Supplementary Activity:

Using the CSV files provided and what we have learned so far in this module complete the following exercises:

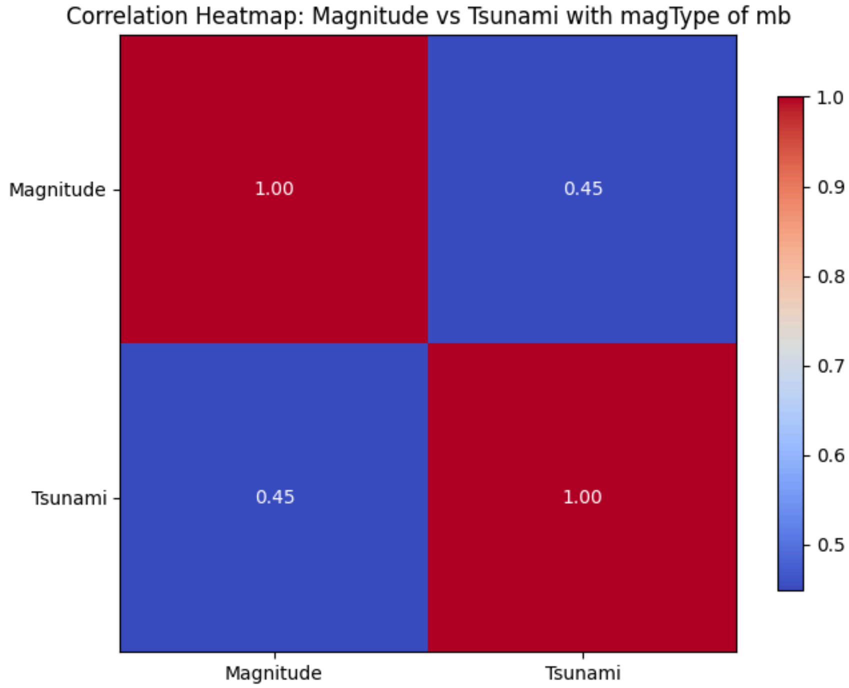
1. Using seaborn, create a heatmap to visualize the correlation coefficients between earthquake magnitude and whether there was a tsunami with the magType of mb.
2. Create a box plot of Facebook volume traded and closing prices, and draw reference lines for the bounds of a Tukey fence with a multiplier of 1.5. The bounds will be at Q1 - 1.5 * IQR and Q3 + 1.5 * IQR. Be sure to use the quantile() method on the data to make this easier. (Pick whichever orientation you prefer for the plot, but make sure to use subplots.)
3. Fill in the area between the bounds in the plot from exercise #2.
4. Use axvspan() to shade a rectangle from '2018-07-25' to '2018-07-31', which marks the large decline in Facebook price on a line plot of the closing price.
5. Using the Facebook stock price data, annotate the following three events on a line plot of the closing price:

- Disappointing user growth announced after close on July 25, 2018
- Cambridge Analytica story breaks on March 19, 2018 (when it affected the market)
- FTC launches investigation on March 20, 2018

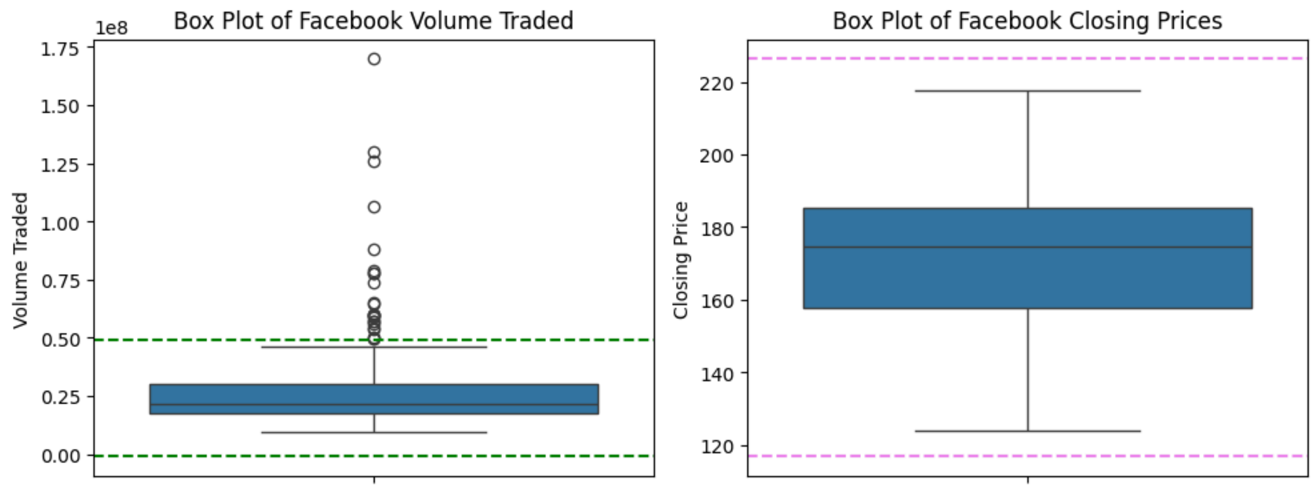
6. Modify the reg_resid_plots() function to use a matplotlib colormap instead of cycling between two colors. Remember, for this use case, we should pick a qualitative colormap or make our own.

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

fb = pd.read_csv('fb_stock_prices_2018.csv', index_col='date', parse_dates=True)
quakes = pd.read_csv('earthquakes-1.csv')
```

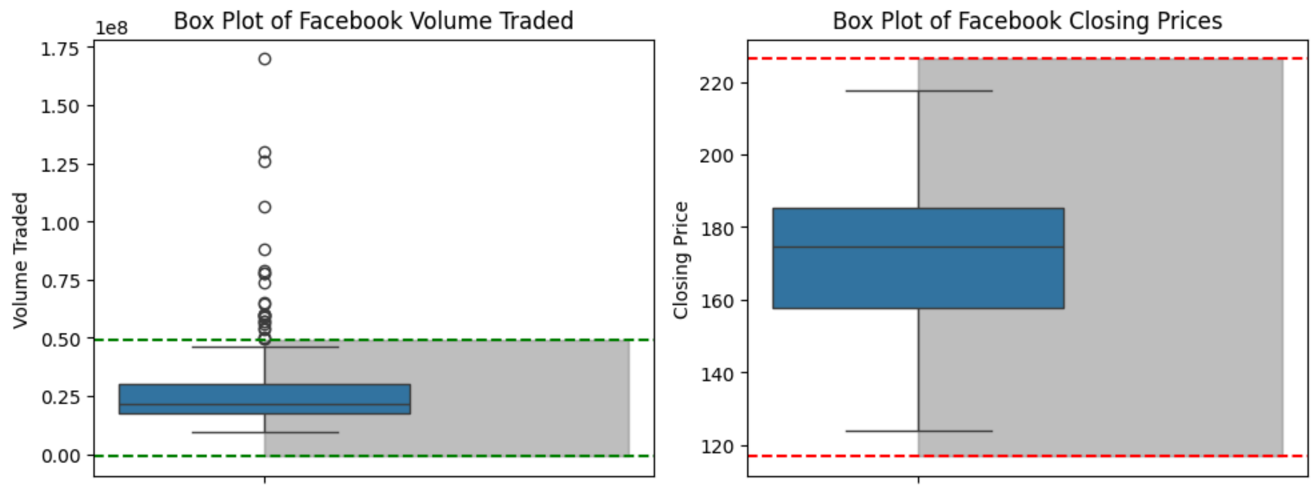
```
# 1. Using seaborn, create a heatmap to visualize the correlation coefficients between earthquake magnitude and whether there was a tsunami w
mb_quakes = quakes[quakes['magType'] == 'mb']
corr_coefficient = np.corrcoef(mb_quakes['mag'], mb_quakes['tsunami'])

# Create the heatmap
fig, ax = plt.subplots(figsize=(8, 6))
heatmap = ax.imshow(corr_coefficient, cmap='coolwarm', interpolation='nearest')
for i in range(corr_coefficient.shape[0]):
    for j in range(corr_coefficient.shape[1]):
        ax.text(j, i, f'{corr_coefficient[i, j]:.2f}', ha='center', va='center', color='white')

ax.set_title('Correlation Heatmap: Magnitude vs Tsunami with magType of mb')
plt.colorbar(heatmap, shrink=0.8)
plt.xticks([0, 1], ['Magnitude', 'Tsunami'])
plt.yticks([0, 1], ['Magnitude', 'Tsunami'])
plt.show()
```




```
# 2. Create a box plot of Facebook volume traded and closing prices, and draw reference lines for the bounds of a Tukey fence with a multip
# The bounds will be at Q1 - 1.5 * IQR and Q3 + 1.5 * IQR. Be sure to use the quantile() method on the data to make this easier.
# (Pick whichever orientation you prefer for the plot, but make sure to use subplots.)

# Calculate quartiles and IQR for volume traded
Q1_vol = fb['volume'].quantile(0.25)
Q3_vol = fb['volume'].quantile(0.75)
IQR_vol = Q3_vol - Q1_vol

# Calculate quartiles and IQR for closing prices
Q1_close = fb['close'].quantile(0.25)
Q3_close = fb['close'].quantile(0.75)
IQR_close = Q3_close - Q1_close

# Calculate Tukey fence bounds for volume traded
lower_bound_vol = Q1_vol - 1.5 * IQR_vol
upper_bound_vol = Q3_vol + 1.5 * IQR_vol

# Calculate Tukey fence bounds for closing prices
lower_bound_close = Q1_close - 1.5 * IQR_close
upper_bound_close = Q3_close + 1.5 * IQR_close

# Creating a figure with subplots
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

# Plot boxplot for volume traded
sns.boxplot(y='volume', data=fb[['volume']], ax=axes[0])
axes[0].axhline(y=lower_bound_vol, color='green', linestyle='--', label='Lower Bound')
axes[0].axhline(y=upper_bound_vol, color='green', linestyle='--', label='Upper Bound')
axes[0].set_ylabel('Volume Traded')
axes[0].set_title('Box Plot of Facebook Volume Traded')

# Plot boxplot for closing prices
sns.boxplot(y='close', data=fb[['close']], ax=axes[1])
axes[1].axhline(y=lower_bound_close, color='violet', linestyle='--', label='Lower Bound')
axes[1].axhline(y=upper_bound_close, color='violet', linestyle='--', label='Upper Bound')
axes[1].set_ylabel('Closing Price')
axes[1].set_title('Box Plot of Facebook Closing Prices')

plt.tight_layout()
```

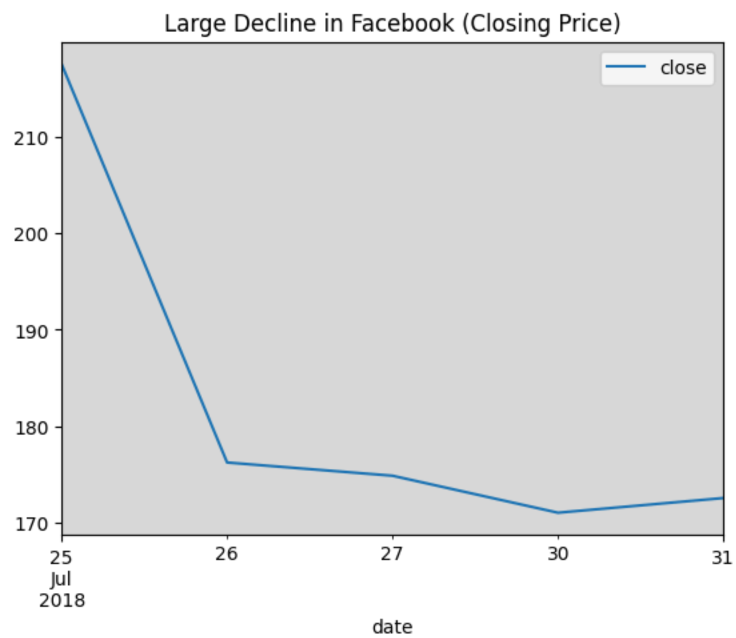



```
# 3. Fill in the area between the bounds in the plot from exercise #2.

# Calculate quartiles and IQR for volume traded
Q1_vol = fb['volume'].quantile(0.25)
Q3_vol = fb['volume'].quantile(0.75)
IQR_vol = Q3_vol - Q1_vol

# Calculate quartiles and IQR for closing prices
Q1_close = fb['close'].quantile(0.25)
Q3_close = fb['close'].quantile(0.75)
IQR_close = Q3_close - Q1_close

# Calculate Tukey fence bounds for volume traded
lower_bound_vol = Q1_vol - 1.5 * IQR_vol
upper_bound_vol = Q3_vol + 1.5 * IQR_vol

# Calculate Tukey fence bounds for closing prices
lower_bound_close = Q1_close - 1.5 * IQR_close
upper_bound_close = Q3_close + 1.5 * IQR_close

# Creating a subplot
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

# Plot boxplot for volume traded
sns.boxplot(y='volume', data=fb[['volume']], ax=axes[0])

# Add Tukey fences and shaded area between bounds
axes[0].axhline(y=lower_bound_vol, color='green', linestyle='--', label='Lower Bound')
axes[0].axhline(y=upper_bound_vol, color='green', linestyle='--', label='Upper Bound')
axes[0].fill_between([0, 1], lower_bound_vol, upper_bound_vol, color='gray', alpha=0.5)
axes[0].set_ylabel('Volume Traded')
axes[0].set_title('Box Plot of Facebook Volume Traded')

# Plot boxplot for closing prices
sns.boxplot(y='close', data=fb[['close']], ax=axes[1])

# Add Tukey fences and shaded area between bounds
axes[1].axhline(y=lower_bound_close, color='red', linestyle='--', label='Lower Bound')
axes[1].axhline(y=upper_bound_close, color='red', linestyle='--', label='Upper Bound')
axes[1].fill_between([0, 1], lower_bound_close, upper_bound_close, color='gray', alpha=0.5)
axes[1].set_ylabel('Closing Price')
axes[1].set_title('Box Plot of Facebook Closing Prices')

plt.tight_layout()
```




```
# 4. Use axvspan() to shade a rectangle from '2018-07-25' to '2018- 07-31', which marks the large decline in Facebook price on a line plot
ax = fb['2018-07-25':'2018-07-31'][['close']].plot(kind='line', title='Large Decline in Facebook (Closing Price)')
ax.axvspan('2018-07-25', '2018-07-31', alpha=0.3, color='gray')
```

```
<matplotlib.patches.Polygon at 0x7f7a2aba5120>
```



```
# 5. Using the Facebook stock price data, annotate the following three events on a line plot of the closing price
  # Disappointing user growth announced after close on July 25, 2018
  # Cambridge Analytica story breaks on March 19, 2018 (when it affected the market)
  # FTC launches investigation on March 20, 2018

from datetime import datetime

# creating the line plot
plt.figure(figsize=(8, 4))
sns.lineplot(data=fb, x=fb.index, y='close')
plt.title('Facebook Stock Price Data')

# converting string to datetime
date_format = '%Y-%m-%d'
date1 = datetime.strptime('2018-07-25', date_format)
date2 = datetime.strptime('2018-07-26', date_format)
date3 = datetime.strptime('2018-03-19', date_format)
date4 = datetime.strptime('2018-03-20', date_format)
date5 = datetime.strptime('2018-03-21', date_format)

# annotating the 3 events
plt.annotate('Disappointing User Growth', xy=(date1, 180), xytext=(date2, 240),
             arrowprops=dict(facecolor='red', arrowstyle='->'), color='red', rotation=90)

plt.annotate('Cambridge Analytica Story', xy=(date3, 170), xytext=(date4, 220),
             arrowprops=dict(facecolor='green', arrowstyle='->'), color='green', rotation=90)

plt.annotate('FTC Launches Investigation', xy=(date4, 170), xytext=(date5, 200),
             arrowprops=dict(facecolor='blue', arrowstyle='->'), color='blue', rotation=80)

plt.xlabel('Date')
plt.ylabel('Closing Price')
```

```
Text(0, 0.5, 'Closing Price')
```


```
# 6. Modify the reg_resid_plots() function to use a matplotlib colormap instead of cycling between two colors.
# Remember, for this use case, we should pick a qualitative colormap or make our own.

# modified
import itertools

import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import colormaps # list of registered colormaps
import random

def reg_resid_plots(data):
    """
    Using seaborn, plot the regression and residuals
    plots side-by-side for every permutation of 2 columns
    in the data.

    Parameters:
        - data: A pandas DataFrame

    Returns:
        A matplotlib Figure object.
    """
```