# Install and import the required libraries and dependencies

In [7]:
```
# Install the required libraries
!pip install prophet
!pip install hvplot
!pip install holoviews
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels
/public/simple/
Requirement already satisfied: prophet in /usr/local/lib/python3.9/dist-packages
(1.1.2)
Requirement already satisfied: matplotlib>=2.0.0 in /usr/local/lib/python3.9/dist-p
ackages (from prophet) (3.7.1)
Requirement already satisfied: LunarCalendar>=0.0.9 in /usr/local/lib/python3.9/dis
t-packages (from prophet) (0.0.9)
Requirement already satisfied: cmdstanpy>=1.0.4 in /usr/local/lib/python3.9/dist-pa
ckages (from prophet) (1.1.0)
Requirement already satisfied: holidays>=0.14.2 in /usr/local/lib/python3.9/dist-pa
ckages (from prophet) (0.22)
Requirement already satisfied: convertdate>=2.1.2 in /usr/local/lib/python3.9/dist-
packages (from prophet) (2.4.0)
Requirement already satisfied: tqdm>=4.36.1 in /usr/local/lib/python3.9/dist-packag
es (from prophet) (4.65.0)
Requirement already satisfied: numpy>=1.15.4 in /usr/local/lib/python3.9/dist-packa
ges (from prophet) (1.22.4)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.9/d
ist-packages (from prophet) (2.8.2)
Requirement already satisfied: pandas>=1.0.4 in /usr/local/lib/python3.9/dist-packa
ges (from prophet) (1.5.3)
Requirement already satisfied: pymeeus<=1,>=0.3.13 in /usr/local/lib/python3.9/dist
-packages (from convertdate>=2.1.2->prophet) (0.5.12)
Requirement already satisfied: hijri-converter in /usr/local/lib/python3.9/dist-pac
kages (from holidays>=0.14.2->prophet) (2.2.4)
Requirement already satisfied: korean-lunar-calendar in /usr/local/lib/python3.9/di
st-packages (from holidays>=0.14.2->prophet) (0.3.1)
Requirement already satisfied: ephem>=3.7.5.3 in /usr/local/lib/python3.9/dist-pack
ages (from LunarCalendar>=0.0.9->prophet) (4.1.4)
Requirement already satisfied: pytz in /usr/local/lib/python3.9/dist-packages (from
LunarCalendar>=0.0.9->prophet) (2022.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.9/dist-pa
ckages (from matplotlib>=2.0.0->prophet) (1.0.7)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.9/dist-p
ackages (from matplotlib>=2.0.0->prophet) (4.39.3)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-pac
kages (from matplotlib>=2.0.0->prophet) (23.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.9/dist-packag
es (from matplotlib>=2.0.0->prophet) (0.11.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-packa
ges (from matplotlib>=2.0.0->prophet) (8.4.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.9/dist-p
ackages (from matplotlib>=2.0.0->prophet) (1.4.4)
Requirement already satisfied: importlib-resources>=3.2.0 in /usr/local/lib/python
3.9/dist-packages (from matplotlib>=2.0.0->prophet) (5.12.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.9/dist-pa
ckages (from matplotlib>=2.0.0->prophet) (3.0.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages
(from python-dateutil>=2.8.0->prophet) (1.16.0)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.9/dist-package
s (from importlib-resources>=3.2.0->matplotlib>=2.0.0->prophet) (3.15.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels
/public/simple/
Collecting hvplot
  Downloading hvplot-0.8.3-py2.py3-none-any.whl (3.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.2/3.2 MB 30.1 MB/s eta 0:00:00
Requirement already satisfied: bokeh>=1.0.0 in /usr/local/lib/python3.9/dist-packag
es (from hvplot) (2.4.3)
```

```
Requirement already satisfied: packaging in /usr/local/lib/python3.9/dist-packages
(from hvplot) (23.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.9/dist-packages (fr
om hvplot) (1.5.3)
Requirement already satisfied: panel>=0.11.0 in /usr/local/lib/python3.9/dist-packa
ges (from hvplot) (0.14.4)
Requirement already satisfied: param>=1.9.0 in /usr/local/lib/python3.9/dist-packag
es (from hvplot) (1.13.0)
Requirement already satisfied: colorcet>=2 in /usr/local/lib/python3.9/dist-package
s (from hvplot) (3.0.1)
Requirement already satisfied: holoviews>=1.11.0 in /usr/local/lib/python3.9/dist-p
ackages (from hvplot) (1.15.4)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.9/dist-package
s (from hvplot) (1.22.4)
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.9/dist-packag
es (from bokeh>=1.0.0->hvplot) (6.0)
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.9/dist-packa
ges (from bokeh>=1.0.0->hvplot) (8.4.0)
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.9/dist-package
s (from bokeh>=1.0.0->hvplot) (3.1.2)
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.9/dist-packag
es (from bokeh>=1.0.0->hvplot) (6.2)
Requirement already satisfied: typing-extensions>=3.10.0 in /usr/local/lib/python3.
9/dist-packages (from bokeh>=1.0.0->hvplot) (4.5.0)
Requirement already satisfied: pyct>=0.4.4 in /usr/local/lib/python3.9/dist-package
s (from colorcet>=2->hvplot) (0.5.0)
Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.9/dist-
packages (from holoviews>=1.11.0->hvplot) (2.2.1)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packag
es (from pandas->hvplot) (2022.7.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.9/d
ist-packages (from pandas->hvplot) (2.8.2)
Requirement already satisfied: setuptools>=42 in /usr/local/lib/python3.9/dist-pack
ages (from panel>=0.11.0->hvplot) (67.6.1)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-packages
(from panel>=0.11.0->hvplot) (2.27.1)
Requirement already satisfied: markdown in /usr/local/lib/python3.9/dist-packages
(from panel>=0.11.0->hvplot) (3.4.3)
Requirement already satisfied: bleach in /usr/local/lib/python3.9/dist-packages (fr
om panel>=0.11.0->hvplot) (6.0.0)
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.9/dist-packag
es (from panel>=0.11.0->hvplot) (4.65.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-pac
kages (from Jinja2>=2.9->bokeh>=1.0.0->hvplot) (2.1.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages
(from python-dateutil>=2.8.1->pandas->hvplot) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.9/dist-packag
es (from bleach->panel>=0.11.0->hvplot) (0.5.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.9/
dist-packages (from markdown->panel>=0.11.0->hvplot) (6.4.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-
packages (from requests->panel>=0.11.0->hvplot) (2022.12.7)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packag
es (from requests->panel>=0.11.0->hvplot) (3.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.
9/dist-packages (from requests->panel>=0.11.0->hvplot) (2.0.12)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/di
st-packages (from requests->panel>=0.11.0->hvplot) (1.26.15)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.9/dist-packages
```

```
(from importlib-metadata>=4.4->markdown->panel>=0.11.0->hvplot) (3.15.0)
Installing collected packages: hvplot
Successfully installed hvplot-0.8.3
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels
/public/simple/
Requirement already satisfied: holoviews in /usr/local/lib/python3.9/dist-packages
(1.15.4)
Requirement already satisfied: colorcet in /usr/local/lib/python3.9/dist-packages
(from holoviews) (3.0.1)
Requirement already satisfied: pyviz-comms>=0.7.4 in /usr/local/lib/python3.9/dist-
packages (from holoviews) (2.2.1)
Requirement already satisfied: param<2.0,>=1.9.3 in /usr/local/lib/python3.9/dist-p
ackages (from holoviews) (1.13.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.9/dist-packages
(from holoviews) (23.1)
Requirement already satisfied: numpy>=1.0 in /usr/local/lib/python3.9/dist-packages
(from holoviews) (1.22.4)
Requirement already satisfied: pandas>=0.20.0 in /usr/local/lib/python3.9/dist-pack
ages (from holoviews) (1.5.3)
Requirement already satisfied: panel>=0.13.1 in /usr/local/lib/python3.9/dist-packa
ges (from holoviews) (0.14.4)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packag
es (from pandas>=0.20.0->holoviews) (2022.7.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.9/d
ist-packages (from pandas>=0.20.0->holoviews) (2.8.2)
Requirement already satisfied: pyct>=0.4.4 in /usr/local/lib/python3.9/dist-package
s (from panel>=0.13.1->holoviews) (0.5.0)
Requirement already satisfied: tqdm>=4.48.0 in /usr/local/lib/python3.9/dist-packag
es (from panel>=0.13.1->holoviews) (4.65.0)
Requirement already satisfied: setuptools>=42 in /usr/local/lib/python3.9/dist-pack
ages (from panel>=0.13.1->holoviews) (67.6.1)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.9/dist-p
ackages (from panel>=0.13.1->holoviews) (4.5.0)
Requirement already satisfied: bleach in /usr/local/lib/python3.9/dist-packages (fr
om panel>=0.13.1->holoviews) (6.0.0)
Requirement already satisfied: markdown in /usr/local/lib/python3.9/dist-packages
(from panel>=0.13.1->holoviews) (3.4.3)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-packages
(from panel>=0.13.1->holoviews) (2.27.1)
Requirement already satisfied: bokeh<2.5.0,>=2.4.0 in /usr/local/lib/python3.9/dist
-packages (from panel>=0.13.1->holoviews) (2.4.3)
Requirement already satisfied: Jinja2>=2.9 in /usr/local/lib/python3.9/dist-package
s (from bokeh<2.5.0,>=2.4.0->panel>=0.13.1->holoviews) (3.1.2)
Requirement already satisfied: tornado>=5.1 in /usr/local/lib/python3.9/dist-packag
es (from bokeh<2.5.0,>=2.4.0->panel>=0.13.1->holoviews) (6.2)
Requirement already satisfied: pillow>=7.1.0 in /usr/local/lib/python3.9/dist-packa
ges (from bokeh<2.5.0,>=2.4.0->panel>=0.13.1->holoviews) (8.4.0)
Requirement already satisfied: PyYAML>=3.10 in /usr/local/lib/python3.9/dist-packag
es (from bokeh<2.5.0,>=2.4.0->panel>=0.13.1->holoviews) (6.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages
(from python-dateutil>=2.8.1->pandas>=0.20.0->holoviews) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.9/dist-packag
es (from bleach->panel>=0.13.1->holoviews) (0.5.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.9/
dist-packages (from markdown->panel>=0.13.1->holoviews) (6.4.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-
packages (from requests->panel>=0.13.1->holoviews) (2022.12.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.
9/dist-packages (from requests->panel>=0.13.1->holoviews) (2.0.12)
```

```
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/di
st-packages (from requests->panel>=0.13.1->holoviews) (1.26.15)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packag
es (from requests->panel>=0.13.1->holoviews) (3.4)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.9/dist-packages
(from importlib-metadata>=4.4->markdown->panel>=0.13.1->holoviews) (3.15.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-pac
kages (from Jinja2>=2.9->bokeh<2.5.0,>=2.4.0->panel>=0.13.1->holoviews) (2.1.2)
```

In [21]:
```
# Import the required libraries and dependencies
import pandas as pd
import numpy as np
import holoviews as hv
from prophet import Prophet
import hvplot.pandas
import datetime as dt
%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?

In [9]:
```
# Upload the "google_hourly_search_trends.csv" file into Colab, then store in a Pan

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

Browse... No files selected. 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.csv
```

In [10]:
```
# Set the "Date" column as the Datetime Index.
df_mercado_trends = pd.read_csv(
    "google_hourly_search_trends.csv",
    index_col="Date",
    infer_datetime_format=True,
    parse_dates=True
    )

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

| Date | Search Trends |
|---|---|
| **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 |

| Date | Search Trends |
|---|---|
| **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 |

In [11]:
```
# 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
```

In [12]:
```
# 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()
```

Out[12]:

### 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?

In [13]:
```
# Calculate the sum of the total search traffic for May 2020
traffic_may_2020 = df_may_2020.median()

# View the traffic_may_2020 value
traffic_may_2020
```

Out[13]:
```
Search Trends    54.0
dtype: float64
```

In [29]:
```
# Calcluate the monhtly median search traffic across all months
# Group the DataFrame by index year and then index month, chain the sum and then th
median_monthly_traffic = df_mercado_trends.median()

# View the median_monthly_traffic value
median_monthly_traffic
```

Out[29]:
```
Search Trends    51.0
dtype: float64
```

Answer the following question:

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

**Answer:** Yes, The Google search traffic increased 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)?

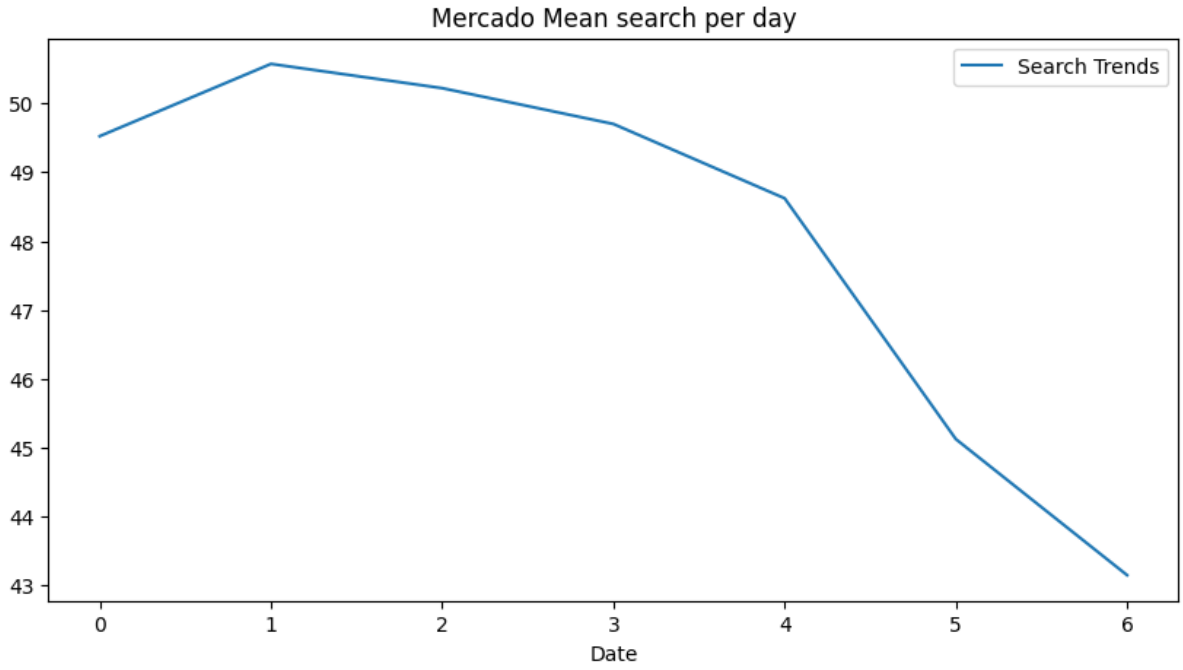
## Step 1: Group the hourly search data to plot the average traffic by the day of the week (for example, Monday vs. Friday).

In [23]:
```
# 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
df_mercado_trends.groupby(by=[df_mercado_trends.index.dayofweek]).mean().plot(
    title="Mercado Mean search per day",
    figsize=[10, 5]
)
```

Out[23]:

```
<Axes: title={'center': 'Mercado Mean search per day'}, xlabel='Date'>
```




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?

In [22]:
```
# 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 h
df_mercado_trends.hvplot.heatmap(
    x="index.hour",
    y="index.dayofweek",
    C="Search Trends",
    cmap="reds"
).aggregate(function=np.mean)
```

Out[22]:

Answer the following question:

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

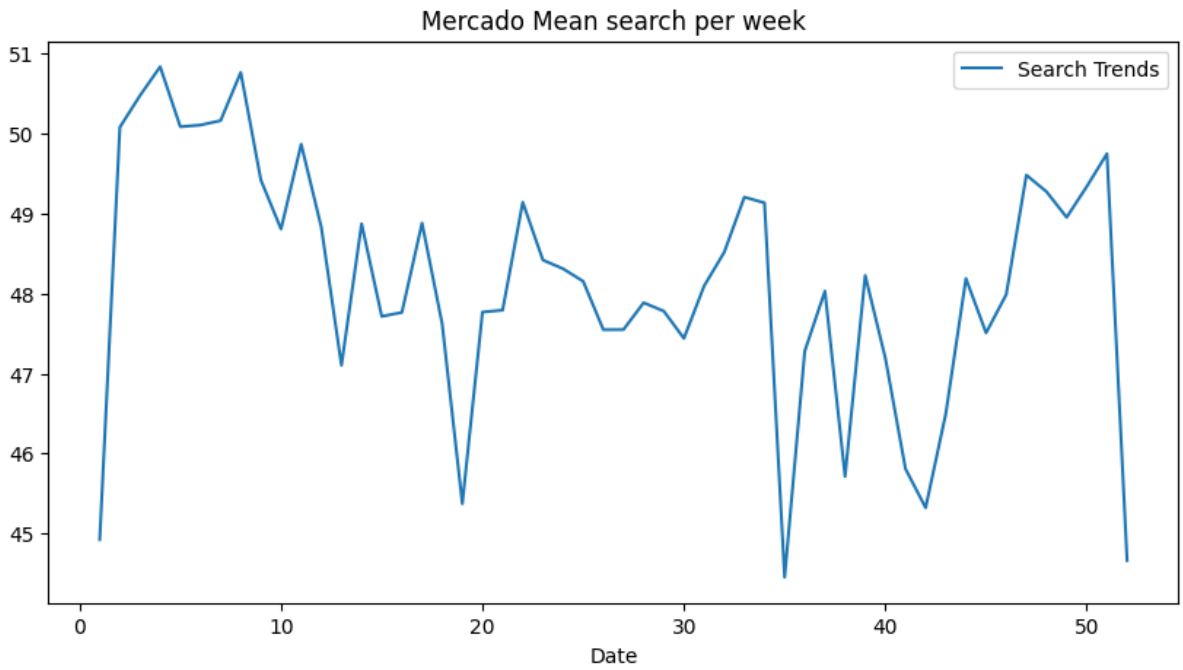
**Answer:** # Day 1 of the week followwed by 2,3 and 4 effect that concentrate in just few hours of the day that is around 22nd and 23rd hour of the day.

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)?

In [24]:
```
# 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
df_mercado_trends.groupby(by=[df_mercado_trends.index.weekofyear]).mean().plot(
    title="Mercado Mean search per week",
    figsize=[10, 5]
)
```

Out[24]:
```
<ipython-input-24-55195e148678>:5: FutureWarning: weekofyear and week have been dep
recated, please use DatetimeIndex.isocalendar().week instead, which returns a Serie
s. To exactly reproduce the behavior of week and weekofyear and return an Index, yo
u may call pd.Int64Index(idx.isocalendar().week)
  df_mercado_trends.groupby(by=[df_mercado_trends.index.weekofyear]).mean().plot(
<Axes: title={'center': 'Mercado Mean search per week'}, xlabel='Date'>
```




Answer the following question:

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

**Answer:** Plot shows that 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 1: Read in and plot the stock price data. Concatenate the stock price data to the search data in a single DataFrame.

In [25]:
```
# Upload the "mercado_stock_price.csv" file into Colab, then store in a Pandas Data
# Set the "date" column as the Datetime Index.
from google.colab import files
uploaded = files.upload()
```

Browse... No files selected. 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.csv
```

In [41]:
```
df_mercado_stock = pd.read_csv(
    "mercado_stock_price.csv",
    index_col="date",
    infer_datetime_format=True,
    parse_dates=True
    )

# Review the first and last five rows of the DataFrame
display(df_mercado_stock.head())
display(df_mercado_stock.tail())
```

| date | close |
|---|---|
| **2015-01-02 09:00:00** | 127.67 |
| **2015-01-02 10:00:00** | 125.44 |
| **2015-01-02 11:00:00** | 125.57 |
| **2015-01-02 12:00:00** | 125.40 |
| **2015-01-02 13:00:00** | 125.17 |

| date | close |
|---|---|
| **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 |

In [42]:
```
# 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.hvplot()
```

Out[42]:

In [49]:
```
# 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 co
mercado_stock_trends_df = pd.concat([df_mercado_trends, df_mercado_stock], axis=1).

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

| | 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 |

| | Search Trends | close |
|---|---|---|
| **2020-07-31 11:00:00** | 20.0 | 1105.780 |
| **2020-07-31 12:00:00** | 32.0 | 1087.925 |
| **2020-07-31 13:00:00** | 41.0 | 1095.800 |
| **2020-07-31 14:00:00** | 47.0 | 1110.650 |
| **2020-07-31 15:00:00** | 53.0 | 1122.510 |

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?

