Assignment

For this project you will visualize time series data using a line chart, bar chart, and box plots. You will use Pandas, Matplotlib, and Seaborn to visualize a dataset containing the number of page views each day on the freeCodeCamp.org forum from 2016-05-09 to 2019-12-03. The data visualizations will help you understand the patterns in visits and identify yearly and monthly growth.

Use the data to complete the following tasks:

- Use Pandas to import the data from "fcc-forum-pageviews.csv". Set the index to the "date" column.
- Clean the data by filtering out days when the page views were in the top 2.5% of the dataset or bottom 2.5% of the dataset.
- Create a `draw_line_plot` function that uses Matplotlib to draw a line chart similar to "examples/Figure_1.png". The title should be "Daily freeCodeCamp Forum Page Views 5/2016-12/2019". The label on the x axis should be "Date" and the label on the y axis should be "Page Views".
- Create a `draw_bar_plot` function that draws a bar chart similar to "examples/Figure_2.png". It should show average daily page views for each month grouped by year. The legend should show month labels and have a title of "Months". On the chart, the label on the x axis should be "Years" and the label on the y axis should be "Average Page Views".
- Create a `draw_box_plot` function that uses Searborn to draw two adjacent box plots similar to "examples/Figure_3.png". These box plots should show how the values are distributed within a given year or month and how it compares over time. The title of the first chart should be "Year-wise Box Plot (Trend)" and the title of the second chart should be "Month-wise Box Plot (Seasonality)". Make sure the month labels on bottom start at "Jan" and the x and x axis are labeled correctly.

For each chart, make sure to use a copy of the data frame. Unit tests are written for you under `test_module.py`.

Development

For development, you can use `main.py` to test your functions. Click the "run" button and `main.py` will run.

Testing

We imported the tests from `test_module.py` to `main.py` for your convenience. The tests will run automatically whenever you hit the "run" button.

Submitting

Copy your project's URL and submit it to freeCodeCamp.

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
```

In [3]:
```
os.getcwd()
```

Out[3]:
```
'C:\\Users\\ANAND\\FreeCodeCamp\\DA using python projects -Florin
Pop'
```

In [4]:
```
os.chdir('C:\\Users\\ANAND\\FreeCodeCamp\csv data')
```

In [30]:
```
df = pd.read_csv('fcc-forum-pageviews.csv', index_col = [0], parse_d
ates = True)
```

In [42]:
```
lt25 = (df['value']< df['value'].quantile(0.025))
gt97 = (df['value']> df['value'].quantile(0.975))
cond = (lt25|gt97)
df = df.drop(index = df[cond].index)
df = df.astype('int64')
```

In [43]:
```
df.dtypes
```

Out[43]:
```
value    int64
dtype: object
```

In [44]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1238 entries, 2016-05-19 to 2019-12-03
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   value   1238 non-null   int64
dtypes: int64(1)
memory usage: 19.3 KB
```

In [47]:
```
int(df.count(numeric_only=True))
```

Out[47]:
```
1238
```

Create a draw_line_plot function that uses Matplotlib to draw a line chart similar to "examples/Figure_1.png". The title should be "Daily freeCodeCamp Forum Page Views 5/2016-12/2019". The label on the x axis should be "Date" and the label on the y axis should be "Page Views".

In [ ]:
```
plt.figure(figsize = (10,5))
plt.plot(df.index, df['value'], color = 'r', linewidth = 1)
plt.title('Daily freeCodeCamp Forum Page Views 5/2016-12/2019')
plt.xlabel('Date')
plt.ylabel('Page Views')
```

Create a draw_bar_plot function that draws a bar chart similar to "examples/Figure_2.png". It should show average daily page views for each month grouped by year. The legend should show month labels and have a title of "Months". On the chart, the label on the x axis should be "Years" and the label on the y axis should be "Average Page Views" Referred to this article. https://medium.com/analytics-vidhya/create-a-grouped-bar-chart-with-matplotlib-and-pandas-9b021c97e0a 1. clean the data as instructed 2. extract month and year in separate columns. 3. Create a list of months 4. Replace month column's data with respective month's name. 5. Make the months column as categorical column so that it can be sorted in order.(don't need to sort the data frame separately) 6. Now create a dataframe containing pivot table type speadsheet using pandas pivot_table method.(It will create multiindexes (hierarchical indexes)-> Year-> month -> aggregate function for values variable. 7. plot the bar chart

