Forecasting Net Prophet

You’re a growth analyst at MercadoLibre. With over 200 million users, MercadoLibre is the most popular e-commerce site in Latin America. You've been tasked with analyzing the company's financial and user data in clever ways to make the company grow. So, you want to find out if the ability to predict search traffic can translate into the ability to successfully trade the stock.

Instructions

This section divides the instructions for this Challenge into four steps and an optional fifth step, as follows:

- Step 1: Find unusual patterns in hourly Google search traffic
- Step 2: Mine the search traffic data for seasonality
- Step 3: Relate the search traffic to stock price patterns
- Step 4: Create a time series model with Prophet
- Step 5 (optional): Forecast revenue by using time series models

The following subsections detail these steps.

Step 1: Find Unusual Patterns in Hourly Google Search Traffic

The data science manager asks if the Google search traffic for the company links to any financial events at the company. Or, does the search traffic data just present random noise? To answer this question, pick out any unusual patterns in the Google search data for the company, and connect them to the corporate financial events.

To do so, complete the following steps:

1. Read the search data into a DataFrame, and then slice the data to just the month of May 2020. (During this month, MercadoLibre released its quarterly financial results.) Use hvPlot to visualize the results. Do any unusual patterns exist?
2. Calculate the total search traffic for the month, and then compare the value to the monthly median across all months. Did the Google search traffic increase during the month that MercadoLibre released its financial results?

Step 2: Mine the Search Traffic Data for Seasonality

Marketing realizes that they can use the hourly search data, too. If they can track and predict interest in the company and its platform for any time of day, they can focus their marketing efforts around the times that have the most traffic. This will get a greater return on investment (ROI) from their marketing budget.

To that end, you want to mine the search traffic data for predictable seasonal patterns of interest in the company. To do so, complete the following steps:

1. Group the hourly search data to plot the average traffic by the day of the week (for example, Monday vs. Friday).
2. Using hvPlot, visualize this traffic as a heatmap, referencing the `index.hour` as the x-axis and the `index.dayofweek` as the y-axis. Does any day-of-week effect that you observe concentrate in just a few hours of that day?
3. Group the search data by the week of the year. Does the search traffic tend to increase during the winter holiday period (weeks 40 through 52)?

Step 3: Relate the Search Traffic to Stock Price Patterns

You mention your work on the search traffic data during a meeting with people in the finance group at the company. They want to know if any relationship between the search data and the company stock price exists, and they ask if you can investigate.

To do so, complete the following steps:

1. Read in and plot the stock price data. Concatenate the stock price data to the search data in a single DataFrame.
2. Market events emerged during the year of 2020 that many companies found difficult. But, after the initial shock to global financial markets, new customers and revenue increased for e-commerce platforms. Slice the data to just the first half of 2020 (`2020-01` to `2020-06` in the DataFrame), and then use hvPlot to plot the data. Do both time series indicate a common trend that’s consistent with this narrative?
3. Create a new column in the DataFrame named “Lagged Search Trends” that offsets, or shifts, the search traffic by one hour. Create two additional columns:
    - “Stock Volatility”, which holds an exponentially weighted four-hour rolling average of the company’s stock volatility

  - “Hourly Stock Return”, which holds the percent change of the company's stock price on an hourly basis

4. Review the time series correlation, and then answer the following question: Does a predictable relationship exist between the lagged search traffic and the stock volatility or between the lagged search traffic and the stock price returns?

Step 4: Create a Time Series Model with Prophet

Now, you need to produce a time series model that analyzes and forecasts patterns in the hourly search data. To do so, complete the following steps:

1. Set up the Google search data for a Prophet forecasting model.
2. After estimating the model, plot the forecast. How's the near-term forecast for the popularity of MercadoLibre?
3. Plot the individual time series components of the model to answer the following questions:
   - What time of day exhibits the greatest popularity?
   - Which day of the week gets the most search traffic?
   - What's the lowest point for search traffic in the calendar year?

Step 5 (Optional): Forecast Revenue by Using Time Series Models

A few weeks after your initial analysis, the finance group follows up to find out if you can help them solve a different problem. Your fame as a growth analyst in the company continues to grow!

Specifically, the finance group wants a forecast of the total sales for the next quarter. This will dramatically increase their ability to plan budgets and to help guide expectations for the company investors.

To do so, complete the following steps:

1. Read in the daily historical sales (that is, revenue) figures, and then apply a Prophet model to the data.
2. Interpret the model output to identify any seasonal patterns in the company's revenue. For example, what are the peak revenue days? (Mondays? Fridays? Something else?)
3. Produce a sales forecast for the finance group. Give them a number for the expected total sales in the next quarter. Include the best- and worst-case scenarios to help them make better plans.

Install and import the required libraries and dependencies

```
# Install the required libraries
!pip install pystan
!pip install prophet
!pip install hvplot
!pip install holoviews
```

```
    Requirement already satisfied: packaging in /usr/local/lib/python3.9/dist-packages (from holoviews) (23.0)
    Requirement already satisfied: panel>=0.8.0 in /usr/local/lib/python3.9/dist-packages (from holoviews) (0.14.4)
    Requirement already satisfied: pandas>=0.20.0 in /usr/local/lib/python3.9/dist-packages (from holoviews) (1.4.4)
    Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.9/dist-packages (from holoviews) (2.2.1)
    Requirement already satisfied: colorcet in /usr/local/lib/python3.9/dist-packages (from holoviews) (3.0.1)
    Requirement already satisfied: param<2.0,>=1.9.3 in /usr/local/lib/python3.9/dist-packages (from holoviews) (1.13.0)
    Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packages (from pandas>=0.20.0->holoviews) (2022.7.1)
    Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.9/dist-packages (from pandas>=0.20.0->holoviews) (2.
    Requirement already satisfied: bleach in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (6.0.0)
    Requirement already satisfied: markdown in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (3.4.1)
    Requirement already satisfied: pyct>=0.4.4 in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (0.5.0)
    Requirement already satisfied: setuptools>=42 in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (63.4.3)
    Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (4.65.0)
    Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (2.25.1)
    Requirement already satisfied: typing-extensions in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (4.5.0)
    Requirement already satisfied: bokeh<2.5.0,>=2.4.0 in /usr/local/lib/python3.9/dist-packages (from panel>=0.8.0->holoviews) (2.4.3)
    Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.9/dist-packages (from bokeh<2.5.0,>=2.4.0->panel>=0.8.0->holovie
    Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.9/dist-packages (from bokeh<2.5.0,>=2.4.0->panel>=0.8.0->holov
    Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.9/dist-packages (from bokeh<2.5.0,>=2.4.0->panel>=0.8.0->holov
    Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.9/dist-packages (from bokeh<2.5.0,>=2.4.0->panel>=0.8.0->holo
    Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (from python-dateutil>=2.8.1->pandas>=0.20.0->holo
    Requirement already satisfied: webencodings in /usr/local/lib/python3.9/dist-packages (from bleach->panel>=0.8.0->holoviews) (0.5.1)
    Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.9/dist-packages (from markdown->panel>=0.8.0->holov
    Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests->panel>=0.8.0->holovie
    Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.9/dist-packages (from requests->panel>=0.8.0->holoviews)
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests->panel>=0.8.0->holoviews)
    Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests->panel>=0.8.0->holoviews) (2.10
```

```
# Import the required libraries and dependencies
import pandas as pd
import hvplot.pandas
import datetime as dt
import holoviews as hv
from prophet import Prophet
import numpy as np
%matplotlib inline
```

Step 1: Find Unusual Patterns in Hourly Google Search Traffic

The data science manager asks if the Google search traffic for the company links to any financial events at the company. Or, does the search traffic data just present random noise? To answer this question, pick out any unusual patterns in the Google search data for the company, and connect them to the corporate financial events.

To do so, complete the following steps:

1. Read the search data into a DataFrame, and then slice the data to just the month of May 2020. (During this month, MercadoLibre released its quarterly financial results.) Use hvPlot to visualize the results. Do any unusual patterns exist?
2. Calculate the total search traffic for the month, and then compare the value to the monthly median across all months. Did the Google search traffic increase during the month that MercadoLibre released its financial results?

Step 1: Read the search data into a DataFrame, and then slice the data to just the month of May 2020. (During this month, MercadoLibre released its quarterly financial results.) Use hvPlot to visualize the results. Do any unusual patterns exist?

```
# Upload the "google_hourly_search_trends.csv" file into Colab, then store in a Pandas DataFrame
# Set the "Date" column as the Datetime Index.

from google.colab import files
uploaded = files.upload()
```

Choose Files google_hou…_trends.csv

- **google_hourly_search_trends.csv**(text/csv) - 645970 bytes, last modified: 3/10/2023 - 100% done

```
Saving google_hourly_search_trends.csv to google_hourly_search_trends (1).csv
```

```
df_mercado_trends= pd.read_csv(
    "google_hourly_search_trends.csv",
    index_col='Date',
    parse_dates=True,
    infer_datetime_format=True
).dropna()
```

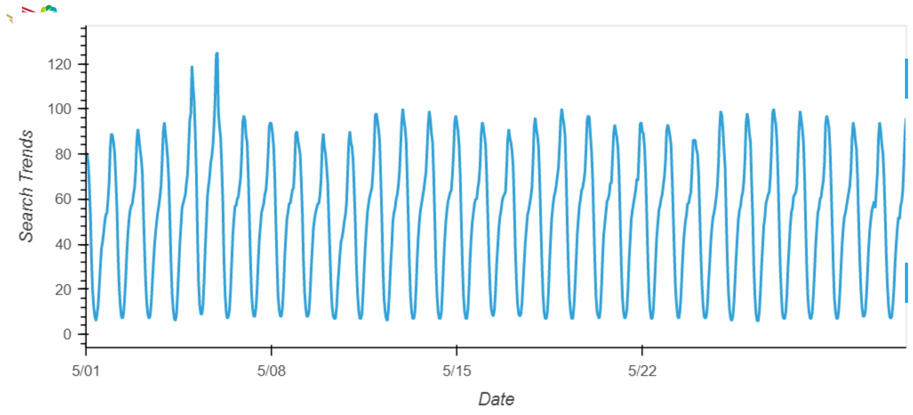
```
# Review the first and last five rows of the DataFrame
df_mercado_trends.head()
```

	Search Trends
Date	
2016-06-01 00:00:00	97
2016-06-01 01:00:00	92
2016-06-01 02:00:00	76
2016-06-01 03:00:00	60
2016-06-01 04:00:00	38

```
# Review the data types of the DataFrame using the info function
df_mercado_trends.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 37106 entries, 2016-06-01 00:00:00 to 2020-09-08 00:00:00
Data columns (total 1 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Search Trends  37106 non-null  int64
dtypes: int64(1)
memory usage: 579.8 KB
```

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')
# Slice the DataFrame to just the month of May 2020
df_may_2020 = df_mercado_trends.loc['2020-05-01':'2020-5-31']
# Use hvPlot to visualize the data for May 2020
df_may_2020.hvplot()
```



Step 2: Calculate the total search traffic for the month, and then compare the value to the monthly median across all months. Did the Google search traffic increase during the month that MercadoLibre released its financial results?

```
# Calculate the sum of the total search traffic for May 2020
traffic_may_2020 = df_may_2020.sum()

# View the traffic_may_2020 value
print(f"Total search traffic in May 2020: {traffic_may_2020}")
```

```
Total search traffic in May 2020: Search Trends    38181
dtype: int64
```

```
# Calcluate the monhtly median search traffic across all months
# Group the DataFrame by index year and then index month, chain the sum and then the median functions
median_monthly_traffic = df_mercado_trends.groupby(by=[df_mercado_trends.index.year,df_mercado_trends.index.month]).sum().median()
# View the median_monthly_traffic value
print(f"Monthly Median traffic tend in May 2020: {median_monthly_traffic}")
```

```
Monthly Median traffic tend in May 2020: Search Trends    35172.5
dtype: float64
```

```
# Compare the seach traffic for the month of May 2020 to the overall monthly median value
traffic_may_2020/median_monthly_traffic

     Search Trends    1.085536
     dtype: float64
```

Answer the following question:

Question: Did the Google search traffic increase during the month that MercadoLibre released its financial results?

Answer: The total search traffic for May 2020 is 38181, which is higher than the monthly median search traffic of 35172. This indicates that there was an increase in search traffic during the month that MercadoLibre released its financial results.

Step 2: Mine the Search Traffic Data for Seasonality

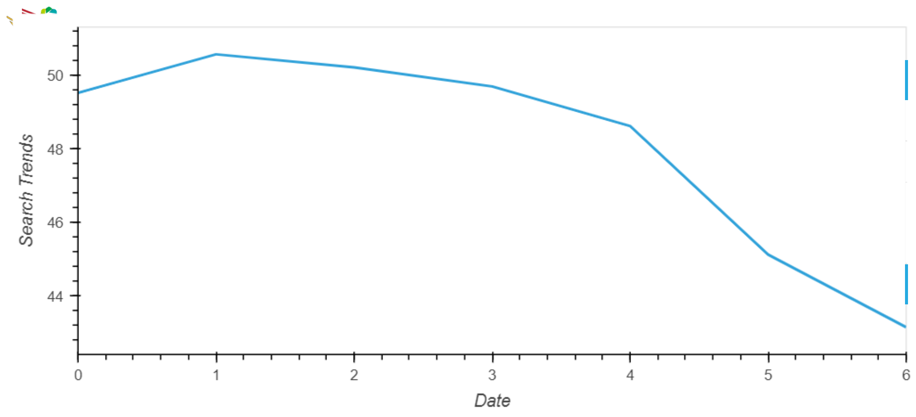
Marketing realizes that they can use the hourly search data, too. If they can track and predict interest in the company and its platform for any time of day, they can focus their marketing efforts around the times that have the most traffic. This will get a greater return on investment (ROI) from their marketing budget.

To that end, you want to mine the search traffic data for predictable seasonal patterns of interest in the company. To do so, complete the following steps:

1. Group the hourly search data to plot the average traffic by the day of the week (for example, Monday vs. Friday).
2. Using hvPlot, visualize this traffic as a heatmap, referencing the `index.hour` as the x-axis and the `index.dayofweek` as the y-axis. Does any day-of-week effect that you observe concentrate in just a few hours of that day?
3. Group the search data by the week of the year. Does the search traffic tend to increase during the winter holiday period (weeks 40 through 52)?

Step 1: Group the hourly search data to plot the average traffic by the day of the week (for example, Monday vs. Friday).

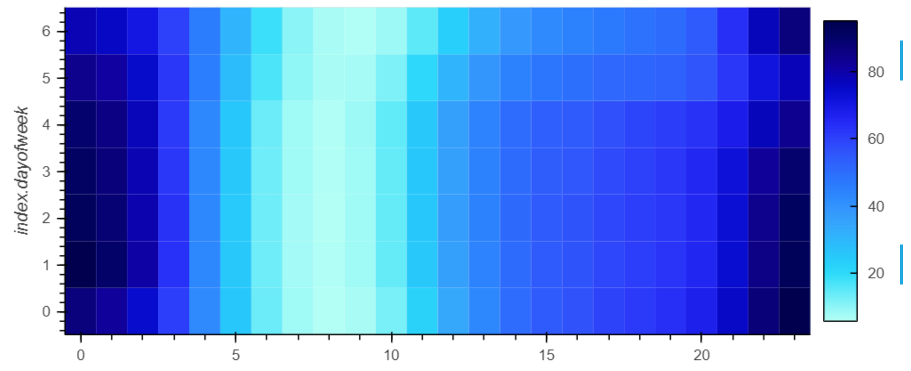
```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Group the hourly search data to plot (use hvPlot) the average traffic by the day of week
df_mercado_trends_plot = df_mercado_trends.groupby(df_mercado_trends.index.dayofweek).mean().hvplot()
df_mercado_trends_plot
```



Step 2: Using hvPlot, visualize this traffic as a heatmap, referencing the `index.hour` as the x-axis and the `index.dayofweek` as the y-axis. Does any day-of-week effect that you observe concentrate in just a few hours of that day?

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the hour of the day and day of week search traffic as a heatmap.
df_mercado_trends.hvplot.heatmap(x='index.hour', y='index.dayofweek',C='Search Trends', cmaps='reds').aggregate(function=np.mean)
```

```
WARNING:param.main: cmaps option not found for heatmap plot with bokeh; similar options include: [
WARNING:param.main:cmaps option not found for heatmap plot with bokeh; similar options include: ['
```



Answer the following question:

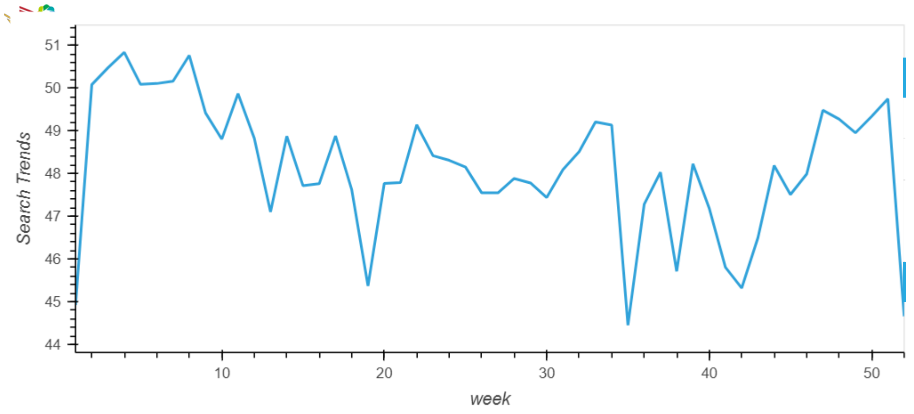
Question: Does any day-of-week effect that you observe concentrate in just a few hours of that day?

Answer: # The heatmap can tell us any specific hours of the day that show a concentration of search traffic on a particular day of the week. If we see areas of the heatmap with the darkest color. For example, if you see a darker color for Monday at 9 am, it means that the search traffic tends to peak around that time on Mondays which "0" in the graph.

Weekends seems traffic seems to decrease as shown by the cooler region on the heat map.

