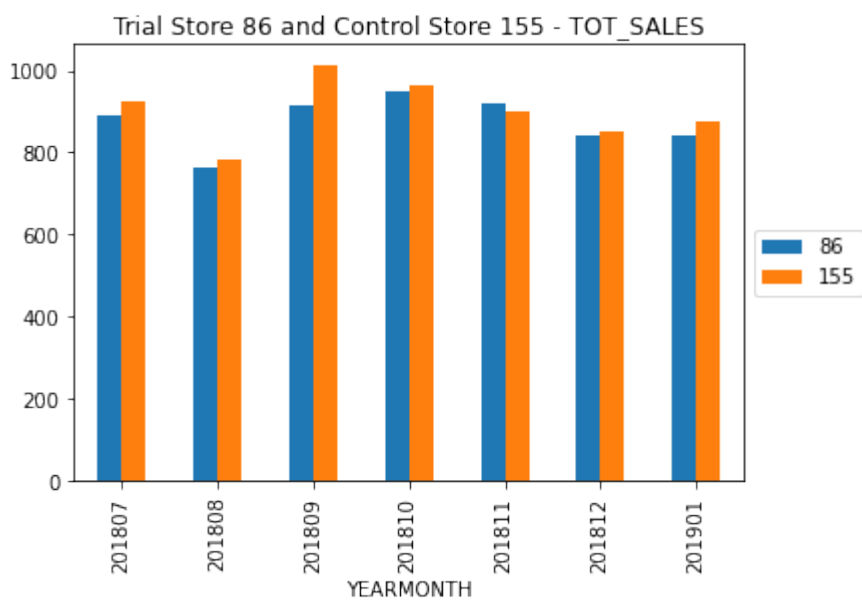
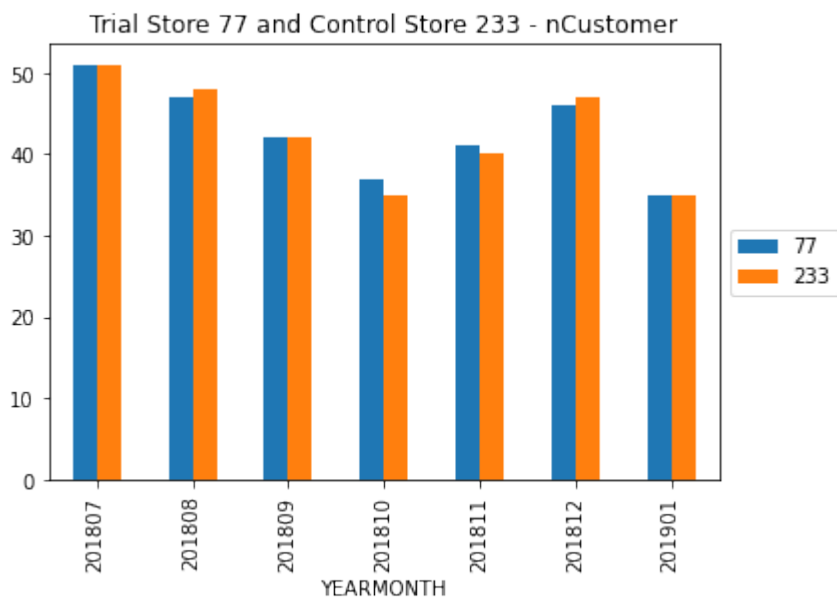
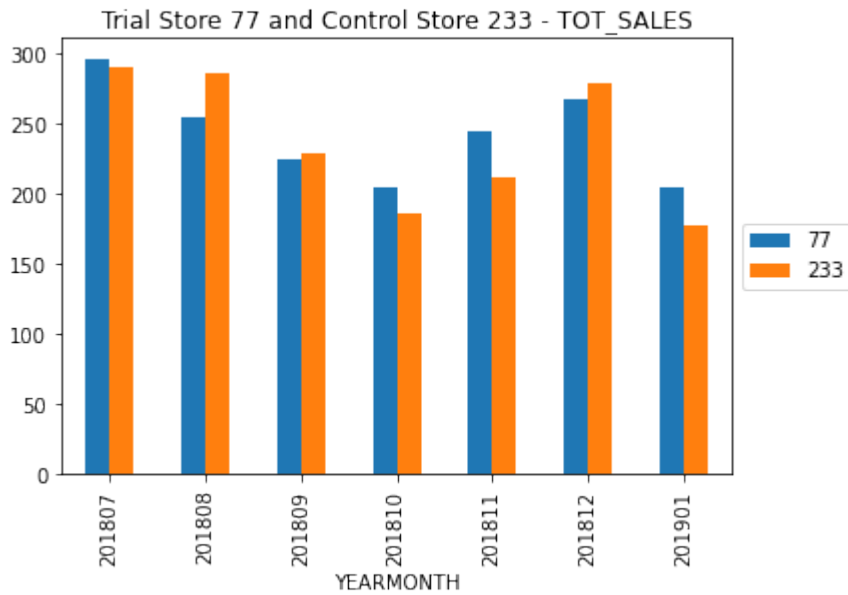
Similarities based on No. of Customers:

1. Trial store 77: Store 233, 41, 111
2. Trial store 86: Store 155, 225, 109
3. Trial store 88: Store 237, 203, 40

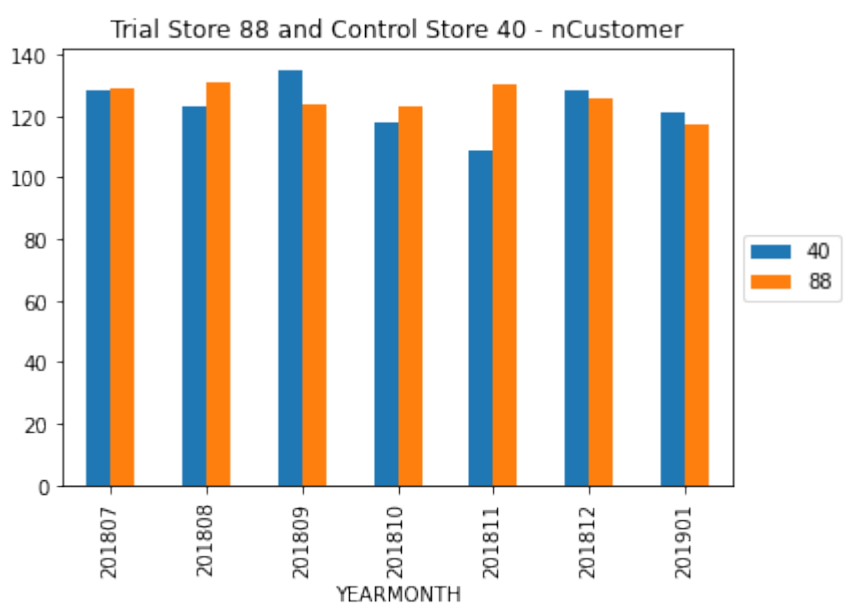
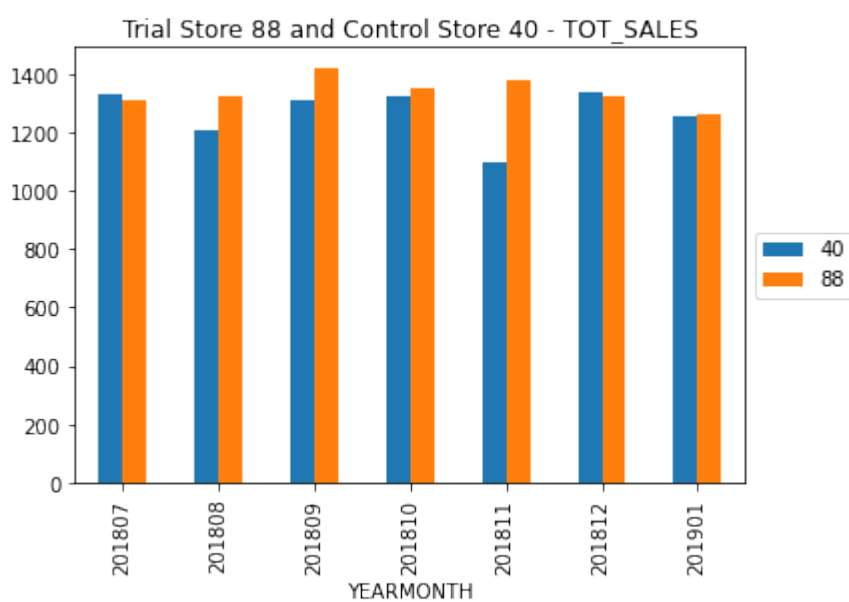
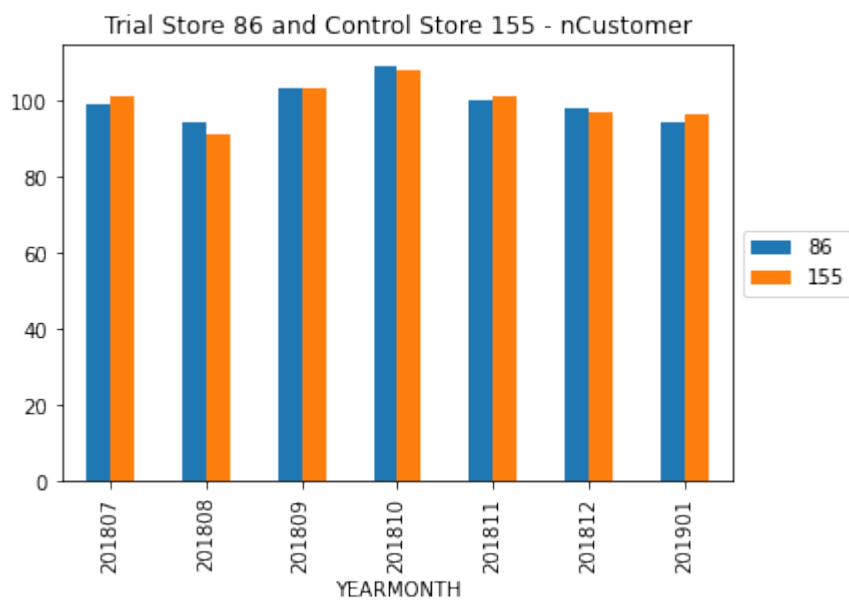
Final Similarities based on Highest average of both features combined:

1. Trial store 77: Store 233
2. Trial store 86: Store 155
3. Trial store 88: Store 40

```
In [22]: trial_control_dic = {77:233, 86:155, 88:40}
for key, val in trial_control_dic.items():
    pretrial_full_observ[pretrial_full_observ["STORE_NBR"].isin([key, val]
        ["YEARMONTH", "STORE_NBR"]).sum()["TOT_SALES"].unstack().plot.bar()
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(key)+" and Control Store "+str(val)+" - Sales")
    plt.show()
    pretrial_full_observ[pretrial_full_observ["STORE_NBR"].isin([key, val]
        ["YEARMONTH", "STORE_NBR"]).sum()["nCustomers"].unstack().plot.bar()
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(key)+" and Control Store "+str(val)+" - Customers")
    plt.show()
    print('\n')
```







Next we'll compare the performance of Trial stores to Control stores during the trial period. To ensure their performance is comparable during Trial period, we need to scale (multiply to ratio of trial / control) of Control stores' performance to Trial store's performance during pre-trial. Starting with TOT\_SA

```
In [23]: #Ratio of Store 77 and its Control store.
sales_ratio_77 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 233] / pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 155]["TOT_SALES"]

#Ratio of Store 86 and its Control store.
sales_ratio_86 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 155] / pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 40]["TOT_SALES"]

#Ratio of Store 77 and its Control store.
sales_ratio_88 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 40] / pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == 233]["TOT_SALES"]

In [25]: trial_full_observ = full_observ[(full_observ["YEARMONTH"] >= 201902) & (full_observ["STORE_NBR"] != trial_store_nbr)]
scaled_sales_control_stores = full_observ[full_observ["STORE_NBR"].isin(control_store_nbrs)]

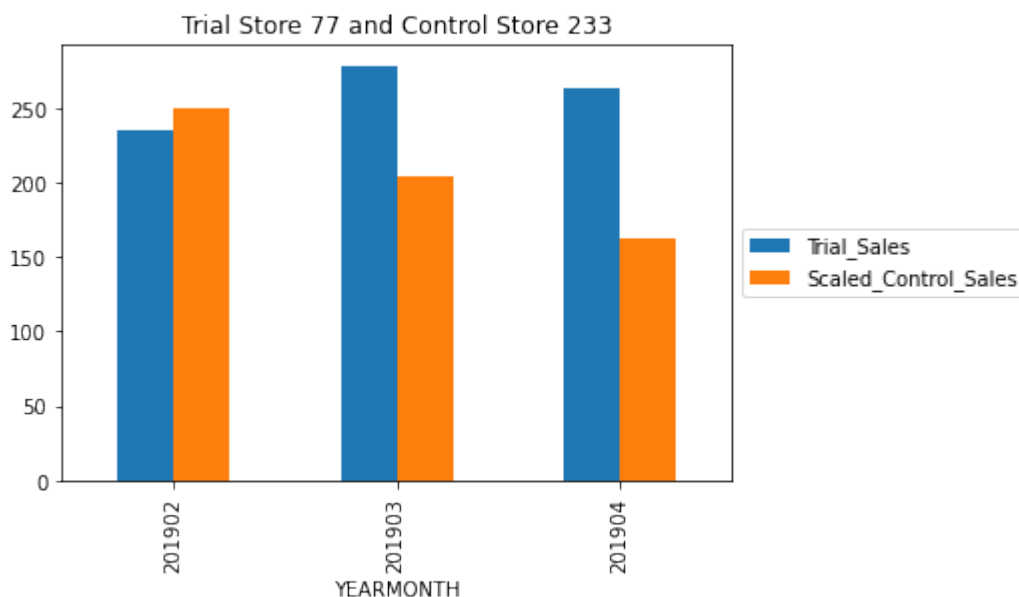
def scaler(row):
    if row["STORE_NBR"] == 233:
        return row["TOT_SALES"] * sales_ratio_77
    elif row["STORE_NBR"] == 155:
        return row["TOT_SALES"] * sales_ratio_86
    elif row["STORE_NBR"] == 40:
        return row["TOT_SALES"] * sales_ratio_88

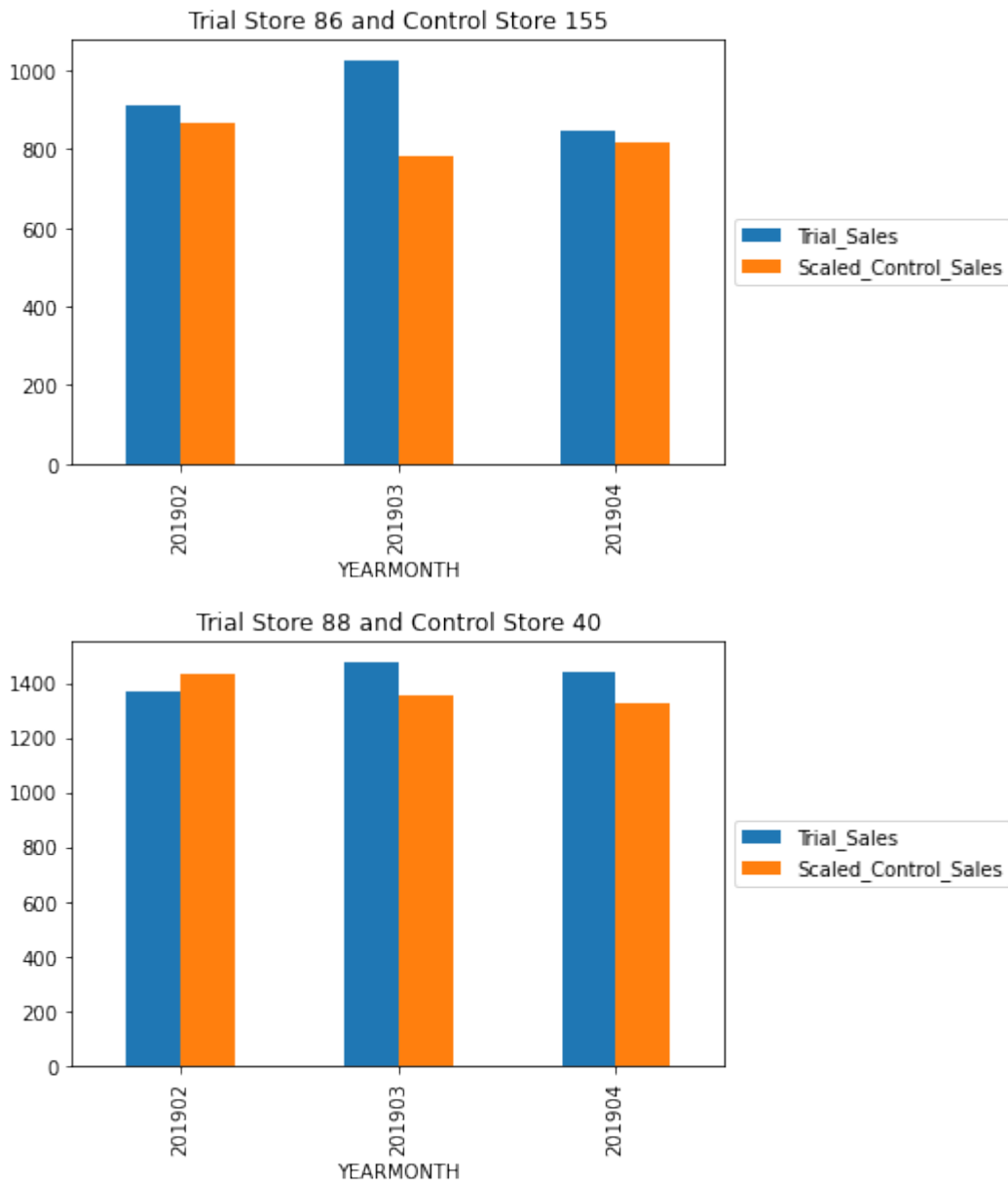
scaled_sales_control_stores["ScaledSales"] = scaled_sales_control_stores["TOT_SALES"] * scaler(scaled_sales_control_stores)

trial_scaled_sales_control_stores = scaled_sales_control_stores[(scaled_sales_control_stores["STORE_NBR"] == trial_store_nbr)]
pretrial_scaled_sales_control_stores = scaled_sales_control_stores[scaled_sales_control_stores["YEARMONTH"] < 201902]

percentage_diff = {}

for trial, control in trial_control_dic.items():
    a = trial_scaled_sales_control_stores[trial_scaled_sales_control_stores["STORE_NBR"] == trial]
    b = trial_full_observ[trial_full_observ["STORE_NBR"] == trial]
    percentage_diff[trial] = b["TOT_SALES"].sum() / a["ScaledSales"].sum()
    b[["YEARMONTH", "TOT_SALES"]].merge(a[["YEARMONTH", "ScaledSales"]], on="YEARMONTH", left=True, right=False)
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(trial)+" and Control Store "+str(control))
```





In [26]: percentage\_diff

Out[26]: {77: 1.2615468650086274, 86: 1.13150143573637, 88: 1.0434583458542188}

```
In [27]: temp1 = scaled_sales_control_stores.sort_values(by=["STORE_NBR", "YEARMON"]
temp2 = full_observ[full_observ["STORE_NBR"].isin([77,86,88])][["STORE_NBR", "YEARMON"]
scaledsales_vs_trial = pd.concat([temp1, temp2], axis=1)
scaledsales_vs_trial.columns = ["c_STORE_NBR", "YEARMONTH", "c_ScaledSales", "Sales_Percentage_Diff"]
scaledsales_vs_trial["Sales_Percentage_Diff"] = (scaledsales_vs_trial["c_ScaledSales"] - scaledsales_vs_trial["Sales_Percentage_Diff"]) / scaledsales_vs_trial["c_ScaledSales"]
def label_period(cell):
    if cell < 201902:
        return "pre"
    elif cell > 201904:
        return "post"
    else:
        return "trial"
scaledsales_vs_trial["trial_period"] = scaledsales_vs_trial["YEARMONTH"].apply(label_period)
scaledsales_vs_trial[scaledsales_vs_trial["trial_period"] == "trial"]
```

```
Out[27]:
```

	c_STORE_NBR	YEARMONTH	c_ScaledSales	t_STORE_NBR	t_TOT_SALES	Sales_Pe
7	233	201902	249.762622	77	235.0	
8	233	201903	203.802205	77	278.5	
9	233	201904	162.345704	77	263.5	
19	155	201902	864.522060	86	913.2	
20	155	201903	780.320405	86	1026.8	
21	155	201904	819.317024	86	848.2	
31	40	201902	1434.399269	88	1370.2	
32	40	201903	1352.064709	88	1477.2	
33	40	201904	1321.797762	88	1439.4	

Check significance of Trial minus Control stores TOT\_SALES Percentage Difference Pre-Trial vs

Step 1: Check null hypothesis of 0 difference between control store's Pre-Trial and Trial period performance.

Step 2: Proof control and trial stores are similar statistically

Check p-value of control store's Pre-Trial vs Trial store's Pre-Trial. If <5%, it is significantly different >5%, it is not significantly different (similar).

Step 3: After checking Null Hypothesis of first 2 step to be true, we can check Null Hypothesis of Percentage Difference between Trial and Control stores during pre-trial is the same as during trial

Check T-Value of Percentage Difference of each Trial month (Feb, March, April 2019). Mean is mean Percentage Difference during pre-trial. Standard deviation is stdev of Percentage Difference during trial. Formula is Trial month's Percentage Difference minus Mean, divided by Standard deviation. Compare each T-Value with 95% percentage significance critical t-value of 6 degrees of freedom (months of sample - 1)

```
In [28]: from scipy.stats import ttest_ind, t

# Step 1
for num in [40, 155, 233]:
    print("Store", num)
    print(ttest_ind(pretrial_scaled_sales_control_stores[pretrial_scaled_sales_control_stores.index(num)],
                    trial_scaled_sales_control_stores[trial_scaled_sales_control_stores.index(num)],
                    equal_var=False), '\n')
    #print(len(pretrial_scaled_sales_control_stores[pretrial_scaled_sales_control_stores.index(num)]))

alpha = 0.05
print("Critical t-value for 95% confidence interval:")
print(t.ppf((alpha/2, 1-alpha/2), df=min([len(pretrial_scaled_sales_control_stores), len(trial_scaled_sales_control_stores[trial_scaled_sales_control_stores.index(num)])])))
```

Store 40

Ttest\_indResult(statistic=-0.5958372343168585, pvalue=0.5722861621434009)

Store 155

Ttest\_indResult(statistic=1.429195687929098, pvalue=0.19727058651603258)

Store 233

Ttest\_indResult(statistic=1.1911026010974504, pvalue=0.29445006064862156)

Critical t-value for 95% confidence interval:

[-4.30265273 4.30265273]

```
In [29]: a = pretrial_scaled_sales_control_stores[pretrial_scaled_sales_control_stores["STORE_NBR"] == 40]
b = trial_scaled_sales_control_stores[trial_scaled_sales_control_stores["STORE_NBR"] == 40]
```

Null hypothesis is true. There isn't any statistically significant difference between control store's s  
Pre-Trial and Trial period sales.

```
In [30]: # Step 2
```

```
for trial, cont in trial_control_dic.items():
    print("Trial store:", trial, ", Control store:", cont)
    print(ttest_ind(pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == trial],
                    pretrial_scaled_sales_control_stores[pretrial_scaled_sales_control_stores["STORE_NBR"] == cont],
                    equal_var=True), '\n')
    #print(len(pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == trial]) == len(pretrial_scaled_sales_control_stores[pretrial_scaled_sales_control_stores["STORE_NBR"] == cont]))
```

```
alpha = 0.05
```

```
print("Critical t-value for 95% confidence interval:")
```

```
print(t.ppf((alpha/2, 1-alpha/2), df=len(pretrial_full_observ[pretrial_full_observ["STORE_NBR"] == trial])))
```

Trial store: 77 , Control store: 233

Ttest\_indResult(statistic=-1.2533353315065926e-15, pvalue=0.9999999999999999)

Trial store: 86 , Control store: 155

Ttest\_indResult(statistic=0.0, pvalue=1.0)

Trial store: 88 , Control store: 40

Ttest\_indResult(statistic=0.0, pvalue=1.0)

Critical t-value for 95% confidence interval:

[-2.44691185 2.44691185]

Null hypothesis is true. There isn't any statistically significant difference between Trial store's sale  
Control store's scaled-sales performance during pre-trial.

In [31]: # Step 3

```
for trial, cont in trial_control_dic.items():
    print("Trial store:", trial, ", Control store:", cont)
    temp_pre = scaledsales_vs_trial[(scaledsales_vs_trial["c_STORE_NBR"] :
    std = temp_pre["Sales_Percentage_Diff"].std()
    mean = temp_pre["Sales_Percentage_Diff"].mean()
    #print(std, mean)
    for t_month in scaledsales_vs_trial[scaledsales_vs_trial["trial_perio
        pdif = scaledsales_vs_trial[(scaledsales_vs_trial["YEARMONTH"] ==
        print(t_month, ":", (float(pdif)-mean)/std)
    print('\n')

print("Critical t-value for 95% confidence interval:")
conf_intv_95 = t.ppf(0.95, df=len(temp_pre)-1)
print(conf_intv_95)
```

```
Trial store: 77 , Control store: 233
201902 : -0.7171038288055888
201903 : 3.035317928855662
201904 : 4.708944418758203
```

```
Trial store: 86 , Control store: 155
201902 : 1.4133618775921797
201903 : 7.123063846042149
201904 : 0.8863824572944162
```

```
Trial store: 88 , Control store: 40
201902 : -0.5481633746817604
201903 : 1.0089992743637755
201904 : 0.9710006270463645
```

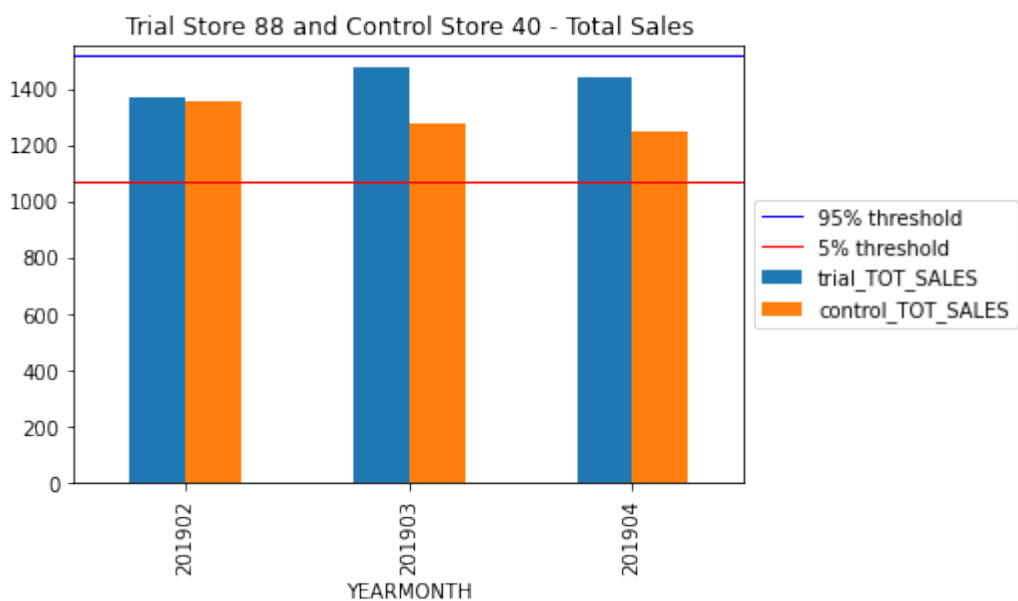
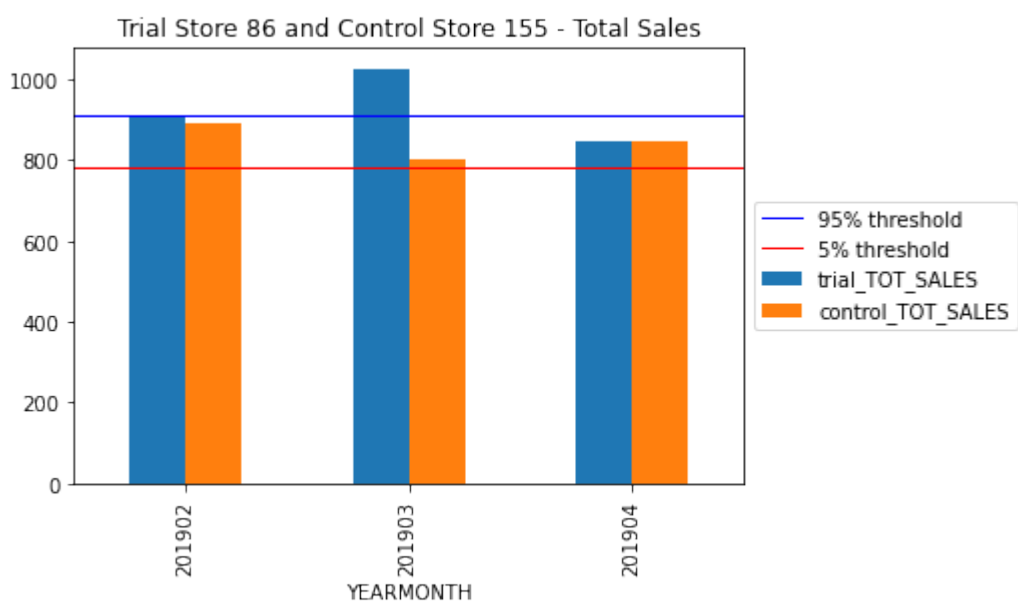
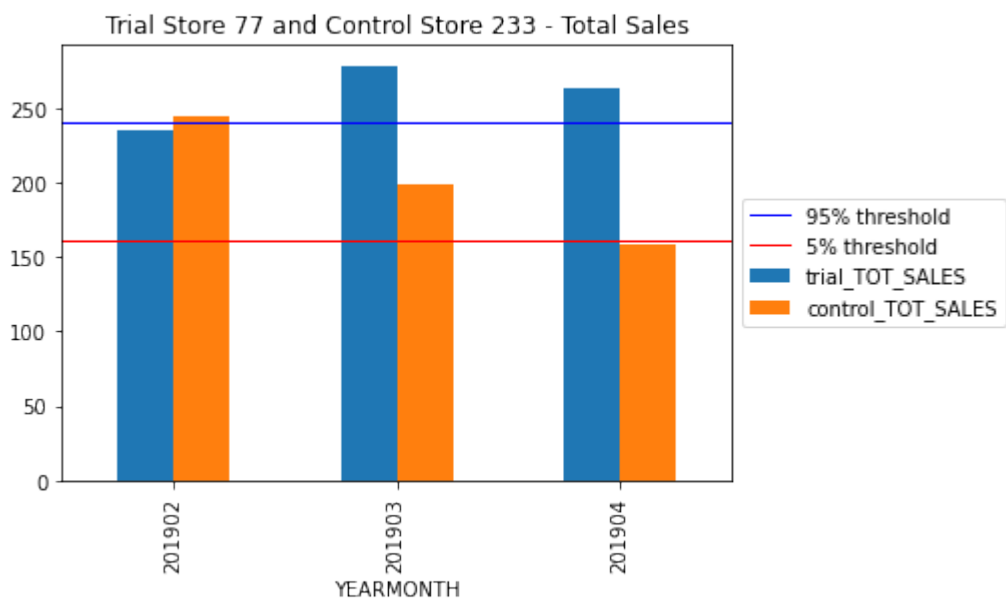
```
Critical t-value for 95% confidence interval:
1.9431802803927816
```

There are 3 months' increase in performance that are statistically significant (Above the 95% confidence interval t-score):

March and April trial months for trial store 77

March trial months for trial store 86

```
In [32]: for trial, control in trial_control_dic.items():
    a = trial_scaled_sales_control_stores[trial_scaled_sales_control_store
    b = trial_full_observ[trial_full_observ["STORE_NBR"] == trial][["STORE
    comb = b[["YEARMONTH", "trial_TOT_SALES"]].merge(a[["YEARMONTH", "control
    comb.plot.bar()
    cont_sc_sales = trial_scaled_sales_control_stores[trial_scaled_sales_store
    std = scaledsales_vs_trial[(scaledsales_vs_trial["c_STORE_NBR"] == control
    thresh95 = cont_sc_sales.mean() + (cont_sc_sales.mean() * std * 2)
    thresh5 = cont_sc_sales.mean() - (cont_sc_sales.mean() * std * 2)
    plt.axhline(y=thresh95, linewidth=1, color='b', label="95% threshold")
    plt.axhline(y=thresh5, linewidth=1, color='r', label="5% threshold")
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(trial)+" and Control Store "+str(control))
    plt.savefig("TS {} and CS {} - TOT_SALES.png".format(trial, control), dpi=100)
```



```

In [33]: #Ratio of Store 77 and its Control store.
ncust_ratio_77 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] ==

#Ratio of Store 86 and its Control store.
ncust_ratio_86 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] ==

#Ratio of Store 77 and its Control store.
ncust_ratio_88 = pretrial_full_observ[pretrial_full_observ["STORE_NBR"] ==

In [34]: #trial_full_observ = full_observ[(full_observ["YEARMONTH"] >= 201902) & (
scaled_ncust_control_stores = full_observ[full_observ["STORE_NBR"].isin([

def scaler_c(row):
    if row["STORE_NBR"] == 233:
        return row["nCustomers"] * ncust_ratio_77
    elif row["STORE_NBR"] == 155:
        return row["nCustomers"] * ncust_ratio_86
    elif row["STORE_NBR"] == 40:
        return row["nCustomers"] * ncust_ratio_88

scaled_ncust_control_stores["ScaledNcust"] = scaled_ncust_control_stores.

trial_scaled_ncust_control_stores = scaled_ncust_control_stores[(scaled_n
pretrial_scaled_ncust_control_stores = scaled_ncust_control_stores[scaled

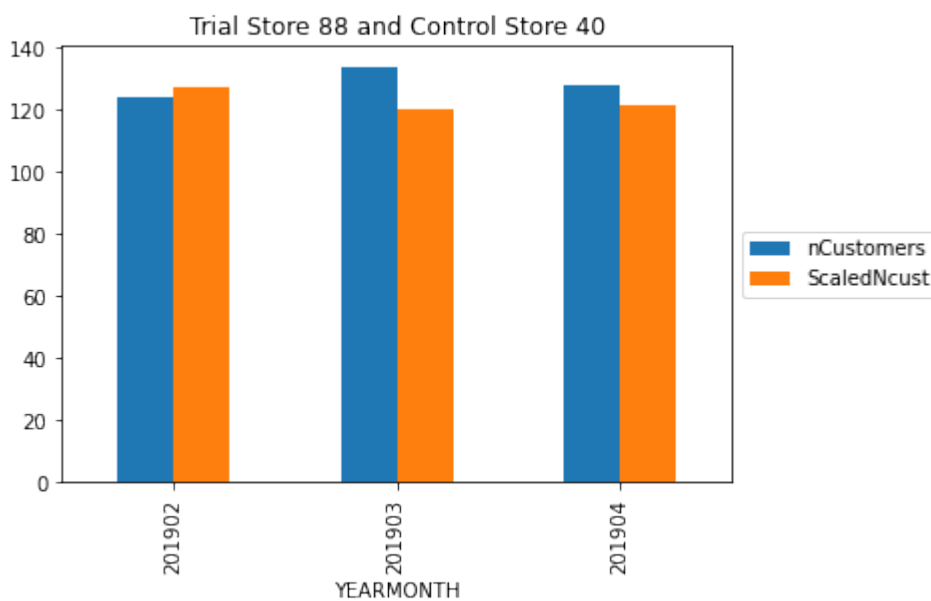
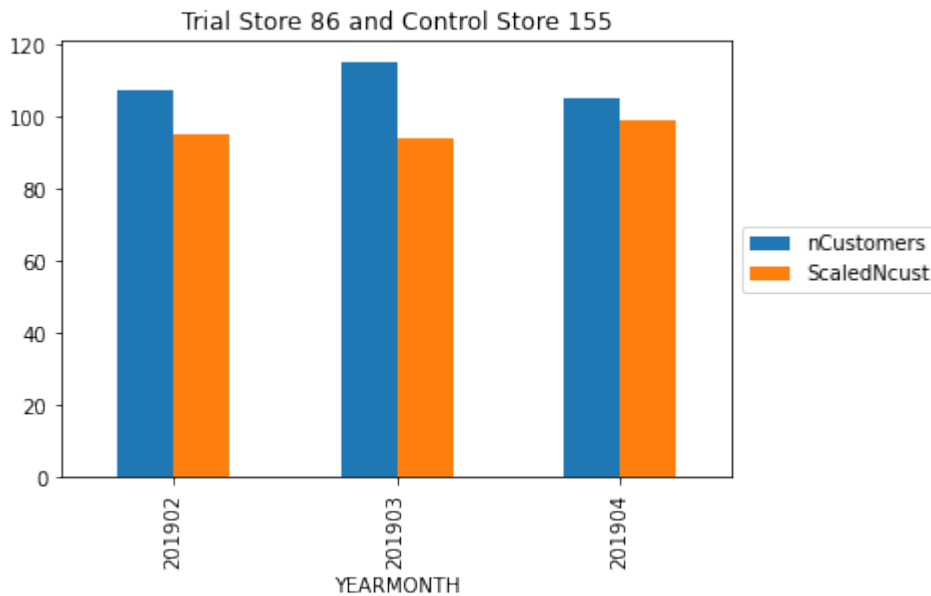
ncust_percentage_diff = {}

for trial, control in trial_control_dic.items():
    a = trial_scaled_ncust_control_stores[trial_scaled_ncust_control_store
    b = trial_full_observ[trial_full_observ["STORE_NBR"] == trial][["STORE
    ncust_percentage_diff[trial] = b["nCustomers"].sum() / a["ScaledNcust"]
    b[["YEARMONTH", "nCustomers"]].merge(a[["YEARMONTH", "ScaledNcust"]],
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(trial)+" and Control Store "+str(control)

```







```
In [35]: ncust_percentage_diff
```

```
Out[35]: {77: 1.2306529009742622, 86: 1.1354166666666667, 88: 1.0444876946258161}
```

```
In [36]: temp1 = scaled_ncust_control_stores.sort_values(by=["STORE_NBR", "YEARMON"]
temp2 = full_observ[full_observ["STORE_NBR"].isin([77,86,88])][["STORE_NBR", "YEARMON"]
scaledncust_vs_trial = pd.concat([temp1, temp2], axis=1)
scaledncust_vs_trial.columns = ["c_STORE_NBR", "YEARMONTH", "c_ScaledNcust", "YEARMON"]
scaledncust_vs_trial["nCust_Percentage_Diff"] = (scaledncust_vs_trial["t_nCust"] /
scaledncust_vs_trial["c_nCust"] - 1) * 100
scaledncust_vs_trial["trial_period"] = scaledncust_vs_trial["YEARMONTH"].dt.strftime("%Y-%m")
scaledncust_vs_trial[scaledncust_vs_trial["trial_period"] == "trial"]
```

```
Out[36]:
```

	c_STORE_NBR	YEARMONTH	c_ScaledNcust	t_STORE_NBR	t_nCustomers	nCust_P
7	233	201902	45.151007	77	45	
8	233	201903	40.134228	77	50	
9	233	201904	30.100671	77	47	
19	155	201902	95.000000	86	107	
20	155	201903	94.000000	86	115	
21	155	201904	99.000000	86	105	
31	40	201902	127.610209	88	124	
32	40	201903	120.464037	88	134	
33	40	201904	121.484919	88	128	

Check significance of Trial minus Control stores nCustomers Percentage Difference Pre-Trial vs

Step 1: Check null hypothesis of 0 difference between control store's Pre-Trial and Trial period performance.

Step 2: Proof control and trial stores are similar statistically

Step 3: After checking Null Hypothesis of first 2 step to be true, we can check Null Hypothesis of Percentage Difference between Trial and Control stores during pre-trial is the same as during trial

```
In [37]: # Step 1
for num in [40, 155, 233]:
    print("Store", num)
    print(ttest_ind(pretrial_scaled_ncust_control_stores[pretrial_scaled_ncust_control_stores.index(num)],
                    trial_scaled_ncust_control_stores[trial_scaled_ncust_control_stores.index(num)],
                    equal_var=False), '\n')

alpha = 0.05
print("Critical t-value for 95% confidence interval:")
print(t.ppf((alpha/2, 1-alpha/2), df=min([len(pretrial_scaled_ncust_control_stores), len(trial_scaled_ncust_control_stores[trial_scaled_ncust_control_stores.index(num)])]))

Store 40
Ttest_indResult(statistic=0.644732693420032, pvalue=0.5376573016017127)

Store 155
Ttest_indResult(statistic=1.3888888888888882, pvalue=0.204345986327886)

Store 233
Ttest_indResult(statistic=0.8442563765225701, pvalue=0.4559280037660254)

Critical t-value for 95% confidence interval:
[-4.30265273  4.30265273]
```

```
In [38]: # Step 2
for trial, cont in trial_control_dic.items():
    print("Trial store:", trial, ", Control store:", cont)
    print(ttest_ind(pretrial_full_observ[pretrial_full_observ["STORE_NBR"]
        pretrial_scaled_ncust_control_stores[pretrial_scaled_ncust_control_stores["STORE_NBR"]
        equal_var=True), '\n')

alpha = 0.05
print("Critical t-value for 95% confidence interval:")
print(t.ppf((alpha/2, 1-alpha/2), df=len(pretrial_full_observ[pretrial_full_observ["STORE_NBR"]

Trial store: 77 , Control store: 233
Ttest_indResult(statistic=0.0, pvalue=1.0)

Trial store: 86 , Control store: 155
Ttest_indResult(statistic=0.0, pvalue=1.0)

Trial store: 88 , Control store: 40
Ttest_indResult(statistic=-7.648483953264653e-15, pvalue=0.9999999999999994)

Critical t-value for 95% confidence interval:
[-2.44691185  2.44691185]
```

```
In [39]: # Step 3
for trial, cont in trial_control_dic.items():
    print("Trial store:", trial, ", Control store:", cont)
    temp_pre = scaledncust_vs_trial[(scaledncust_vs_trial["c_STORE_NBR"] :
    std = temp_pre["nCust_Percentage_Diff"].std()
    mean = temp_pre["nCust_Percentage_Diff"].mean()
    #print(std, mean)
    for t_month in scaledncust_vs_trial[scaledncust_vs_trial["trial_period"] == t_month]:
        pdif = scaledncust_vs_trial[(scaledncust_vs_trial["YEARMONTH"] == t_month) &&
        print(t_month, ":", (float(pdif) - mean) / std)
    print('\n')

print("Critical t-value for 95% confidence interval:")
conf_intv_95 = t.ppf(0.95, df=len(temp_pre)-1)
print(conf_intv_95)

Trial store: 77 , Control store: 233
201902 : -0.19886295797440687
201903 : 8.009609025380932
201904 : 16.114474772873923

Trial store: 86 , Control store: 155
201902 : 6.220524882227514
201903 : 10.52599074274189
201904 : 3.0763575852842706

Trial store: 88 , Control store: 40
201902 : -0.3592881735131531
201903 : 1.2575196020616801
201904 : 0.6092905590514273

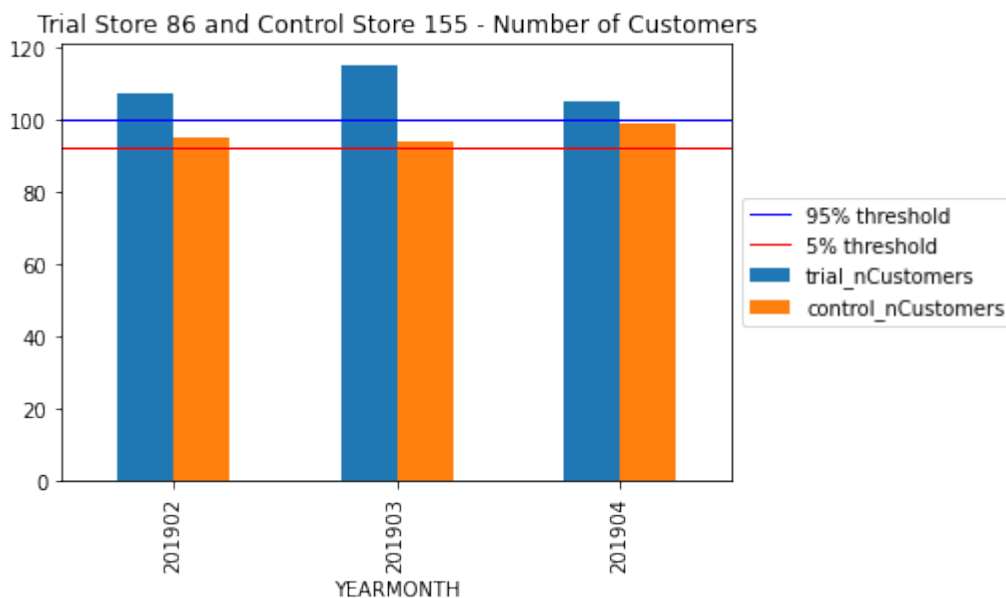
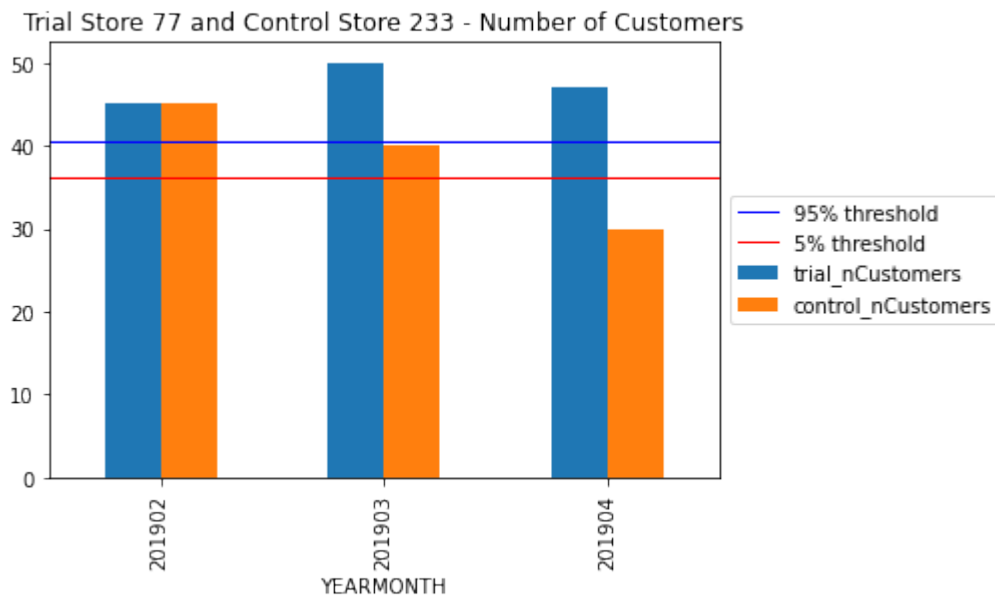
Critical t-value for 95% confidence interval:
1.9431802803927816
```

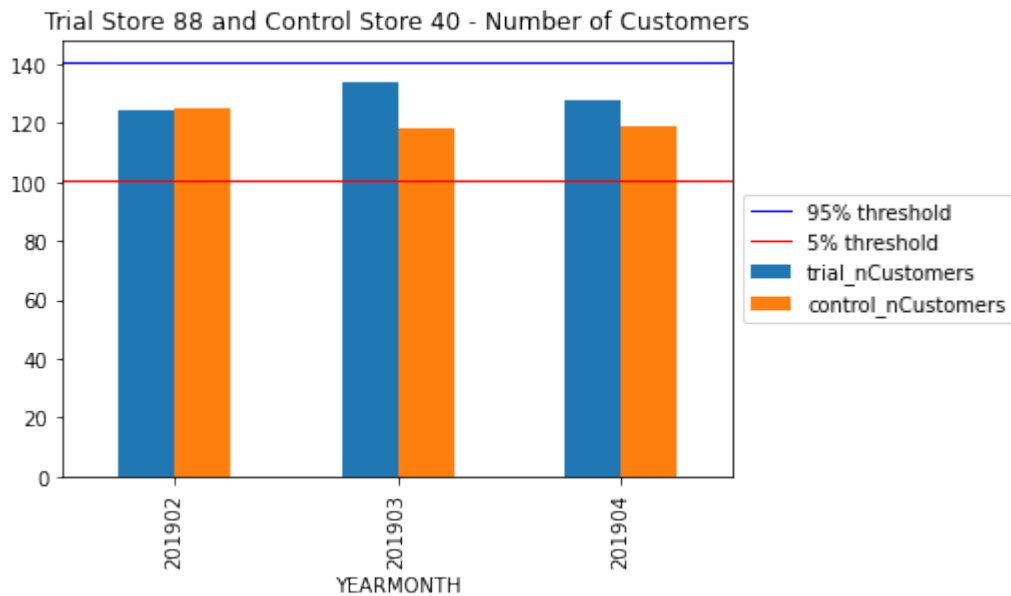
There are 5 months' increase in performance that are statistically significant (Above the 95% confidence interval t-score):

March and April trial months for trial store 77

Feb, March and April trial months for trial store 86

```
In [40]: for trial, control in trial_control_dic.items():
    a = trial_scaled_ncust_control_stores[trial_scaled_ncust_control_store]
    b = trial_full_observ[trial_full_observ["STORE_NBR"] == trial][["STORE_NBR", "YEARMONTH", "trial_nCustomers"]]
    comb = b[["YEARMONTH", "trial_nCustomers"]].merge(a[["YEARMONTH", "control_nCustomers"]])
    comb.plot.bar()
    cont_sc_ncust = trial_scaled_ncust_control_stores[trial_scaled_ncust_control_store]
    std = scaledncust_vs_trial[(scaledncust_vs_trial["c_STORE_NBR"] == control_scaled_ncust_control_store)].std()
    thresh95 = cont_sc_ncust.mean() + (cont_sc_ncust.mean() * std * 2)
    thresh5 = cont_sc_ncust.mean() - (cont_sc_ncust.mean() * std * 2)
    plt.axhline(y=thresh95,linewidth=1, color='b', label="95% threshold")
    plt.axhline(y=thresh5,linewidth=1, color='r', label="5% threshold")
    plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
    plt.title("Trial Store "+str(trial)+" and Control Store "+str(control))
    plt.savefig("TS {} and CS {} - nCustomers.png".format(trial,control),
```





We can see that Trial store 77 sales for Feb, March, and April exceeds 95% threshold of control : Same goes to store 86 sales for all 3 trial months.

1. Trial store 77: Control store 233
2. Trial store 86: Control store 155
3. Trial store 88: Control store 40
4. Both trial store 77 and 86 showed significant increase in Total Sales and Number of Customers during trial period. But not for trial store 88. Perhaps the client knows if there's anything about that differs it from the other two trial.
5. Overall the trial showed positive significant result.