In [ ]:
```
df['Year'] = df.index.year
df['Month'] = df.index.month
```

In [ ]:
```
months = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
```

In [ ]:
```
df['Month'] = df['Month'].apply(lambda data: months[data-1])
```

In [ ]:
```
#Make this a categorical column so it can be sorted by the order of values
df['Month'] = pd.Categorical(df['Month'], categories = months)
```

In [ ]:
```
df.head()
```

In [ ]:
```
#pivot_table ceatesa spreadsheet-style pivot table as a DataFrame.
#The levels in the pivot table will be stored in MultiIndex objects
(hierarchical indexes) on the index and columns of the result DataFr
ame.
df_pivot = pd.pivot_table(df, values ='value', index = 'Year',column
s ='Month', aggfunc='mean')
df_pivot
```

In [ ]:
```
fig = df_pivot.plot(kind = 'bar')
fig.set_xlabel('Years')
fig.set_ylabel('Average Page Views')
fig.legend(months, title = 'Months')
```

In [ ]:
```
#df['date'].groupby(df['date'].dt.year.rename('Year'),df['date'].dt.
month.rename('Month')).agg({'mean'}) #doesn't work
#df_mon_avg = df.groupby(df.index.to_period("M")).agg('mean')
```

In [ ]:
```
#df_mon_avg.groupby(df_mon_avg['Year']).value
```

Create a draw_box_plot function that uses Searborn to draw two adjacent box plots similar to "examples/Figure_3.png". These box plots should show how the values are distributed within a given year or month and how it compares over time. The title of the first chart should be "Year-wise Box Plot (Trend)" and the title of the second chart should be "Month-wise Box Plot (Seasonality)". Make sure the month labels on bottom start at "Jan" and the x and x axis are labeled correctly.

In [ ]:
```
sns.boxplot(x = df['Year'], y = df['value'] )
```

In [ ]:
```
#ax = sns.boxplot(x= df['Month'], y = df['value'] )
#fig = ax.get_figure()
#fig = plt.subplots(figsize=(10,7))
g = sns.boxplot( x= df['Month'] , y = df['value'] )
fig = g.figure
fig.set_size_inches(12,6)
```

In [ ]:
```
#both plots in the same chart
fig, (ax1, ax2) = plt.subplots(1,2, figsize = (10,6))
sns.boxplot(ax = ax1, x = df['Year'], y = df['value'] )
ax1.set(xlabel = 'Year', ylabel = 'Page Views', title = 'Year-wise B
ox Plot (Trend)')

#fig, ax = plt.subplots((1,1,2), figsize=(12,6))
sns.boxplot(ax = ax2, x= df['Month'] , y = df['value'] )
ax2.set(xlabel = 'Month', ylabel = 'Page Views', title = 'Month-wise
Box Plot (Seasonality)')
```

In [95]:
```
df.head()
```

Out[95]:

date	value
2016-05-19	19736
2016-05-26	18060
2016-05-27	19997
2016-05-28	19044
2016-05-29	20325

In [96]:
```
df_box = df.copy()
df_box.reset_index(inplace=True)
df_box['year'] = [d.year for d in df_box.date]
df_box['month'] = [d.strftime('%b') for d in df_box.date]
```

In [ ]:
```
fig, (ax1, ax2) = plt.subplots(1,2, figsize = (10,6))
sns.boxplot(ax = ax1, x = df_box['year'], y = df_box['value'] )
ax1.set(xlabel = 'Year', ylabel = 'Page Views', title = 'Year-wise B
ox Plot (Trend)')

#fig, ax = plt.subplots((1,1,2), figsize=(12,6))
sns.boxplot(ax = ax2, x= df_box['month'] , y = df_box['value'] )
ax2.set(xlabel = 'Month', ylabel = 'Page Views', title = 'Month-wise
Box Plot (Seasonality)')
```