Step 3: Group the search data by the week of the year. Does the search traffic tend to increase during the winter holiday period (weeks 40 through 52)?

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Group the hourly search data to plot (use hvPlot) the average traffic by the week of the year
df_mercado_trends.groupby(df_mercado_trends.index.isocalendar().week).mean().hvplot()
```



Answer the following question:

Question: Does the search traffic tend to increase during the winter holiday period (weeks 40 through 52)?

Answer: This graph gives an average search traffic for weeks 40 through 52 and for the rest of the yea. You can then compare the two averages to see if there is a noticeable difference in search traffic during the winter holiday period. If the average search traffic during weeks 40 through 52 is significantly higher than the rest of the year, it suggests that there is an increase in search traffic during the winter holiday period. The winter traffic increases from 44 and reaches the peak on the 51st week of the year. This tells that people are going out for shopping.

Step 3: Relate the Search Traffic to Stock Price Patterns

You mention your work on the search traffic data during a meeting with people in the finance group at the company. They want to know if any relationship between the search data and the company stock price exists, and they ask if you can investigate.

To do so, complete the following steps:

1. Read in and plot the stock price data. Concatenate the stock price data to the search data in a single DataFrame.
2. Market events emerged during the year of 2020 that many companies found difficult. But, after the initial shock to global financial markets, new customers and revenue increased for e-commerce platforms. Slice the data to just the first half of 2020 (`2020-01` to `2020-06` in the DataFrame), and then use hvPlot to plot the data. Do both time series indicate a common trend that's consistent with this narrative?
3. Create a new column in the DataFrame named "Lagged Search Trends" that offsets, or shifts, the search traffic by one hour. Create two additional columns:
    - "Stock Volatility", which holds an exponentially weighted four-hour rolling average of the company's stock volatility
    - "Hourly Stock Return", which holds the percent change of the company's stock price on an hourly basis
4. Review the time series correlation, and then answer the following question: Does a predictable relationship exist between the lagged search traffic and the stock volatility or between the lagged search traffic and the stock price returns?

Step 1: Read in and plot the stock price data. Concatenate the stock price data to the search data in a single DataFrame.

```
# Upload the "mercado_stock_price.csv" file into Colab, then store in a Pandas DataFrame
# Set the "date" column as the Datetime Index.
from google.colab import files
uploaded = files.upload()
```

Choose Files mercado_stock_price.csv

- **mercado_stock_price.csv**(text/csv) - 1082540 bytes, last modified: 3/10/2023 - 100% done

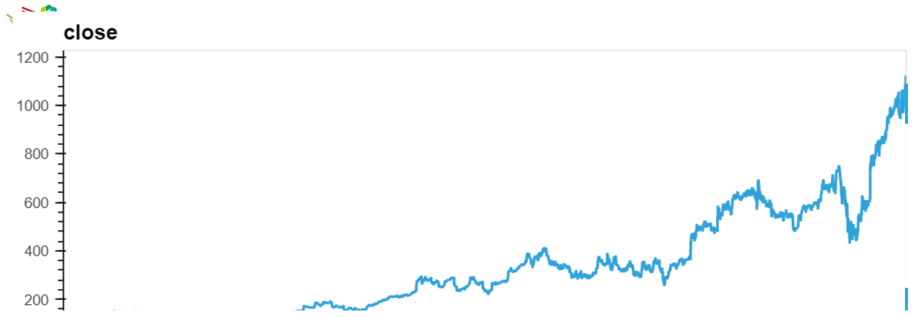
```
Saving mercado_stock_price.csv to mercado_stock_price (2).csv
```

```
df_mercado_stock = stock_prices = pd.read_csv('mercado_stock_price.csv', index_col='date', parse_dates=True).dropna()

# View the first and last five rows of the DataFrame
df_mercado_stock
```

	close
date	
2015-01-02 09:00:00	127.670
2015-01-02 10:00:00	125.440
2015-01-02 11:00:00	125.570
2015-01-02 12:00:00	125.400
2015-01-02 13:00:00	125.170
...	...
2020-07-31 11:00:00	1105.780
2020-07-31 12:00:00	1087.925
2020-07-31 13:00:00	1095.800
2020-07-31 14:00:00	1110.650
2020-07-31 15:00:00	1122.510

9336 rows × 1 columns

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the closing price of the df_mercado_stock DataFrame
df_mercado_stock['close'].hvplot()
```



```
# Concatenate the df_mercado_stock DataFrame with the df_mercado_trends DataFrame
# Concatenate the DataFrame by columns (axis=1), and drop and rows with only one column of data
mercado_stock_trends_df = pd.concat([df_mercado_trends, df_mercado_stock], axis=1).dropna()


# View the first and last five rows of the DataFrame
mercado_stock_trends_df.head()
```

	Search Trends	close
2016-06-01 09:00:00	6.0	135.16
2016-06-01 10:00:00	12.0	136.63
2016-06-01 11:00:00	22.0	136.56
2016-06-01 12:00:00	33.0	136.42
2016-06-01 13:00:00	40.0	136.10

Step 2: Market events emerged during the year of 2020 that many companies found difficult. But, after the initial shock to global financial markets, new customers and revenue increased for e-commerce platforms. Slice the data to just the first half of 2020 (`2020-01` to `2020-06` in the DataFrame), and then use hvPlot to plot the data. Do both time series indicate a common trend that's consistent with this narrative?

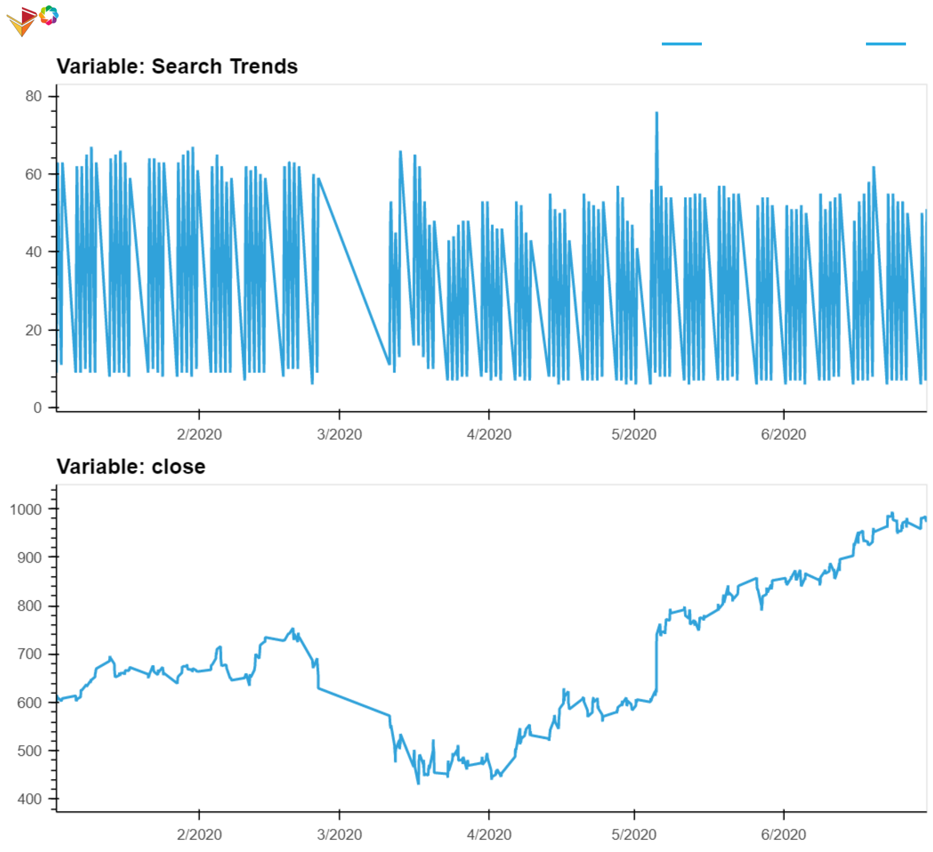
```
# For the combined dataframe, slice to just the first half of 2020 (2020-01 through 2020-06)
first_half_2020 = mercado_stock_trends_df.loc['2020-01-01':'2020-06-30']

# View the first and last five rows of first_half_2020 DataFrame
first_half_2020.head()
```

	Search Trends	close
2020-01-02 09:00:00	9.0	601.085
2020-01-02 10:00:00	14.0	601.290
2020-01-02 11:00:00	25.0	615.410
2020-01-02 12:00:00	37.0	611.400
2020-01-02 13:00:00	50.0	611.830

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the close and Search Trends data
# Plot each column on a separate axes using the following syntax
# `hvplot(shared_axes=False, subplots=True).cols(1)`
first_half_2020.hvplot(y=['Search Trends', 'close'], title='Search Trends and Stock Price (Jan-Jun 2020)',
                       xlabel='Date', ylabel='Search Trends / Stock Price')
```

Search Trends and Stock Price (Jan-Jun 2020)
ds / Stock Price
1000
800
600

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')
first_half_2020.hvplot(shared_axes=False,subplots=True).cols(1)
```



Answer the following question:

Question: Do both time series indicate a common trend that's consistent with this narrative?

Answer: You can then visually inspect the plot to see if there is a common trend that's consistent with the search trend and stock activity.

There seems to be lower search activity during mid of march and the stock price was lowest during mid of march to begining of april. Also search trend spiked on may 5th before the increase in the stock prices. This indicates that people are interested in the earnings of the company.

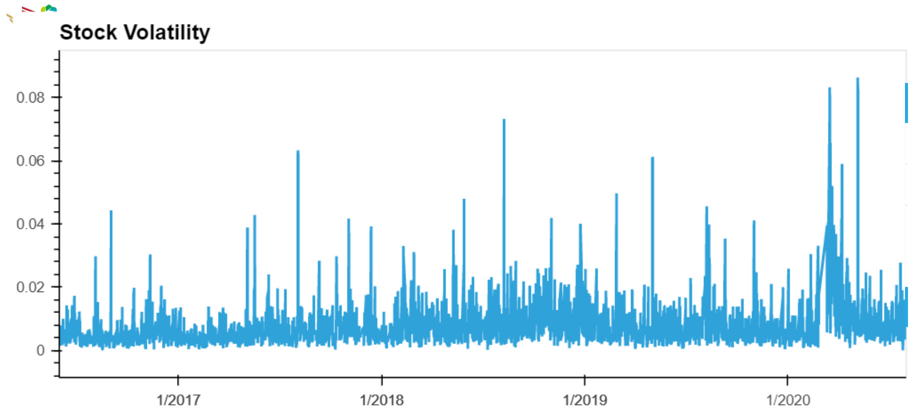
Step 3: Create a new column in the DataFrame named "Lagged Search Trends" that offsets, or shifts, the search traffic by one hour. Create two additional columns:

- "Stock Volatility", which holds an exponentially weighted four-hour rolling average of the company's stock volatility
- "Hourly Stock Return", which holds the percent change of the company's stock price on an hourly basis

```
# Create a new column in the mercado_stock_trends_df DataFrame called Lagged Search Trends
# This column should shift the Search Trends information by one hour
mercado_stock_trends_df['Lagged Search Trends'] = mercado_stock_trends_df['Search Trends'].shift(1)
```

```
# Create a new column in the mercado_stock_trends_df DataFrame called Stock Volatility
# This column should calculate the standard deviation of the closing stock price return data over a 4 period rolling window
mercado_stock_trends_df['Stock Volatility'] = mercado_stock_trends_df['close'].pct_change().rolling(window=4).std()
```

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the stock volatility
mercado_stock_trends_df['Stock Volatility'].hvplot()
```



Solution Note: Note how volatility spiked, and tended to stay high, during the first half of 2020. This is a common characteristic of volatility in stock returns worldwide: high volatility days tend to be followed by yet more high volatility days. When it rains, it pours.

