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 fbprophet
```

```
!pip install hvplot
!pip install holoviews
```

```
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: typing-extensions>=3.7.4 in /usr/local/lib/python
Requirement already satisfied: param>=1.7.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: pyct>=0.4.4 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: panel>=0.8.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.7/di
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: bleach in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: markdown in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: webencodings in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/l
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/di
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dis
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-pac
Installing collected packages: hvplot
Successfully installed hvplot-0.8.0
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
Requirement already satisfied: holoviews in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: param<2.0,>=1.9.3 in /usr/local/lib/python3.7/dis
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: numpy>=1.0 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: colorcet in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: panel>=0.8.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: pandas>=0.20.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.7/di
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: bokeh<2.4.0,>=2.3.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: bleach in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: markdown in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: pyct>=0.4.4 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: typing-extensions>=3.7.4 in /usr/local/lib/python
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/dist
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: webencodings in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packag
```

```
    Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dis
    Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-pacl
    Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/l
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/di
```

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

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
# Set the "Date" column as the Datetime Index.

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


df_mercado_trends = pd.read_csv(
    "google_hourly_search_trends.csv",
```

```
    index_col='Date',
    parse_dates=True,
    infer_datetime_format=True
)


# Verify the data type transformation using the info function
df_mercado_trends.info()

# Review the first and last five rows of the DataFrame
review_dfmt = pd.concat([df_mercado_trends.head(), df_mercado_trends.tail()])
review_dfmt
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving google_hourly_search_trends.csv to google_hourly_search_trends (2).csv
<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
```

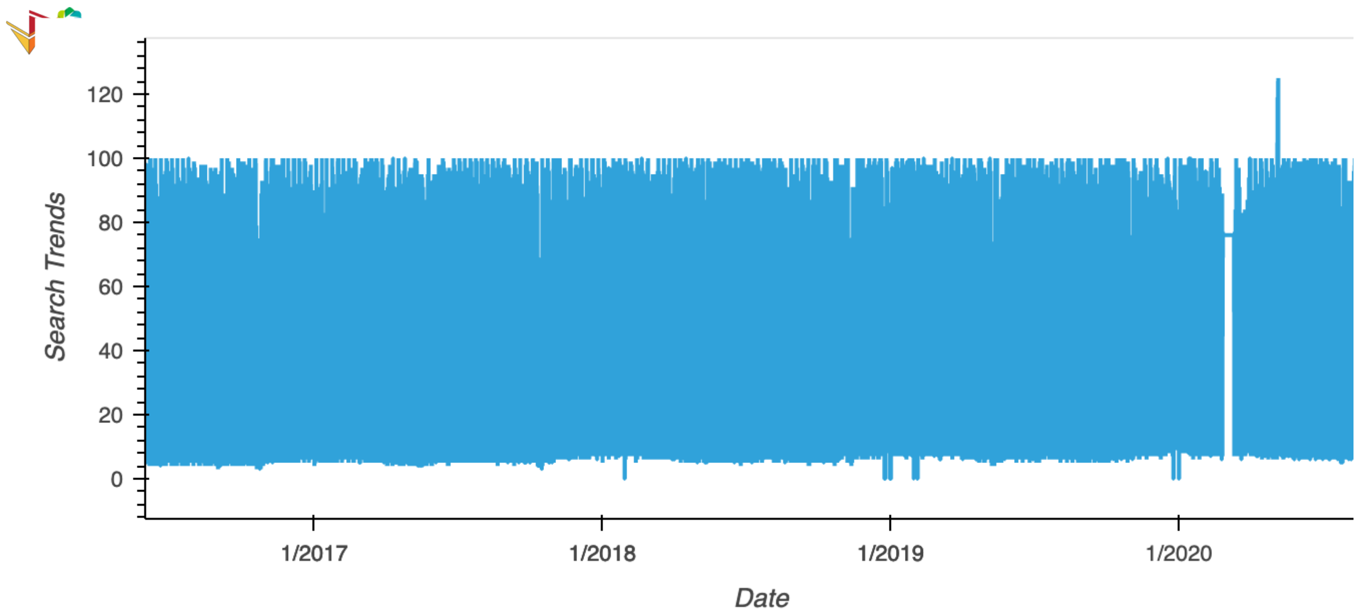
	Search Trends
Date	
2016-06-01 00:00:00	97
2016-06-01 01:00:00	92
2016-06-01 02:00:00	76
2016-06-01 03:00:00	60
2016-06-01 04:00:00	38
2020-09-07 20:00:00	71
2020-09-07 21:00:00	83
2020-09-07 22:00:00	96
2020-09-07 23:00:00	97
2020-09-08 00:00:00	96

```
# 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')
df_mercado_trends.hvplot()
```




```
# 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':'2020-05-31']

# 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['Search Trends'].sum()

# View the traffic_may_2020 value
traffic_may_2020
```

```
38181
```

```
# Calcluate the monhtly median search traffic across all months
mm_st= df_mercado_trends['Search Trends'].groupby(by=[df_mercado_trends.index.month]).
mm= mm_st*24*30
mm
```

```
Date
1     38160.0
2     38160.0
3     37440.0
4     36720.0
5     36000.0
6     36720.0
7     36720.0
8     36720.0
9     36000.0
10    34560.0
11    37440.0
12    36000.0
Name: Search Trends, dtype: float64
```

```
# Group the DataFrame by index year and then index month, chain the sum and then the m
monthly_traffic = df_mercado_trends["Search Trends"].groupby(by=[df_mercado_trends.inc
# View the median_monthly_traffic value
monthly_traffic
```

```
Date  Date
2016  6       33196
      7       33898
      8       34459
      9       32376
      10      32334
```

```
        11      33793
        12      33789
  2017  1       32984
        2       31901
        3       35363
        4       32522
        5       33216
        6       34211
        7       34988
        8       36113
        9       33693
        10      32842
        11      35144
        12      35420
  2018  1       37347
        2       33748
        3       36051
        4       35283
        5       35309
        6       34115
        7       35927
        8       37012
        9       34037
        10      35879
        11      34686
        12      35245
  2019  1       38505
        2       34129
        3       37331
        4       35505
        5       34983
        6       36120
        7       37089
        8       37540
        9       35201
        10      37212
        11      36280
        12      37825
  2020  1       39177
        2       30838
        3       24805
        4       35229
        5       38181
        6       35758
        7       35620
        8       33530
        9        8126
  Name: Search Trends, dtype: int64
```

```
# Compare the seach traffic for the month of May 2020 to the overall monthly median va
comparative = mm/traffic_may_2020
comparative
```

```
  Date
```

```
1     0.999450
2     0.999450
3     0.980592
4     0.961735
5     0.942877
6     0.961735
7     0.961735
8     0.961735
9     0.942877
10    0.905162
11    0.980592
12    0.942877
Name: Search Trends, dtype: float64
```

Answer the following question:

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

Answer: # Tricky question. Searches peaked to all time highes in May, 5, 2020 at 23:00 with 125 views per hour but it did not seem to have a big impact on the overall monthly average.

Step 2: Mine the Search Traffic Data for Seasonality

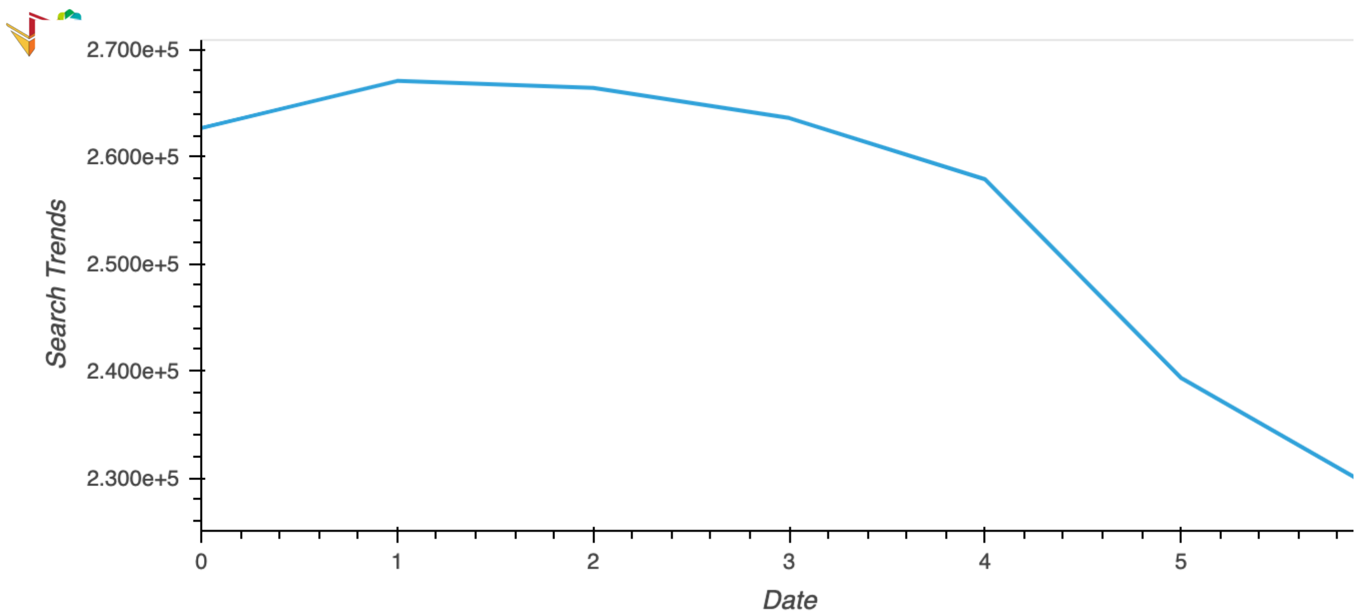
Marketing realizes that they can use the hourly search data, too. If they can track and predict interest in the company and its platform for any time of day, they can focus their marketing efforts around the times that have the most traffic. This will get a greater return on investment (ROI) from their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


# Then Groupby that, choosing an aggregation function
dow_grouped = df_mercado_trends.groupby(by=dayofweek).sum()
dow_grouped.hvplot()
```



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 hourly trends across days of week in a heatmap
df_mercado_trends.hvplot.heatmap(x='index.hour', y='index.dayofweek', C='Search Trends
```

Answer the following question:

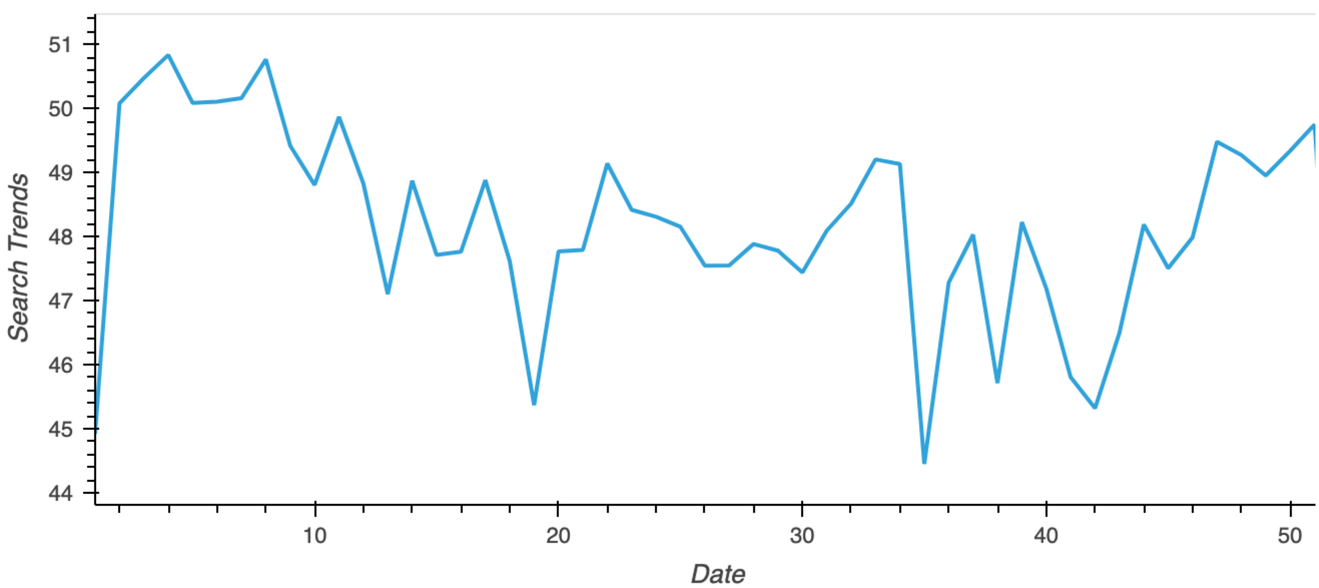
Question: Does any day-of-week effect that you observe concentrate in just a few hours of that day?

Answer: # Yes, there is a concentration between 21:00 and 03:00.

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
df_mercado_trends.groupby(df_mercado_trends.index.weekofyear).mean().hvplot()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: FutureWarnin
"""
```



Answer the following question:

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

Answer: # Yes it increases but not as much as during the period between week one and nine.

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 DataFra
# Set the "date" column as the Datetime Index.
from google.colab import files
```

```
uploaded = files.upload()

df_mercado_stock = pd.read_csv(
    "mercado_stock_price.csv",
    index_col='date',
    parse_dates=True,
    infer_datetime_format=True
)

mixed_df = pd.concat([df_mercado_stock, df_mercado_trends])
mixed_df
```

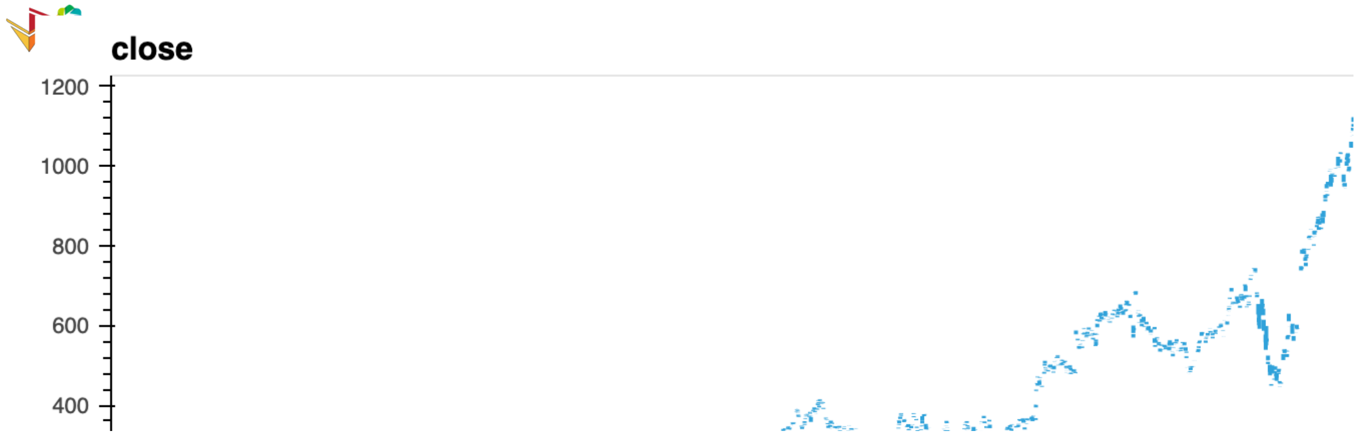
Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving mercado_stock_price.csv to mercado_stock_price (9).csv
```

	close	Search Trends
2015-01-02 09:00:00	127.67	NaN
2015-01-02 10:00:00	125.44	NaN
2015-01-02 11:00:00	125.57	NaN
2015-01-02 12:00:00	125.40	NaN
2015-01-02 13:00:00	125.17	NaN
...	...	...
2020-09-07 20:00:00	NaN	71.0
2020-09-07 21:00:00	NaN	83.0
2020-09-07 22:00:00	NaN	96.0
2020-09-07 23:00:00	NaN	97.0
2020-09-08 00:00:00	NaN	96.0

86001 rows × 2 columns

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

# Use hvPlot to visualize the closing price of the df_mercado_stock DataFrame
cp_df = mixed_df["close"].hvplot()
cp_df
```




```
# 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 colur
mercado_stock_trends_df = pd.concat([df_mercado_stock, df_mercado_trends], axis=1).drc
mercado_stock_trends_df
```

	close	Search Trends
2016-06-01 09:00:00	135.160	6.0
2016-06-01 10:00:00	136.630	12.0
2016-06-01 11:00:00	136.560	22.0
2016-06-01 12:00:00	136.420	33.0
2016-06-01 13:00:00	136.100	40.0
...	...	...
2020-07-31 11:00:00	1105.780	20.0
2020-07-31 12:00:00	1087.925	32.0
2020-07-31 13:00:00	1095.800	41.0
2020-07-31 14:00:00	1110.650	47.0
2020-07-31 15:00:00	1122.510	53.0

7067 rows × 2 columns

```
# View the first and last five rows of the DataFrame
rev = pd.concat([mercado_stock_trends_df.head(), mercado_stock_trends_df.tail()])
rev
```

	close	Search Trends
2016-06-01 09:00:00	135.160	6.0
2016-06-01 10:00:00	136.630	12.0
2016-06-01 11:00:00	136.560	22.0
2016-06-01 12:00:00	136.420	33.0
2016-06-01 13:00:00	136.100	40.0
2020-07-31 11:00:00	1105.780	20.0

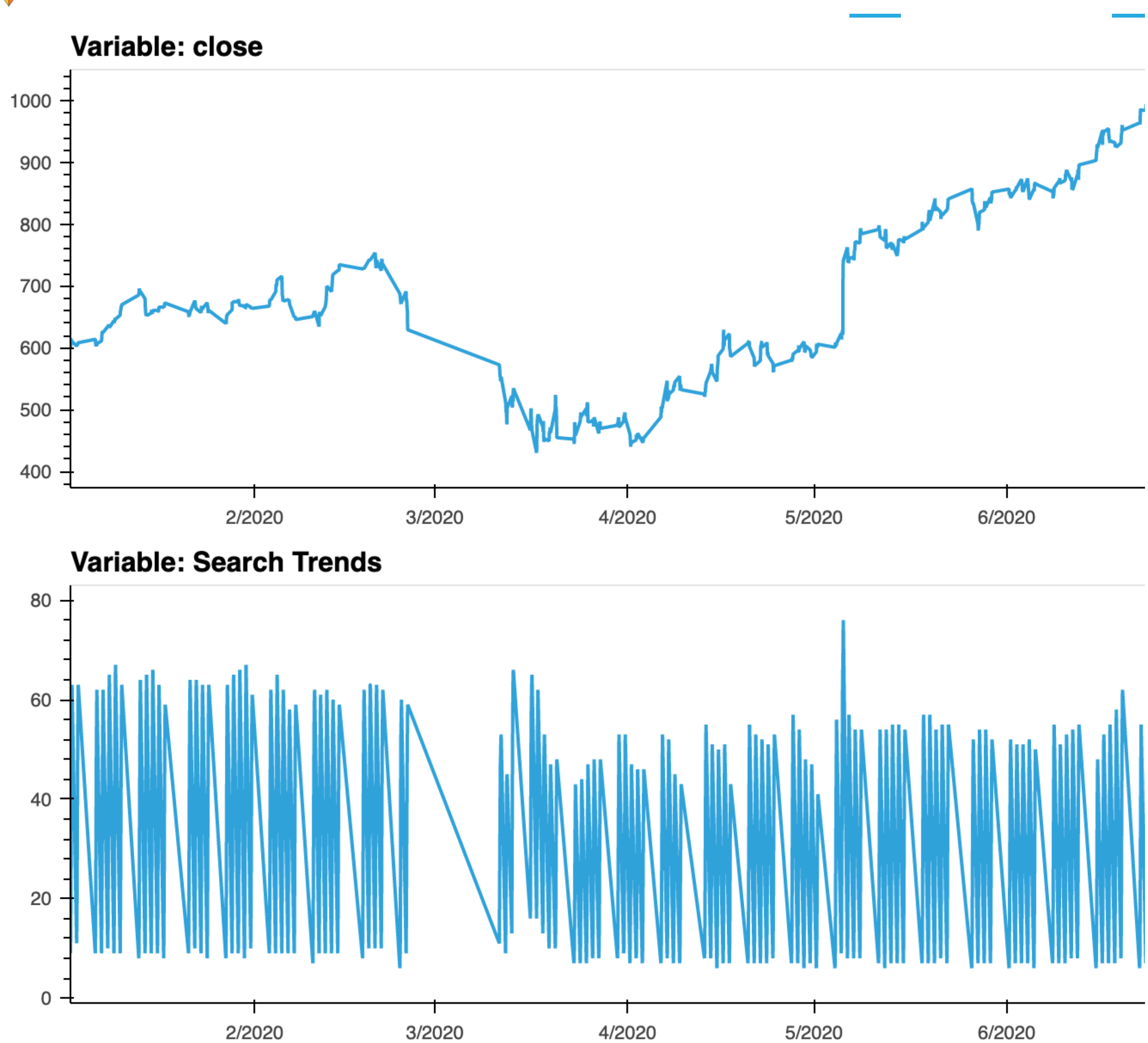
Step 2: Market events emerged during the year of 2020 that many companies found difficult. But, after the initial shock to global financial markets, new customers and revenue increased for e-commerce platforms. Slice the data to just the first half of 2020 (`2020-01` to `2020-06` in the DataFrame), and 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 2(

combi = mercado_stock_trends_df.loc['2020-01-01':'2020-06-30']
# View the first and last five rows of first_half_2020 DataFrame
combi
```

	close	Search Trends
2020-01-02 09:00:00	601.085	9.0
2020-01-02 10:00:00	601.290	14.0
2020-01-02 11:00:00	615.410	25.0
2020-01-02 12:00:00	611.400	37.0
2020-01-02 13:00:00	611.830	50.0
...	...	...
2020-06-30 11:00:00	976.170	17.0
2020-06-30 12:00:00	977.500	27.0
2020-06-30 13:00:00	973.230	37.0
2020-06-30 14:00:00	976.500	45.0
2020-06-30 15:00:00	984.930	51.0

807 rows × 2 columns

```
# 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)`
combi.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: # Yes, a deop in 3/2020 is visible in both charts, as well as an increase around 5/20

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 Tr
# This column should shift the Search Trends information by one hour
mercado_stock_trends_df['Lagged Volume'] = mercado_stock_trends_df['Search Trends'].sh
mercado_stock_trends_df
```

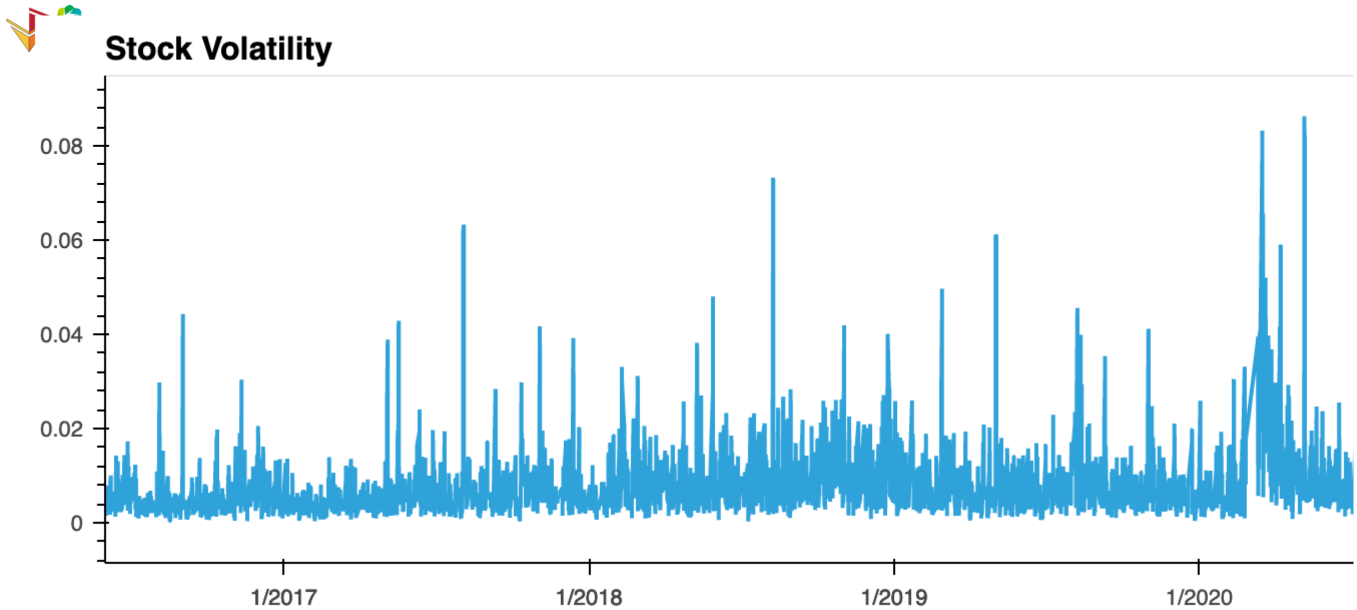
	close	Search Trends	Lagged Volume
2016-06-01 09:00:00	135.160	6.0	NaN
2016-06-01 10:00:00	136.630	12.0	6.0
2016-06-01 11:00:00	136.560	22.0	12.0
2016-06-01 12:00:00	136.420	33.0	22.0
2016-06-01 13:00:00	136.100	40.0	33.0
...	...	...	...
2020-07-31 11:00:00	1105.780	20.0	11.0
2020-07-31 12:00:00	1087.925	32.0	20.0
2020-07-31 13:00:00	1095.800	41.0	32.0
2020-07-31 14:00:00	1110.650	47.0	41.0
2020-07-31 15:00:00	1122.510	53.0	47.0

7067 rows × 3 columns

```
# 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 retur
mercado_stock_trends_df['Stock Volatility'] = mercado_stock_trends_df['close'].pct_cha
```

```
# 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 Ret
# This column should calculate hourly return percentage of the closing price
mercado_stock_trends_df['Hourly Stock Return'] = mercado_stock_trends_df['close'].pct_
```

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

	close	Search Trends	Lagged Volume	Stock Volatility	Hourly Stock Return
2016-06-01 09:00:00	135.160	6.0	NaN	NaN	NaN
2016-06-01					

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?

12:00:00

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

	Stock Volatility	Lagged Volume	Hourly Stock Return
Stock Volatility	1.000000	-0.148938	0.061424
Lagged Volume	-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 seems to be no correlation

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 nc
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
```

	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
...	...	...
37101	2020-09-07 20:00:00	71
37102	2020-09-07 21:00:00	83
37103	2020-09-07 22:00:00	96
37104	2020-09-07 23:00:00	97
37105	2020-09-08 00:00:00	96

37106 rows × 2 columns

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

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

```
<fbprophet.forecaster.Prophet at 0x7f8796ca4b50>
```

```
# 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=

# 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

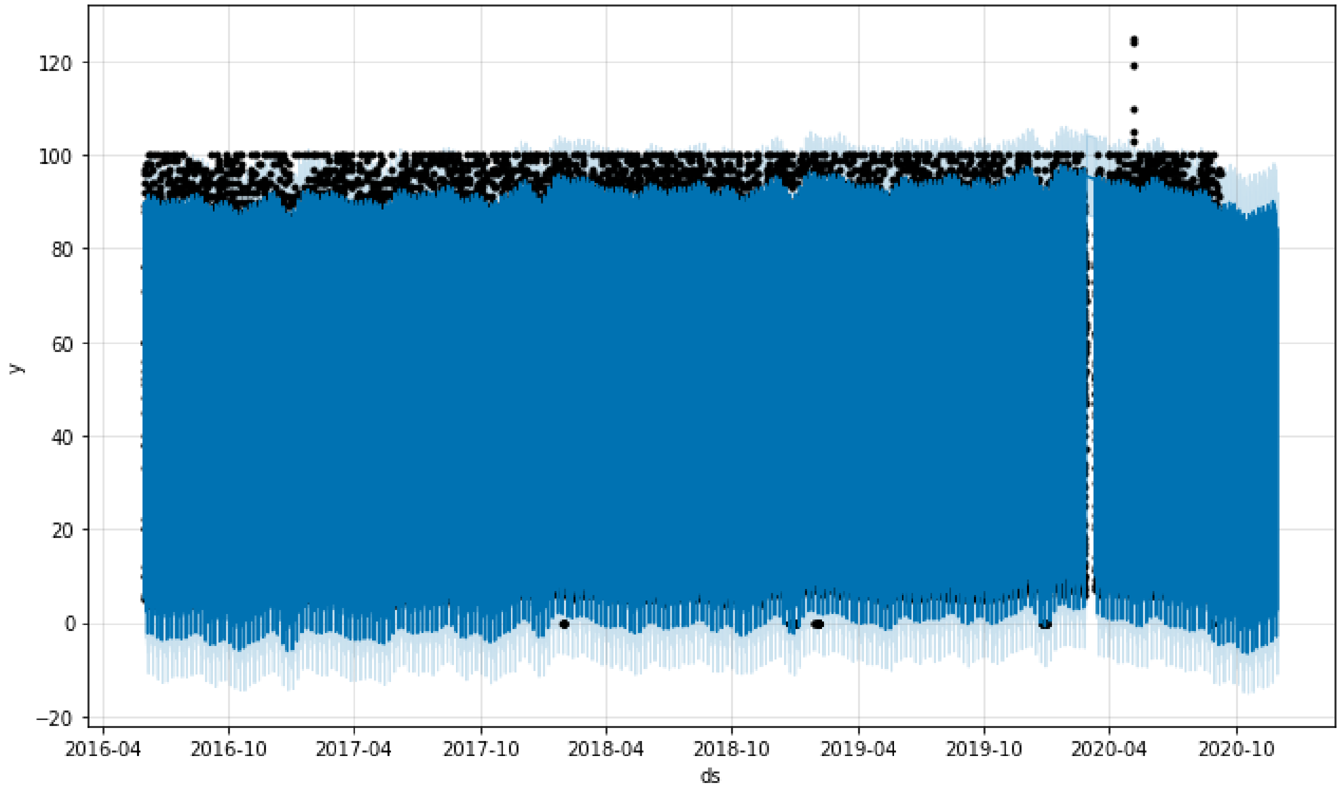
```
# Make the predictions for the trend data using the future_mercado_trends DataFrame
forecast_mercado_trends = model_mercado_trends.predict(future_mercado_trends)
```

```
# Display the first five rows of the forecast_mercado_trends DataFrame
forecast_mercado_trends
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additi
0	2016-06-01 00:00:00	44.437254	81.591648	97.727624	44.437254	44.437254	
1	2016-06-01 01:00:00	44.438181	78.109386	94.223090	44.438181	44.438181	
2	2016-06-01 02:00:00	44.439108	67.420714	84.131880	44.439108	44.439108	
3	2016-06-01 03:00:00	44.440034	51.430331	68.792793	44.440034	44.440034	
4	2016-06-01 04:00:00	44.440961	35.167797	51.812724	44.440961	44.440961	

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: #Needs work. A Decline can be seen mid 2020 and moving to the future.

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')
forecast_mercado_trends
```

ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_ter
2016-06-01 00:00:00	44.437254	81.591648	97.727624	44.437254	44.437254	45.1987
2016-06-01 01:00:00	44.438181	78.109386	94.223090	44.438181	44.438181	41.6445
2016-06-01 02:00:00	44.439108	67.420714	84.131880	44.439108	44.439108	31.3210
2016-06-01 03:00:00	44.440034	51.430331	68.792793	44.440034	44.440034	16.0537
2016-06-01 04:00:00	44.440961	35.167797	51.812724	44.440961	44.440961	-1.0610
...	...	...	...	...	...	
2020-11-30 04:00:00	45.120891	31.388366	48.619417	44.187880	46.061138	-5.3900
2020-11-30 05:00:00	45.120148	15.710089	32.664637	44.185920	46.060989	-20.8604
2020-11-30 06:00:00	45.119405	3.928660	20.289104	44.183960	46.060841	-32.8253
2020-11-30 07:00:00	45.118662	-2.768420	13.057217	44.182001	46.060692	-40.0967
2020-11-30	45.117920	-5.654220	11.621939	44.180097	46.060544	-42.2909

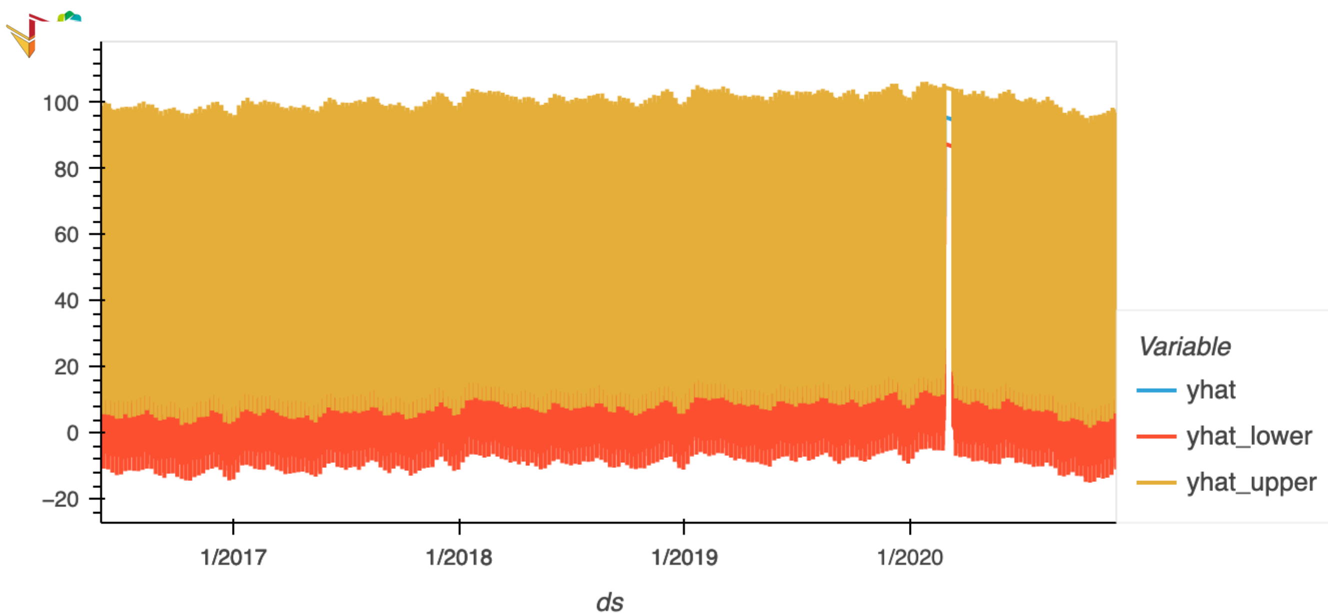
```
# View the only the yhat,yhat_lower and yhat_upper columns from the DataFrame
forecast_mercado_trends[['yhat', 'yhat_lower', 'yhat_upper']].head()
```

yhat yhat_lower yhat_upper

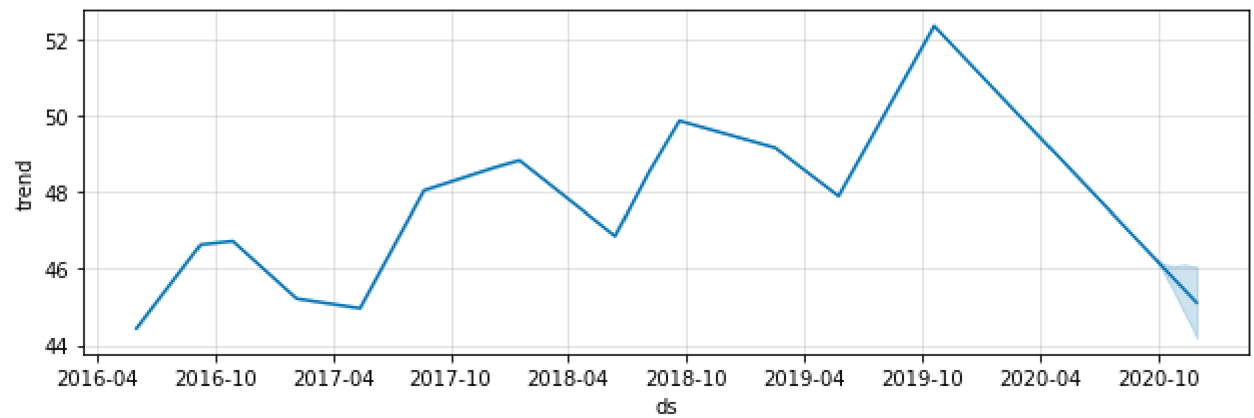
ds

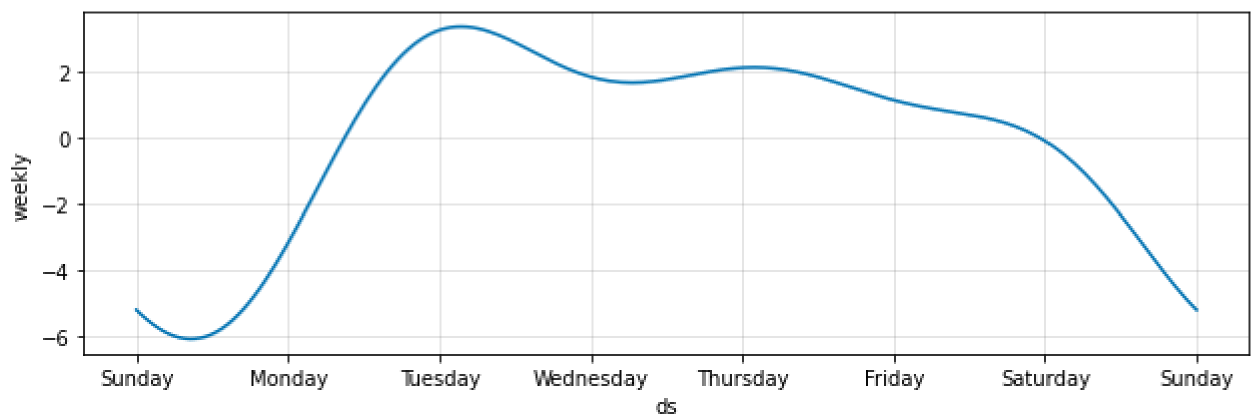
Solutions Note: `yhat` represents the most likely (average) forecast, whereas `yhat_lower` and `yhat_upper` represents the worst and best case prediction (based on what are known as 95% confidence intervals).

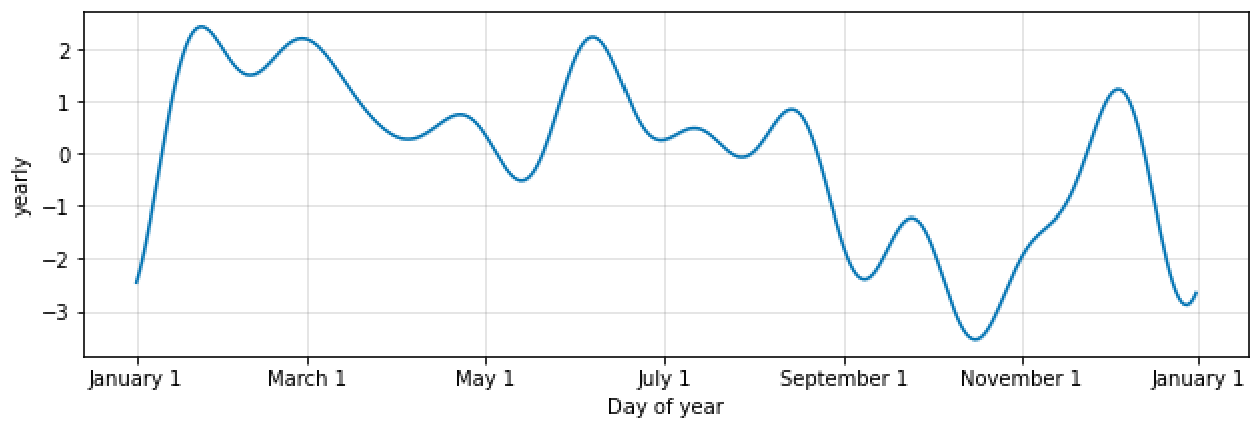
```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# 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']].hvplot()
```

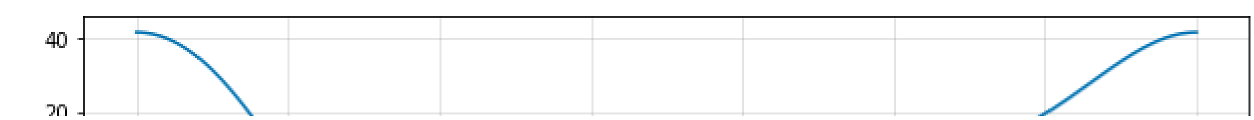



```
# Reset the index in the forecast_mercado_trends DataFrame
forecast_mercado_trends = forecast_mercado_trends.reset_index()

# Use the plot_components function to visualize the forecast results
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: Midnight

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

Answer: # Tuesday

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

Answer: # Late October

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.

[ ] ↳ 6 cells hidden

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?)

[ ] ↳ 3 cells hidden

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.

[ ] ↳ 4 cells hidden

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