In [74]:
```
df
```

Out[74]:

date	date	value	Year	Month
2016-05-19	2016-05-19	19736	2016	May
2016-05-26	2016-05-26	18060	2016	May
2016-05-27	2016-05-27	19997	2016	May
2016-05-28	2016-05-28	19044	2016	May
2016-05-29	2016-05-29	20325	2016	May
...	...	...	...	...
2019-11-24	2019-11-24	138875	2019	November
2019-11-29	2019-11-29	171584	2019	November
2019-11-30	2019-11-30	141161	2019	November
2019-12-01	2019-12-01	142918	2019	December
2019-12-03	2019-12-03	158549	2019	December

1238 rows × 4 columns

In [108]:
```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()

# Import data (Make sure to parse dates. Consider setting index colu
mn to 'date'.)
df = pd.read_csv('fcc-forum-pageviews.csv')

# Clean data
df = df.set_index(df['date'])
df.index = pd.to_datetime(df.index)
df = df[(df['value']>= df['value'].quantile(0.025)) & (df['value']<=
df['value'].quantile(0.975))]
df.drop(columns ='date', inplace = True)
df = df.astype(int)

def draw_line_plot():
    # Draw line plot
    df_line = df.copy()
    fig, ax = plt.subplots()
    plt.plot(df_line.index, df_line['value'], color = 'r', linewidth
= 1)
    plt.title('Daily freeCodeCamp Forum Page Views 5/2016-12/2019')
    plt.xlabel('Date')
    plt.ylabel('Page Views')
    plt.close()
    # Save image and return fig (don't change this part)
    fig.savefig('line_plot.png')
    return fig

def draw_bar_plot():
    # Copy and modify data for monthly bar plot
    df_br = df.copy()
    df_br['Year'] = df.index.year
    df_br['Month'] = df.index.month
    months = ['January', 'February', 'March', 'April', 'May', 'June
', 'July', 'August', 'September', 'October', 'November', 'December']
    df_br['Month'] = df_br['Month'].apply(lambda data: months[data-
1])
    df_br['Month'] = pd.Categorical(df_br['Month'], categories = mon
ths)

    df_bar = pd.pivot_table(df_br, values ='value', index = 'Year',c
olumns ='Month', aggfunc='mean')

    # Draw bar plot
    ax = df_bar.plot(kind = 'bar')
    ax.set_xlabel('Years')
    ax.set_ylabel('Average Page Views')
    ax.legend(months, title = 'Months')
    fig = ax.get_figure()
    fig.set_size_inches(7, 6)
    # Save image and return fig (don't change this part)
    plt.close()
    fig.savefig('bar_plot.png')
    return fig
```

```
def draw_box_plot():
    # Prepare data for box plots (this part is done!)
    df_box = df.copy()
    df_box.reset_index(inplace=True)
    df_box['year'] = [d.year for d in df_box.date]
    df_box['month'] = [d.strftime('%b') for d in df_box.date]

    # Draw box plots (using Seaborn)

    fig, (ax1, ax2) = plt.subplots(1,2, figsize = (10,6))
    sns.boxplot(ax = ax1, x = df_box['year'], y = df_box['value'] )
    ax1.set(xlabel = 'Year', ylabel = 'Page Views', title = 'Year-wise Box Plot (Trend)')

    #fig, ax = plt.subplots((1,1,2), figsize=(12,6))
    sns.boxplot(ax = ax2, x= df_box['month'] , y = df_box['value'], order =['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'] )
    ax2.set(xlabel = 'Month', ylabel = 'Page Views', title = 'Month-wise Box Plot (Seasonality)')

    # Save image and return fig (don't change this part)
    plt.close()
    fig.savefig('box_plot.png')
    return fig
```

In [109]:
```
draw_line_plot()
```

Out[109]: