```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "b863255e-4e03-4bc0-844b-2c55ba20145b",
   "metadata": {},
   "source": [
    "## Load Required Libraries and Data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "cd2eb46e-e752-4e0b-a563-e553977138c7",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "import seaborn as sns\n",
    "\n",
    "# Load the dataset\n",
    "data = pd.read_csv('C:/Users/Lekhansh/Downloads/QVI_data.csv')\n",
    "\n",
    "# Set default theme for plots\n",
    "sns.set_theme(style=\"whitegrid\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "176ad335-2020-40e3-95b0-9f8343339a63",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Introduction to Trial Analysis for Stores 77, 86, and 88\n",
    "\n",
    "This analysis focuses on assessing the impact of a store trial for stores 77, 86, and 88. The objective is to evaluate how the trial affected sales and cu\n",
    "\n",
    "## Steps Followed:\n",
    "\n",
    "1. Data Preparation:\n",
    "\n",
    "Loaded transaction data for all stores.\n",
    "Aggregated monthly sales data (totSales) and customer count data (nCustomers) for each store.\n",
    "Segmented the data into pre-trial and trial periods, with the pre-trial period covering 12 months and the trial period covering from February to April 201\n",
    "Control Store Selection:\n",
    "\n",
    "2. Trial Stores Analyzed: 77, 86, and 88.\n",
    "\n",
    "For each trial store, we calculated similarity metrics for all potential control stores.\n",
    "Correlation: Calculated correlation between trial store and control stores for both total sales and number of customers.\n",
    "Magnitude Distance: Calculated magnitude distance between trial store and control stores for both total sales and number of customers.\n",
    "Combined Scores: Merged the correlation and magnitude scores to create a combined score for each control store.\n",
    "\n",
    "Selected the control store with the highest combined score, excluding the trial store itself:\n",
    "Control Store for Trial Store 77: Store 233.\n",
    "Control Store for Trial Store 86: Store 155.\n",
    "Control Store for Trial Store 88: Store 237.\n",
    "Pre-Trial Visual Comparison:\n",
    "\n",
    "Total Sales Comparison: Generated line plots to visually compare the total sales trends for each trial store against its control store during the pre-tria\n",
    "Customer Count Comparison: Generated line plots to visually compare the number of customers for each trial store against its control store during the pre-\n",
    "Confirmed that the control stores showed similar trends to the trial stores before the trial began.\n",
    "Trial Impact Assessment:\n",
    "\n",
    "Sales Scaling: Scaled the control store’s sales data to match the trial store’s pre-trial sales levels using a scaling factor.\n",
    "Percentage Difference: Calculated the percentage difference between the trial store’s sales and the scaled control store’s sales during the trial period (\n",
    "Statistical Analysis: Used the percentage difference to assess whether the trial period performance deviated significantly from the pre-trial period.\n",
    "Visualizing Trial Impact:\n",
    "\n",
    "Created plots showing the trial store’s sales performance vs. the control store’s scaled sales during the trial period.\n",
    "Generated confidence intervals around the control store’s sales to visually assess the significance of deviations in the trial store's performance.\n",
    "Key Results:\n",
    "\n",
    "Trial Store 77: Significant difference between trial and control sales in at least two out of three trial months.\n",
    "Trial Store 86: The trial did not show a significant difference compared to the control store.\n",
    "Trial Store 88: Significant difference between trial and control sales, with the trial store performance lying outside the confidence interval in two out \n",
    "\n",
    "## Conclusion:\n",
    "This comprehensive analysis highlights the effectiveness of the store trial for stores 77 and 88, where significant differences were observed compared to \n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "35b68e91-14a8-47f6-854b-1a4fdb54fc52",
   "metadata": {},
   "source": [
    "## Create Metrics for Each Store"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "9dc832ab-62ac-42ed-be2b-00ff31a63f76",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
```

```
"    }\n",
"\n",
"    .dataframe tbody tr th {\n",
"        vertical-align: top;\n",
"    }\n",
"\n",
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>LYLTY_CARD_NBR</th>\n",
"      <th>DATE</th>\n",
"      <th>STORE_NBR</th>\n",
"      <th>TXN_ID</th>\n",
"      <th>PROD_NBR</th>\n",
"      <th>PROD_NAME</th>\n",
"      <th>PROD_QTY</th>\n",
"      <th>TOT_SALES</th>\n",
"      <th>PACK_SIZE</th>\n",
"      <th>BRAND</th>\n",
"      <th>LIFESTAGE</th>\n",
"      <th>PREMIUM_CUSTOMER</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>1000</td>\n",
"      <td>2018-10-17</td>\n",
"      <td>1</td>\n",
"      <td>1</td>\n",
"      <td>5</td>\n",
"      <td>Natural Chip        Compny SeaSalt175g</td>\n",
"      <td>2</td>\n",
"      <td>6.0</td>\n",
"      <td>175</td>\n",
"      <td>NATURAL</td>\n",
"      <td>YOUNG SINGLES/COUPLES</td>\n",
"      <td>Premium</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>1002</td>\n",
"      <td>2018-09-16</td>\n",
"      <td>1</td>\n",
"      <td>2</td>\n",
"      <td>58</td>\n",
"      <td>Red Rock Deli Chikn&amp;amp;Garlic Aioli 150g</td>\n",
"      <td>1</td>\n",
"      <td>2.7</td>\n",
"      <td>150</td>\n",
"      <td>RRD</td>\n",
"      <td>YOUNG SINGLES/COUPLES</td>\n",
"      <td>Mainstream</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>1003</td>\n",
"      <td>2019-03-07</td>\n",
"      <td>1</td>\n",
"      <td>3</td>\n",
"      <td>52</td>\n",
"      <td>Grain Waves Sour    Cream&amp;amp;Chives 210G</td>\n",
"      <td>1</td>\n",
"      <td>3.6</td>\n",
"      <td>210</td>\n",
"      <td>GRNWVES</td>\n",
"      <td>YOUNG FAMILIES</td>\n",
"      <td>Budget</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>1003</td>\n",
"      <td>2019-03-08</td>\n",
"      <td>1</td>\n",
"      <td>4</td>\n",
"      <td>106</td>\n",
"      <td>Natural ChipCo      Hony Soy Chckn175g</td>\n",
"      <td>1</td>\n",
"      <td>3.0</td>\n",
"      <td>175</td>\n",
"      <td>NATURAL</td>\n",
"      <td>YOUNG FAMILIES</td>\n",
"      <td>Budget</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>1004</td>\n",
"      <td>2018-11-02</td>\n",
"      <td>1</td>\n",
"      <td>5</td>\n",
"      <td>96</td>\n",
"      <td>WW Original Stacked Chips 160g</td>\n",
"      <td>1</td>\n",
"      <td>1.9</td>\n",
"      <td>160</td>\n",
"      <td>WOOLWORTHS</td>\n",
"      <td>OLDER SINGLES/COUPLES</td>\n",
"      <td>Mainstream</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
```

```
      "</div>"
     ],
     "text/plain": [
      "   LYLTY_CARD_NBR        DATE  STORE_NBR  TXN_ID  PROD_NBR  \\n",
      "0            1000  2018-10-17          1       1         5   \n",
      "1            1002  2018-09-16          1       2        58   \n",
      "2            1003  2019-03-07          1       3        52   \n",
      "3            1003  2019-03-08          1       4       106   \n",
      "4            1004  2018-11-02          1       5        96   \n",
      "\n",
      "                                PROD_NAME  PROD_QTY  TOT_SALES  PACK_SIZE  \\n",
      "0  Natural Chip        Compny SeaSalt175g         2        6.0        175   \n",
      "1   Red Rock Deli Chikn&Garlic Aioli 150g         1        2.7        150   \n",
      "2   Grain Waves Sour    Cream&Chives 210G         1        3.6        210   \n",
      "3  Natural ChipCo      Hony Soy Chckhl175g         1        3.0        175   \n",
      "4          WW Original Stacked Chips 160g         1        1.9        160   \n",
      "\n",
      "        BRAND              LIFESTAGE PREMIUM_CUSTOMER  \n",
      "0     NATURAL  YOUNG SINGLES/COUPLES          Premium  \n",
      "1         RRD  YOUNG SINGLES/COUPLES       Mainstream  \n",
      "2     GRNWVES         YOUNG FAMILIES           Budget  \n",
      "3     NATURAL         YOUNG FAMILIES           Budget  \n",
      "4  WOOLWORTHS  OLDER SINGLES/COUPLES       Mainstream  "
     ]
    },
    "execution_count": 2,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "data.head()"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "f476d116-2969-4ff8-80fa-c0f88bed2b20",
  "metadata": {},
  "source": [
   "The goal is to create monthly sales metrics for each store, including total sales, number of customers, and transactions per customer."
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 3,
  "id": "7b81c13b-e074-483e-ad00-4ae13c9ef331",
  "metadata": {},
  "outputs": [],
  "source": [
   "# Convert the date column if needed and create 'YEARMONTH'\n",
   "data['DATE'] = pd.to_datetime(data['DATE'])\n",
   "data['YEARMONTH'] = data['DATE'].dt.to_period('M')\n",
   "\n",
   "# Create metrics for each store and month\n",
   "metrics = data.groupby(['STORE_NBR', 'YEARMONTH']).agg(\n",
   "    totSales=('TOT_SALES', 'sum'),\n",
   "    nCustomers=('LYLTY_CARD_NBR', 'nunique'),\n",
   "    nTxnPerCust=('TXN_ID', 'count')\n",
   ").reset_index()\n",
   "\n",
   "# Filter stores with full observation periods (12 months pre-trial)\n",
   "pre_trial_data = metrics[metrics['YEARMONTH'] < '2019-02']\n",
   "stores_with_full_obs = pre_trial_data.groupby('STORE_NBR').filter(lambda x: len(x) == 12)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "a3bd74d3-5abe-4f77-95a4-b95b0646e120",
  "metadata": {},
  "source": [
   "## Function to Calculate Correlation\n",
   "\n",
   "Let's Write a function to calculate correlation for the trial and control stores based on different metrics."
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 4,
  "id": "e82a345d-cfe9-4ad4-9ff8-758619347b41",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "     Trial_Store  Control_Store  Correlation\n",
     "0             77              1     0.075218\n",
     "1             77              2    -0.263079\n",
     "2             77              3     0.806644\n",
     "3             77              4    -0.263300\n",
     "4             77              5    -0.110652\n",
     "..           ...            ...          ...\n",
     "259           77            268     0.344757\n",
     "260           77            269    -0.315730\n",
     "261           77            270     0.315430\n",
     "262           77            271     0.355487\n",
     "263           77            272     0.117622\n",
     "\n",
     "[264 rows x 3 columns]\n"
    ]
   }
  ],
  "source": [
   "# Function to calculate correlation between trial store and control stores\n",
   "def calculate_correlation(input_data, metric_col, trial_store):\n",
   "    # Filter trial store data\n",
```

```
  "    trial_data = input_data[input_data['STORE_NBR'] == trial_store][['YEARMONTH', metric_col]]\n",
  "    \n",
  "    correlation_list = []\n",
  "    \n",
  "    for store in input_data['STORE_NBR'].unique():\n",
  "        if store != trial_store:\n",
  "            control_data = input_data[input_data['STORE_NBR'] == store][['YEARMONTH', metric_col]]\n",
  "            \n",
  "            # Merge trial and control data to ensure they have the same months\n",
  "            merged_data = pd.merge(trial_data, control_data, on='YEARMONTH', suffixes=('_trial', '_control'))\n",
  "            \n",
  "            # Calculate correlation only if there are enough matching months\n",
  "            if len(merged_data) > 1:  # Ensure enough data points for correlation\n",
  "                corr_value = merged_data[metric_col + '_trial'].corr(merged_data[metric_col + '_control'])\n",
  "                correlation_list.append((trial_store, store, corr_value))\n",
  "    \n",
  "    return pd.DataFrame(correlation_list, columns=['Trial_Store', 'Control_Store', 'Correlation'])\n",
  "\n",
  "# Example: Calculate correlation for store 77 based on total sales\n",
  "correlation_sales = calculate_correlation(pre_trial_data, 'totSales', 77)\n",
  "print(correlation_sales)"
 ]
},
{
 "cell_type": "markdown",
 "id": "30fdc1dd-760a-452a-b334-d44b5fde8604",
 "metadata": {},
 "source": [
  "## Function to Calculate Magnitude Distance\n",
  "\n",
  "This function calculates the magnitude distance, which measures the difference between trial and control stores based on a given metric."
 ]
},
{
 "cell_type": "code",
 "execution_count": 5,
 "id": "db48ab10-7356-4b22-a897-a4ce1646d4d9",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "     Trial_Store  Control_Store     Distance\n",
    "0             77             1    59.900000\n",
    "1             77             2    81.500000\n",
    "2             77             3   832.450000\n",
    "3             77             4  1061.142857\n",
    "4             77             5   577.242857\n",
    "..           ...           ...          ...\n",
    "265           77           268    50.064286\n",
    "266           77           269   709.357143\n",
    "267           77           270   714.135714\n",
    "268           77           271   580.871429\n",
    "269           77           272   149.335714\n",
    "\n",
    "[270 rows x 3 columns]\n"
   ]
  }
 ],
 "source": [
  "# Function to calculate magnitude distance\n",
  "def calculate_magnitude_distance(input_data, metric_col, trial_store):\n",
  "    # Filter trial store data\n",
  "    trial_values = input_data[input_data['STORE_NBR'] == trial_store][['YEARMONTH', metric_col]]\n",
  "    \n",
  "    dist_list = []\n",
  "    \n",
  "    for store in input_data['STORE_NBR'].unique():\n",
  "        if store != trial_store:\n",
  "            control_values = input_data[input_data['STORE_NBR'] == store][['YEARMONTH', metric_col]]\n",
  "            \n",
  "            # Merge trial and control data to ensure they have the same months\n",
  "            merged_data = pd.merge(trial_values, control_values, on='YEARMONTH', suffixes=('_trial', '_control'))\n",
  "            \n",
  "            if len(merged_data) > 0:  # Ensure there is data to calculate distance\n",
  "                # Calculate absolute difference between the two stores for the given metric\n",
  "                distance = np.abs(merged_data[metric_col + '_trial'] - merged_data[metric_col + '_control']).mean()\n",
  "                dist_list.append((trial_store, store, distance))\n",
  "    \n",
  "    return pd.DataFrame(dist_list, columns=['Trial_Store', 'Control_Store', 'Distance'])\n",
  "\n",
  "# Example: Calculate magnitude distance for store 77 based on total sales\n",
  "magnitude_sales = calculate_magnitude_distance(pre_trial_data, 'totSales', 77)\n",
  "print(magnitude_sales)"
 ]
},
{
 "cell_type": "markdown",
 "id": "91262962-82c4-44a9-90e0-2de9c7fec67f",
 "metadata": {},
 "source": [
  "## Combine Correlation and Magnitude Scores\n",
  "\n",
  "Next, combine the correlation and magnitude distance scores to rank potential control stores."
 ]
},
{
 "cell_type": "code",
 "execution_count": 6,
 "id": "927893ba-fe0d-48fe-8c4b-403abac2d1d4",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
```

```
    "text": [
     "     Trial_Store  Control_Store  Correlation     Distance  Final_Score\n",
     "225           77            233     0.903774    18.828571    -8.462399\n",
     "246           77            255     0.191091    28.571429   -13.690169\n",
     "183           77            188     0.042708    31.100000   -15.028646\n",
     "52            77             53     0.532764    32.471429   -15.469332\n",
     "126           77            131     0.403299    33.042857   -15.819779\n",
     "..           ...            ...          ...          ...          ...\n",
     "57            77             58     0.115051  1068.535714  -533.710332\n",
     "160           77            165     0.343931  1076.971429  -537.813749\n",
     "229           77            237     0.575200  1095.714286  -547.069543\n",
     "84            77             88    -0.114199  1097.800000  -548.457099\n",
     "218           77            226     0.167132  1220.021429  -609.427148\n",
     "\n",
     "[264 rows x 5 columns]\n"
    ]
   }
  ],
  "source": [
   "# Combine correlation and magnitude distance to get a score\n",
   "def combine_scores(corr_df, mag_df):\n",
   "    combined_df = pd.merge(corr_df, mag_df, on=['Trial_Store', 'Control_Store'])\n",
   "    combined_df['Final_Score'] = 0.5 * combined_df['Correlation'] + 0.5 * (1 - combined_df['Distance'])\n",
   "    return combined_df.sort_values(by='Final_Score', ascending=False)\n",
   "\n",
   "# Combine scores for store 77\n",
   "combined_scores = combine_scores(correlation_sales, magnitude_sales)\n",
   "print(combined_scores)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "0b1f1242-ab00-44fd-b8d0-52b654cd34b6",
  "metadata": {},
  "source": [
   "## Visualize Pre-Trial Trends\n",
   "\n",
   "Now, plotting the pre-trial trends for the trial and control stores to visually inspect their similarity."
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 10,
  "id": "8016c37d-1747-4d81-b1d9-ec0fe826facf",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "C:\\Users\\Lekhansh\\AppData\\Local\\Temp\\ipykernel_10084\\725089237.py:8: SettingWithCopyWarning: \n",
     "A value is trying to be set on a copy of a slice from a DataFrame.\n",
     "Try using .loc[row_indexer,col_indexer] = value instead\n",
     "\n",
     "See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy\n",
     "  trial_data['YEARMONTH'] = trial_data['YEARMONTH'].dt.to_timestamp()\n",
     "C:\\Users\\Lekhansh\\AppData\\Local\\Temp\\ipykernel_10084\\725089237.py:9: SettingWithCopyWarning: \n",
     "A value is trying to be set on a copy of a slice from a DataFrame.\n",
     "Try using .loc[row_indexer,col_indexer] = value instead\n",
     "\n",
     "See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy\n",
     "  control_data['YEARMONTH'] = control_data['YEARMONTH'].dt.to_timestamp()\n"
    ]
   },
   {
    "data": {
     "image/png":
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   }
  ],
  "source": [
   "# Plot sales trends between trial and control stores before the trial period\n",
   "def plot_sales_trends(data, trial_store, control_store):\n",
   "    # Filter the trial and control data\n",
   "    trial_data = data[data['STORE_NBR'] == trial_store]\n",
   "    control_data = data[data['STORE_NBR'] == control_store]\n",
   "    \n",
   "    # Convert 'YEARMONTH' from Period to Datetime for plotting\n",
   "    trial_data['YEARMONTH'] = trial_data['YEARMONTH'].dt.to_timestamp()\n",
   "    control_data['YEARMONTH'] = control_data['YEARMONTH'].dt.to_timestamp()\n",
   "    \n",
   "    # Plot the sales trend\n",
   "    plt.figure(figsize=(10, 6))\n",
   "    plt.plot(trial_data['YEARMONTH'], trial_data['totSales'], label=f'Trial Store {trial_store}')\n",
   "    plt.plot(control_data['YEARMONTH'], control_data['totSales'], label=f'Control Store {control_store}')\n",
   "    plt.title('Pre-Trial Sales Comparison')\n",
   "    plt.xlabel('Year-Month')\n",
   "    plt.ylabel('Total Sales')\n",
   "    plt.xticks(rotation=45)\n",
   "    plt.legend()\n",
   "    plt.show()\n",
   "\n",
   "# Example: Visualize the sales trend comparison\n",
   "plot_sales_trends(pre_trial_data, 77, combined_scores.iloc[0]['Control_Store'])"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "0d11c49f-7b13-43ab-bada-9e37ce6d9a13",
  "metadata": {},
  "source": [
```

```
  "## Assess the Trial Impact\n",
  "\n",
  "To assess the impact of the trial, calculate the percentage difference between the trial and control stores during the trial period."
 ]
},
{
 "cell_type": "code",
 "execution_count": 13,
 "id": "1a4d538a-69e6-404e-b6a6-4da4d1ba9059",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "  YEARMONTH  STORE_NBR_trial  totSales  Control_Sales_Scaled  Percentage_Diff\n",
    "0   2019-02               77     235.0            249.762622       -5.910661\n",
    "1   2019-03               77     278.5            203.802205       36.652103\n",
    "2   2019-04               77     263.5            162.345704       62.307960\n"
   ]
  },
  {
   "name": "stderr",
   "output_type": "stream",
   "text": [
    "C:\\Users\\Lekhansh\\AppData\\Local\\Temp\\ipykernel_21992\\828308693.py:12: SettingWithCopyWarning: \n",
    "A value is trying to be set on a copy of a slice from a DataFrame.\n",
    "Try using .loc[row_indexer,col_indexer] = value instead\n",
    "\n",
    "See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy\n",
    "  trial_period_data['Control_Sales_Scaled'] = trial_period_data[trial_period_data['STORE_NBR'] == control_store]['totSales'] * scaling_factor\n"
   ]
  }
 ],
 "source": [
  "# Filter data for the trial period (Feb 2019 to April 2019)\n",
  "trial_period_data = metrics[metrics['YEARMONTH'].between('2019-02', '2019-04')]\n",
  "\n",
  "# Get the control store for trial store 77\n",
  "control_store = combined_scores.iloc[0]['Control_Store']\n",
  "\n",
  "# Scale control store sales based on pre-trial data\n",
  "scaling_factor = pre_trial_data[pre_trial_data['STORE_NBR'] == 77]['totSales'].sum() / \\\n",
  "                 pre_trial_data[pre_trial_data['STORE_NBR'] == control_store]['totSales'].sum()\n",
  "\n",
  "# Apply scaling factor to control store sales during the trial period\n",
  "trial_period_data['Control_Sales_Scaled'] = trial_period_data[trial_period_data['STORE_NBR'] == control_store]['totSales'] * scaling_factor\n",
  "\n",
  "# Filter data for trial store 77 during the trial period\n",
  "trial_store_data = trial_period_data[trial_period_data['STORE_NBR'] == 77]\n",
  "\n",
  "# Filter data for the scaled control store sales during the trial period\n",
  "control_store_data = trial_period_data[trial_period_data['STORE_NBR'] == control_store]\n",
  "\n",
  "# Merge the trial store data with control store data, keeping STORE_NBR\n",
  "trial_period_data = pd.merge(trial_store_data[['YEARMONTH', 'STORE_NBR', 'totSales']], \n",
  "                             control_store_data[['YEARMONTH', 'STORE_NBR', 'Control_Sales_Scaled']], \n",
  "                             on='YEARMONTH', suffixes=('_trial', '_control'))\n",
  "\n",
  "# Calculate percentage difference between trial and scaled control sales\n",
  "trial_period_data['Percentage_Diff'] = ((trial_period_data['totSales'] - trial_period_data['Control_Sales_Scaled']) / trial_period_data['Control_Sales_Sca
  "\n",
  "print(trial_period_data[['YEARMONTH', 'STORE_NBR_trial', 'totSales', 'Control_Sales_Scaled', 'Percentage_Diff']])"
 ]
},
{
 "cell_type": "markdown",
 "id": "6cd24a7a-5435-42bc-b591-81f44b3a57f5",
 "metadata": {},
 "source": [
  "## Statistical Testing\n",
  "\n",
  "To test if the observed differences are statistically significant, use a t-test."
 ]
},
{
 "cell_type": "code",
 "execution_count": 14,
 "id": "458ae29f-74c2-4e44-bc7a-d7a508cc6389",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "T-statistic: -4.532299398442595e-16, P-value: 0.9999999999999997\n"
   ]
  }
 ],
 "source": [
  "from scipy.stats import ttest_1samp\n",
  "\n",
  "# Perform t-test on the percentage differences\n",
  "pre_trial_diffs = pre_trial_data[pre_trial_data['STORE_NBR'] == 77]['totSales'].values - \\\n",
  "                  pre_trial_data[pre_trial_data['STORE_NBR'] == control_store]['totSales'].values * scaling_factor\n",
  "\n",
  "t_stat, p_value = ttest_1samp(pre_trial_diffs, 0)\n",
  "\n",
  "print(f\"T-statistic: {t_stat}, P-value: {p_value}\")"
 ]
},
{
 "cell_type": "markdown",
 "id": "ab4fbb71-0dca-4f32-abaa-283e64511872",
 "metadata": {},
 "source": [
```

```
  "## Plot Results During Trial Period\n",
  "\n",
  "Finally, plot the trial and control store sales and indicate the confidence intervals."
 ]
},
{
 "cell_type": "code",
 "execution_count": 11,
 "id": "72bda761-bd98-452e-b048-935e73c1ec77",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Index(['YEARMONTH', 'totSales', 'Control_Sales_Scaled', 'Percentage_Diff'], dtype='object')\n"
   ]
  }
 ],
 "source": [
  "# Check the columns in trial_period_data to see if 'STORE_NBR' is present\n",
  "print(trial_period_data.columns)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 16,
 "id": "07fe7825-f091-4896-b814-a8f8ea48d5a6",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Plot the sales during the trial period\n",
  "def plot_trial_results(data, trial_store, control_store):\n",
  "    # Convert 'YEARMONTH' from Period to Datetime for plotting\n",
  "    data['YEARMONTH'] = data['YEARMONTH'].dt.to_timestamp()\n",
  "    \n",
  "    plt.figure(figsize=(10, 6))\n",
  "    \n",
  "    # Plot trial store sales\n",
  "    plt.plot(data['YEARMONTH'], data['totSales'], label=f'Trial Store {trial_store}')\n",
  "    \n",
  "    # Plot scaled control store sales\n",
  "    plt.plot(data['YEARMONTH'], data['Control_Sales_Scaled'], label=f'Control Store {control_store} (Scaled)')\n",
  "    \n",
  "    # Add confidence interval (±5%)\n",
  "    plt.fill_between(data['YEARMONTH'], data['Control_Sales_Scaled'] * 0.95, \n",
  "                     data['Control_Sales_Scaled'] * 1.05, color='gray', alpha=0.3, label='95% CI')\n",
  "    \n",
  "    plt.title('Trial vs Control Sales')\n",
  "    plt.xlabel('Year-Month')\n",
  "    plt.ylabel('Total Sales')\n",
  "    plt.xticks(rotation=45)\n",
  "    plt.legend()\n",
  "    plt.show()\n",
  "\n",
  "# Example: Plot trial period results for store 77\n",
  "plot_trial_results(trial_period_data, 77, combined_scores.iloc[0]['Control_Store'])"
 ]
},
{
 "cell_type": "markdown",
 "id": "d20851b3-9c5e-4d53-9693-fd30b8efef0e",
 "metadata": {},
 "source": [
  "## Calculate Correlation and Magnitude Distance for Trial Stores 86 and 88\n",
  "\n",
  "We will first calculate the correlation and magnitude distance for sales and customer counts for the potential control stores of Trial Stores 86 and 88."
 ]
},
{
 "cell_type": "code",
 "execution_count": 17,
 "id": "f9659a78-8471-4dfd-9e93-deb9b736d621",
 "metadata": {},
 "outputs": [
  {
   "name": "stderr",
   "output_type": "stream",
   "text": [
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2897: RuntimeWarning: invalid value encountered in divide\n",
    "  c /= stddev[:, None]\n",
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2898: RuntimeWarning: invalid value encountered in divide\n",
    "  c /= stddev[None, :]\n",
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2897: RuntimeWarning: invalid value encountered in divide\n",
    "  c /= stddev[:, None]\n",
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2898: RuntimeWarning: invalid value encountered in divide\n",
    "  c /= stddev[None, :]\n"
   ]
  },
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Selected Control Store for Trial Store 86: 155.0\n"
   ]
```

```
   },
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2897: RuntimeWarning: invalid value encountered in divide\n",
     "  c /= stddev[:, None]\n",
     "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2898: RuntimeWarning: invalid value encountered in divide\n",
     "  c /= stddev[None, :]\n",
     "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2897: RuntimeWarning: invalid value encountered in divide\n",
     "  c /= stddev[:, None]\n",
     "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\numpy\\lib\\function_base.py:2898: RuntimeWarning: invalid value encountered in divide\n",
     "  c /= stddev[None, :]\n"
    ]
   },
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Selected Control Store for Trial Store 88: 237.0\n"
    ]
   }
  ],
  "source": [
   "# Define the trial stores\n",
   "trial_stores = [86, 88]\n",
   "\n",
   "# Placeholder to store results for both trial stores\n",
   "results = {}\n",
   "\n",
   "for trial_store in trial_stores:\n",
   "    # Calculate correlation for sales\n",
   "    corr_nSales = calculate_correlation(pre_trial_data, 'totSales', trial_store)\n",
   "    # Calculate correlation for number of customers\n",
   "    corr_nCustomers = calculate_correlation(pre_trial_data, 'nCustomers', trial_store)\n",
   "    \n",
   "    # Calculate magnitude distance for sales\n",
   "    magnitude_nSales = calculate_magnitude_distance(pre_trial_data, 'totSales', trial_store)\n",
   "    # Calculate magnitude distance for number of customers\n",
   "    magnitude_nCustomers = calculate_magnitude_distance(pre_trial_data, 'nCustomers', trial_store)\n",
   "    \n",
   "    # Combine scores: sales and customers\n",
   "    score_nSales = combine_scores(corr_nSales, magnitude_nSales)\n",
   "    score_nCustomers = combine_scores(corr_nCustomers, magnitude_nCustomers)\n",
   "    \n",
   "    # Combine final control scores\n",
   "    score_Control = pd.merge(score_nSales[['Control_Store', 'Final_Score']], \n",
   "                             score_nCustomers[['Control_Store', 'Final_Score']], \n",
   "                             on='Control_Store', suffixes=('_Sales', '_Customers'))\n",
   "    \n",
   "    # Calculate the final combined score by averaging the sales and customer scores\n",
   "    score_Control['finalControlScore'] = (score_Control['Final_Score_Sales'] + score_Control['Final_Score_Customers']) / 2\n",
   "    \n",
   "    # Select the best control store (highest score, but not the store itself)\n",
   "    control_store = score_Control.sort_values(by='finalControlScore', ascending=False).iloc[0]['Control_Store']\n",
   "    results[trial_store] = control_store\n",
   "    print(f\"Selected Control Store for Trial Store {trial_store}: {control_store}\")"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "680488a2-f28d-4636-9f60-4c2d41fa8162",
  "metadata": {},
  "source": [
   "## Visualize Pre-Trial Trends for Sales and Customer Counts\n",
   "\n",
   "We will create visualizations to check if the selected control stores are good matches for the trial stores based on sales and customer trends in the pre-
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 18,
  "id": "84007aef-9d60-409e-9880-775a276e5bb2",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "image/png":
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   },
   {
    "data": {
     "image/png":
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   },
   {
    "data": {
     "image/png":
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   },
   {
```

```
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Visual checks on trends based on the drivers for pre-trial period\n",
  "def plot_pre_trial_comparison(trial_store, control_store, metric, metric_label):\n",
  "    trial_data = pre_trial_data[pre_trial_data['STORE_NBR'] == trial_store]\n",
  "    control_data = pre_trial_data[pre_trial_data['STORE_NBR'] == control_store]\n",
  "    \n",
  "    # Plot the comparison between trial and control stores\n",
  "    plt.figure(figsize=(10, 6))\n",
  "    plt.plot(trial_data['YEARMONTH'].dt.to_timestamp(), trial_data[metric], label=f'Trial Store {trial_store}')\n",
  "    plt.plot(control_data['YEARMONTH'].dt.to_timestamp(), control_data[metric], label=f'Control Store {control_store}')\n",
  "    plt.title(f'{metric_label} Comparison: Trial Store {trial_store} vs Control Store {control_store}')\n",
  "    plt.xlabel('Year-Month')\n",
  "    plt.ylabel(metric_label)\n",
  "    plt.legend()\n",
  "    plt.xticks(rotation=45)\n",
  "    plt.show()\n",
  "\n",
  "# Plot sales comparison for both trial stores\n",
  "for trial_store in trial_stores:\n",
  "    control_store = results[trial_store]\n",
  "    plot_pre_trial_comparison(trial_store, control_store, 'totSales', 'Total Sales')\n",
  "    plot_pre_trial_comparison(trial_store, control_store, 'nCustomers', 'Number of Customers')"
 ]
},
{
 "cell_type": "markdown",
 "id": "95df44cd-269c-4138-a7ff-a7f7ecca2b47",
 "metadata": {},
 "source": [
  "## Assess the Impact of the Trial\n",
  "\n",
  "We will now assess the impact of the trial by calculating the percentage difference between the scaled control store sales and the trial store sales durin
 ]
},
{
 "cell_type": "code",
 "execution_count": 23,
 "id": "37acf720-b25d-4874-9d45-81c924df2f6b",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Trial Impact for Store 86 vs Control Store 155.0:\n",
    "  YEARMONTH  totSales  Control_Sales_Scaled  Percentage_Diff\n",
    "0   2019-02     913.2           864.522060         5.630619\n",
    "1   2019-03    1026.8           780.320405        31.586973\n",
    "2   2019-04     848.2           819.317024         3.525250\n",
    "Trial Impact for Store 88 vs Control Store 237.0:\n",
    "  YEARMONTH  totSales  Control_Sales_Scaled  Percentage_Diff\n",
    "0   2019-02    1370.2          1406.989143        -2.614742\n",
    "1   2019-03    1477.2          1210.082775        22.074294\n",
    "2   2019-04    1439.4          1206.477165        19.306029\n"
   ]
  }
 ],
 "source": [
  "# Function to assess trial impact based on percentage difference\n",
  "def assess_trial_impact(trial_store, control_store):\n",
  "    # Filter data for the trial period (Feb 2019 to April 2019)\n",
  "    trial_period_data = metrics[metrics['YEARMONTH'].between('2019-02', '2019-04')]\n",
  "\n",
  "    # Scale control store sales to match pre-trial trial store sales\n",
  "    scaling_factor_sales = pre_trial_data[pre_trial_data['STORE_NBR'] == trial_store]['totSales'].sum() / \\\n",
  "                           pre_trial_data[pre_trial_data['STORE_NBR'] == control_store]['totSales'].sum()\n",
  "\n",
  "    # Apply scaling factor to control store sales during the trial period\n",
  "    control_store_data = trial_period_data[trial_period_data['STORE_NBR'] == control_store].copy()\n",
  "    control_store_data['Control_Sales_Scaled'] = control_store_data['totSales'] * scaling_factor_sales\n",
  "\n",
  "    # Filter trial store data during the trial period\n",
  "    trial_store_data = trial_period_data[trial_period_data['STORE_NBR'] == trial_store]\n",
  "\n",
  "    # Merge the trial store data with the scaled control store data on 'YEARMONTH'\n",
  "    trial_period_data_merged = pd.merge(trial_store_data[['YEARMONTH', 'totSales']],\n",
  "                                        control_store_data[['YEARMONTH', 'Control_Sales_Scaled']],\n",
  "                                        on='YEARMONTH', how='inner')\n",
  "\n",
  "    # Calculate the percentage difference\n",
  "    trial_period_data_merged['Percentage_Diff'] = ((trial_period_data_merged['totSales'] - \n",
  "                                                    trial_period_data_merged['Control_Sales_Scaled']) / \n",
  "                                                   trial_period_data_merged['Control_Sales_Scaled']) * 100\n",
  "    \n",
  "    return trial_period_data_merged\n",
  "\n",
  "# Assess trial impact for both trial stores\n",
  "for trial_store in trial_stores:\n",
  "    control_store = results[trial_store]\n",
  "    trial_impact = assess_trial_impact(trial_store, control_store)\n",
  "    print(f\"Trial Impact for Store {trial_store} vs Control Store {control_store}:\")\n",
  "    print(trial_impact[['YEARMONTH', 'totSales', 'Control_Sales_Scaled', 'Percentage_Diff']])"
 ]
},
{
 "cell_type": "markdown",
```

```
   "id": "b43d83ad-909d-4c8c-a72f-61424435fd22",
   "metadata": {},
   "source": [
    "## Conclusion\n",
    "\n",
    "For each trial store (86 and 88), we calculate and visualize:\n",
    "1. Correlation and magnitude distance between the trial and control stores.\n",
    "2. Total sales and number of customer trends before the trial.\n",
    "3. Impact of the trial during the trial period, using scaled control store data.\n",
    "\n",
    "We can conclude whether the trial stores show a significant difference in sales or customer numbers compared to their respective control stores. This allo
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "efa4144e-a6a2-420f-8654-ecc523c004b9",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.11.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```