```
# Create a new column in the mercado_stock_trends_df DataFrame called Hourly Stock Return
# This column should calculate hourly return percentage of the closing price
mercado_stock_trends_df['Hourly Stock Return'] = mercado_stock_trends_df['close'].pct_change()
```

```
# View the first and last five rows of the mercado_stock_trends_df DataFrame
mercado_stock_trends_df
```

	Search Trends	close	Lagged Search Trends	Stock Volatility	Hourly Stock Return
2016-06-01 09:00:00	6.0	135.160	NaN	NaN	NaN
2016-06-01 10:00:00	12.0	136.630	6.0	NaN	0.010876
2016-06-01 11:00:00	22.0	136.560	12.0	NaN	-0.000512
2016-06-01 12:00:00	33.0	136.420	22.0	NaN	-0.001025
2016-06-01 13:00:00	40.0	136.100	33.0	0.006134	-0.002346
...	...	...	...	...	...
2020-07-31 11:00:00	20.0	1105.780	11.0	0.012837	0.006380
2020-07-31 12:00:00	32.0	1087.925	20.0	0.013549	-0.016147

Step 4: Review the time series correlation, and then answer the following question: Does a predictable relationship exist between the lagged search traffic and the stock volatility or between the lagged search traffic and the stock price returns?

```
# Construct correlation table of Stock Volatility, Lagged Search Trends, and Hourly Stock Return
mercado_stock_trends_df[['Stock Volatility','Lagged Search Trends','Hourly Stock Return']].corr()
```

	Stock Volatility	Lagged Search Trends	Hourly Stock Return
Stock Volatility	1.000000	-0.148938	0.061424
Lagged Search Trends	-0.148938	1.000000	0.017929
Hourly Stock Return	0.061424	0.017929	1.000000

Answer the following question:

Question: Does a predictable relationship exist between the lagged search traffic and the stock volatility or between the lagged search traffic and the stock price returns?

Answer: There is a negative correlation shown between lagged search trend and stock volatility. There is a positve correlation shown between lagged search trends and Hourly stock returns. It seems like if search goes up then the stock prices goes up.

Step 4: Create a Time Series Model with Prophet

Now, you need to produce a time series model that analyzes and forecasts patterns in the hourly search data. To do so, complete the following steps:

1. Set up the Google search data for a Prophet forecasting model.
2. After estimating the model, plot the forecast. How's the near-term forecast for the popularity of MercadoLibre?
3. Plot the individual time series components of the model to answer the following questions:
    - What time of day exhibits the greatest popularity?
    - Which day of the week gets the most search traffic?
    - What's the lowest point for search traffic in the calendar year?

Step 1: Set up the Google search data for a Prophet forecasting model.

```
# Using the df_mercado_trends DataFrame, reset the index so the date information is no longer the index
mercado_prophet_df = df_mercado_trends.reset_index()

# Label the columns ds and y so that the syntax is recognized by Prophet
mercado_prophet_df.columns = ['ds','y']

# Drop an NaN values from the prophet_df DataFrame
mercado_prophet_df = mercado_prophet_df.dropna()

# View the first and last five rows of the mercado_prophet_df DataFrame
mercado_prophet_df.head()
```

	ds	y
0	2016-06-01 00:00:00	97
1	2016-06-01 01:00:00	92
2	2016-06-01 02:00:00	76
3	2016-06-01 03:00:00	60
4	2016-06-01 04:00:00	38

```
# Call the Prophet function, store as an object
model_mercado_trends = Prophet()
model_mercado_trends
```

```
<prophet.forecaster.Prophet at 0x7f35e05dcf70>
```

```
# Fit the time-series model.
model_mercado_trends.fit(mercado_prophet_df)
```

```
DEBUG:cmdstanpy:input tempfile: /tmp/tmp6i_xm00e/mmr9nlxc.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmp6i_xm00e/fs4j8eku.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.9/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=26147', 'c
01:28:31 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
01:29:07 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f35e05dcf70>
```

```
# Create a future dataframe to hold predictions
# Make the prediction go out as far as 2000 hours (approx 80 days)
future_mercado_trends = model_mercado_trends.make_future_dataframe(periods=2000, freq='H')

# View the last five rows of the future_mercado_trends DataFrame
future_mercado_trends.tail()
```

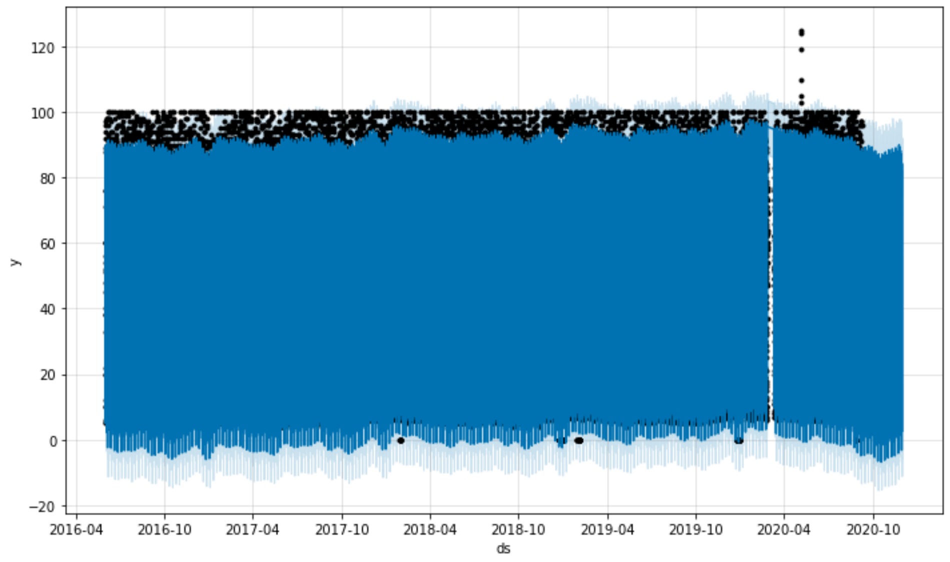
	ds
39101	2020-11-30 04:00:00
39102	2020-11-30 05:00:00
39103	2020-11-30 06:00:00
39104	2020-11-30 07:00:00
39105	2020-11-30 08:00:00

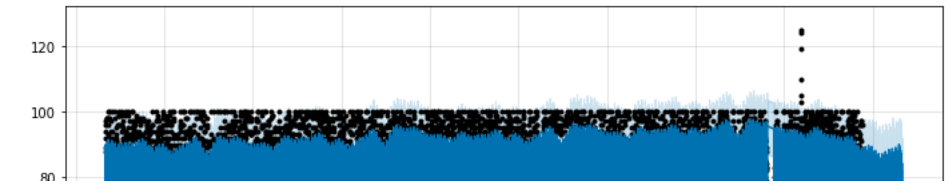
```
# Make the predictions for the trend data using the future_mercado_trends DataFrame
forecast_mercado_trends = model_mercado_trends.predict(future_mercado_trends)

# Display the first five rows of the forecast_mercado_trends DataFrame
forecast_mercado_trends.head()
```

ive_terms_upper	daily	...	weekly	weekly_lower	weekly_upper	yearly	yearly_low
45.290820	41.452721	...	1.860346	1.860346	1.860346	1.977752	1.9777
41.736647	37.943554	...	1.810263	1.810263	1.810263	1.982830	1.9828
31.413187	27.656623	...	1.768689	1.768689	1.768689	1.987876	1.9878
16.145999	12.417392	...	1.735716	1.735716	1.735716	1.992891	1.9928
-0.968848	-4.678022	...	1.711300	1.711300	1.711300	1.997874	1.9978

Step 2: After estimating the model, plot the forecast. How's the near-term forecast for the popularity of MercadoLibre?

```
# Plot the Prophet predictions for the Mercado trends data
model_mercado_trends.plot(forecast_mercado_trends)
```





Answer the following question:

Question: How's the near-term forecast for the popularity of MercadoLibre?

Answer: The near-term forecast shows that the search acitivty for 2020 remain steady.

Step 3: Plot the individual time series components of the model to answer the following questions:

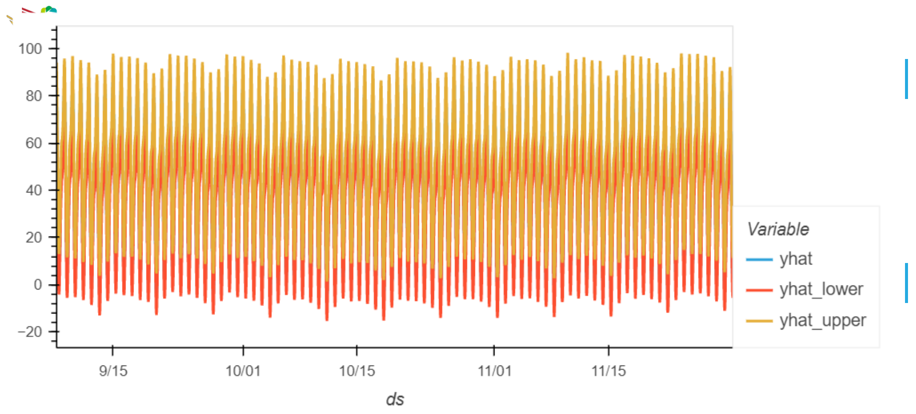
- What time of day exhibits the greatest popularity?
- Which day of the week gets the most search traffic?
- What's the lowest point for search traffic in the calendar year?

```
# Set the index in the forecast_mercado_trends DataFrame to the ds datetime column
forecast_mercado_trends = forecast_mercado_trends.set_index('ds')

# View the only the yhat,yhat_lower and yhat_upper columns from the DataFrame
forecast_mercado_trends[['yhat','yhat_lower','yhat_upper']].head()
```

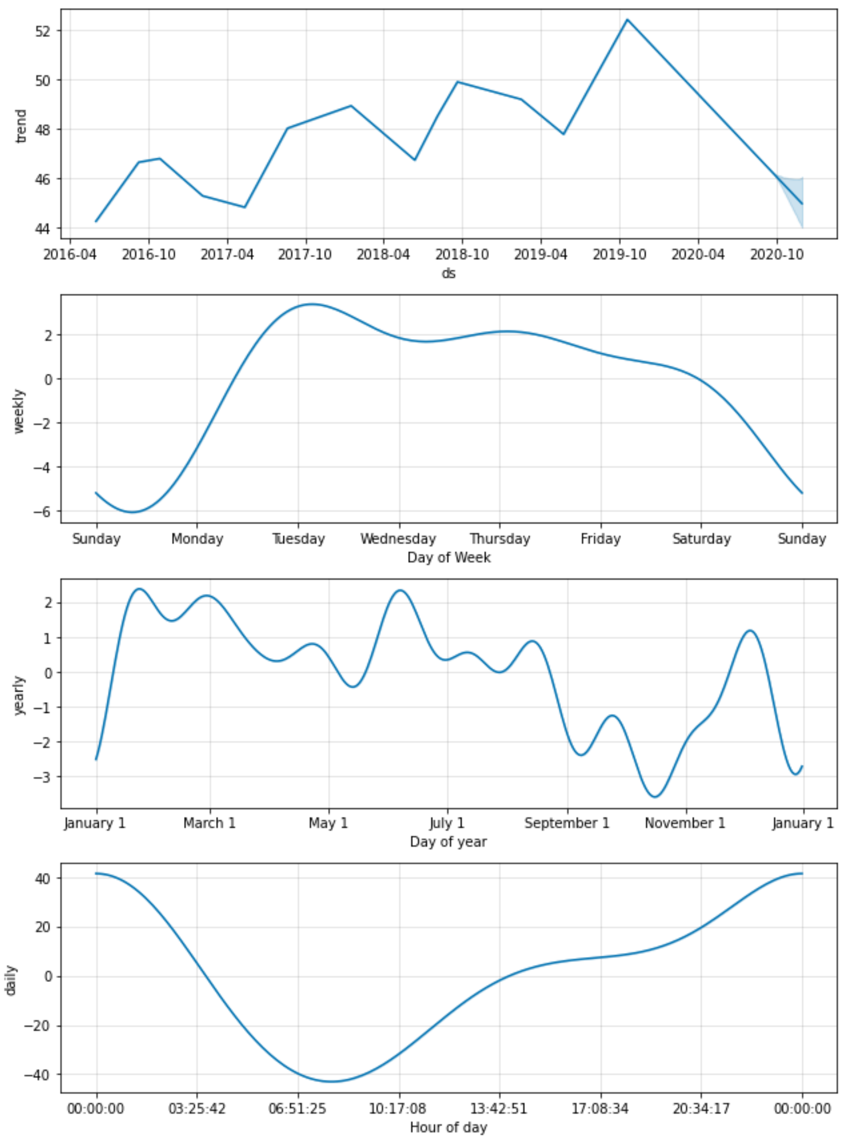
	yhat	yhat_lower	yhat_upper
ds			
2016-06-01 00:00:00	89.564320	81.276052	98.100035
2016-06-01 01:00:00	86.011152	77.796224	94.596162
2016-06-01 02:00:00	75.688698	67.011331	83.723757
2016-06-01 03:00:00	60.422515	51.813209	68.839246
2016-06-01 04:00:00	43.308673	35.345291	50.874286

Solutions Note: `yhat` represents the most likely (average) forecast, whereas `yhat_lower` and `yhat_upper` represents the worst and best case prediction (based on what are known as 95% confidence intervals).

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# From the forecast_mercado_trends DataFrame, use hvPlot to visualize
#  the yhat, yhat_lower, and yhat_upper columns over the last 2000 hours
forecast_mercado_trends[['yhat','yhat_lower','yhat_upper']].iloc[-2000:,:].hvplot()
```



```
# Reset the index in the forecast_mercado_trends DataFrame
forecast_mercado_trends = forecast_mercado_trends.reset_index()
# Use the plot_components function to visualize the forecast results
# for the forecast_canada DataFrame
figures_mercado_trends = model_mercado_trends.plot_components(forecast_mercado_trends)
```



Answer the following questions:

Question: What time of day exhibits the greatest popularity?

Answer: Late hours towards the midnight exhibits the greated popularity

Question: Which day of week gets the most search traffic?

Answer: From the graph it looks like Tuesday gets the most search traffic

Question: What's the lowest point for search traffic in the calendar year?

Answer: The graph shows mid of Oct and end of the year woth lowest point for search traffic due to holidays

Step 5 (Optional): Forecast Revenue by Using Time Series Models

A few weeks after your initial analysis, the finance group follows up to find out if you can help them solve a different problem. Your fame as a growth analyst in the company continues to grow!

Specifically, the finance group wants a forecast of the total sales for the next quarter. This will dramatically increase their ability to plan budgets and to help guide expectations for the company investors.

To do so, complete the following steps:

1. Read in the daily historical sales (that is, revenue) figures, and then apply a Prophet model to the data. The daily sales figures are quoted in millions of USD dollars.
2. Interpret the model output to identify any seasonal patterns in the company's revenue. For example, what are the peak revenue days? (Mondays? Fridays? Something else?)
3. Produce a sales forecast for the finance group. Give them a number for the expected total sales in the next quarter. Include the best- and worst-case scenarios to help them make better plans.

Step 1: Read in the daily historical sales (that is, revenue) figures, and then apply a Prophet model to the data.

```
# Upload the "mercado_daily_revenue.csv" file into Colab, then store in a Pandas DataFrame
# Set the "date" column as the DatetimeIndex
# Sales are quoted in millions of US dollars
from google.colab import files
uploaded = files.upload()

df_mercado_sales = # YOUR CODE HERE

# Review the DataFrame
# YOUR CODE HERE
```

```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the daily sales figures
# YOUR CODE HERE
```

```
# Apply a Facebook Prophet model to the data.

# Set up the dataframe in the neccessary format:
# Reset the index so that date becomes a column in the DataFrame
mercado_sales_prophet_df = # YOUR CODE HERE

# Adjust the columns names to the Prophet syntax
mercado_sales_prophet_df.columns = # YOUR CODE HERE

# Visualize the DataFrame
# YOUR CODE HERE
# Create the model
mercado_sales_prophet_model = # YOUR CODE HERE

# Fit the model
# YOUR CODE HERE
```

```
# Predict sales for 90 days (1 quarter) out into the future.

# Start by making a future dataframe
mercado_sales_prophet_future = # YOUR CODE HERE

# Display the last five rows of the future DataFrame
# YOUR CODE HERE
```

```
# Make predictions for the sales each day over the next quarter
mercado_sales_prophet_forecast = # YOUR CODE HERE

# Display the first 5 rows of the resulting DataFrame
# YOUR CODE HERE
```

Step 2: Interpret the model output to identify any seasonal patterns in the company's revenue. For example, what are the peak revenue days? (Mondays? Fridays? Something else?)

```
# Use the plot_components function to analyze seasonal patterns in the company's revenue
# YOUR CODE HERE
```

Answer the following question:

Question: For example, what are the peak revenue days? (Mondays? Fridays? Something else?)

Answer: # YOUR ANSWER HERE

Step 3: Produce a sales forecast for the finance group. Give them a number for the expected total sales in the next quarter. Include the best- and worst-case scenarios to help them make better plans.

```
# Plot the predictions for the Mercado sales
# YOUR CODE HERE
```

```
# For the mercado_sales_prophet_forecast DataFrame, set the ds column as the DataFrame Index
mercado_sales_prophet_forecast = # YOUR CODE HERE

# Display the first and last five rows of the DataFrame
# YOUR CODE HERE
```

```
# Produce a sales forecast for the finance division
# giving them a number for expected total sales next quarter.
# Provide best case (yhat_upper), worst case (yhat_lower), and most likely (yhat) scenarios.

# Create a forecast_quarter Dataframe for the period 2020-07-01 to 2020-09-30
# The DataFrame should include the columns yhat_upper, yhat_lower, and yhat
mercado_sales_forecast_quarter = # YOUR CODE HERE

# Update the column names for the forecast_quarter DataFrame
# to match what the finance division is looking for
mercado_sales_forecast_quarter = # YOUR CODE HERE

# Review the last five rows of the DataFrame
# YOUR CODE HERE
# Displayed the summed values for all the rows in the forecast_quarter DataFrame
# YOUR CODE HERE
```

Based on the forecast information generated above, produce a sales forecast for the finance division, giving them a number for expected total sales next quarter. Include best and worst case scenarios, to better help the finance team plan.

Answer: # YOUR ANSWER HERE

✓ 4s completed at 8:59 PM