In [50]:
```
# For the combined dataframe, slice to just the first half of 2020 (2020-01 through
first_half_2020 = mercado_stock_trends_df.loc["2020-01":"2020-06"]

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

| | 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 |

| | Search Trends | close |
|---|---|---|
| **2020-06-30 11:00:00** | 17.0 | 976.17 |
| **2020-06-30 12:00:00** | 27.0 | 977.50 |
| **2020-06-30 13:00:00** | 37.0 | 973.23 |
| **2020-06-30 14:00:00** | 45.0 | 976.50 |
| **2020-06-30 15:00:00** | 51.0 | 984.93 |

In [48]:
```
# 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(shared_axes=False, subplots=True).cols(1)
```

Out[48]:

Answer the following question:

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

**Answer:** Both the time series does not indicate a common trend that's consistent with this narrative.

## 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

In [55]:
```
# Create a new column in the mercado_stock_trends_df DataFrame called Lagged Search
# This column should shift the Search Trends information by one hour
mercado_stock_trends_df['Lagged Search Trends'] = mercado_stock_trends_df["Search T
mercado_stock_trends_df.head()
```

Out[55]:

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

In [58]:
```
# Create a new column in the mercado_stock_trends_df DataFrame called Stock Volatil
# This column should calculate the standard deviation of the closing stock price re
mercado_stock_trends_df['Stock Volatility'] = mercado_stock_trends_df["close"].roll
mercado_stock_trends_df.head()
```

Out[58]:

| | Search Trends | close | Lagged Search Trends | Stock Volatility |
|---|---|---|---|---|
| **2016-06-01 09:00:00** | 6.0 | 135.16 | NaN | NaN |
| **2016-06-01 10:00:00** | 12.0 | 136.63 | 6.0 | NaN |
| **2016-06-01 11:00:00** | 22.0 | 136.56 | 12.0 | NaN |
| **2016-06-01 12:00:00** | 33.0 | 136.42 | 22.0 | 0.693848 |
| **2016-06-01 13:00:00** | 40.0 | 136.10 | 33.0 | 0.235142 |

In [60]:
```
# Holoviews extension to render hvPlots in Colab
hv.extension('bokeh')

# Use hvPlot to visualize the stock volatility
mercado_stock_trends_df.hvplot()
```

Out[60]:

**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.

In [64]:
```
# Create a new column in the mercado_stock_trends_df DataFrame called Hourly Stock
# This column should calculate hourly return percentage of the closing price
mercado_stock_trends_df["Hourly Stock Return"] = mercado_stock_trends_df["close"].p
```

In [65]:
```
# View the first and last five rows of the mercado_stock_trends_df DataFrame
display(mercado_stock_trends_df.head())
display(mercado_stock_trends_df.tail())
```

| | Search Trends | close | Lagged Search Trends | Stock Volatility | Hourly Stock Return |
|---|---|---|---|---|---|
| **2016-06-01 09:00:00** | 6.0 | 135.16 | NaN | NaN | NaN |
| **2016-06-01 10:00:00** | 12.0 | 136.63 | 6.0 | NaN | 0.010876 |
| **2016-06-01 11:00:00** | 22.0 | 136.56 | 12.0 | NaN | -0.000512 |
| **2016-06-01 12:00:00** | 33.0 | 136.42 | 22.0 | 0.693848 | -0.001025 |
| **2016-06-01 13:00:00** | 40.0 | 136.10 | 33.0 | 0.235142 | -0.002346 |

| | Search Trends | close | Lagged Search Trends | Stock Volatility | Hourly Stock Return |
|---|---|---|---|---|---|
| **2020-07-31 11:00:00** | 20.0 | 1105.780 | 11.0 | 7.495900 | 0.006380 |
| **2020-07-31 12:00:00** | 32.0 | 1087.925 | 20.0 | 12.188462 | -0.016147 |
| **2020-07-31 13:00:00** | 41.0 | 1095.800 | 32.0 | 7.393646 | 0.007239 |
| **2020-07-31 14:00:00** | 47.0 | 1110.650 | 41.0 | 10.169735 | 0.013552 |
| **2020-07-31 15:00:00** | 53.0 | 1122.510 | 47.0 | 15.408790 | 0.010678 |

## 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?

In [66]:
```
# Construct correlation table of Stock Volatility, Lagged Search Trends, and Hourly
mercado_stock_trends_df[["Stock Volatility", "Lagged Search Trends", "Hourly Stock
```

Out[66]:

| | Stock Volatility | Lagged Search Trends | Hourly Stock Return |
|---|---|---|---|
| **Stock Volatility** | 1.000000 | -0.118945 | 0.046723 |
| **Lagged Search Trends** | -0.118945 | 1.000000 | 0.017929 |
| **Hourly Stock Return** | 0.046723 | 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 no 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 1: Set up the Google search data for a Prophet forecasting model.

In [67]:
```
# Using the df_mercado_trends DataFrame, reset the index so the date information is
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
display(mercado_prophet_df.head())
display(mercado_prophet_df.tail())
```

| | 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 |

| | ds | y |
|---|---|---|
| **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 |

In [71]:
```
# Call the Prophet function, store as an object
model_mercado_trends = Prophet()
model_mercado_trends
```

Out[71]:
```
<prophet.forecaster.Prophet at 0x7f5eb59c63d0>
```

In [72]:
```
# Fit the time-series model.
model_mercado_trends.fit(mercado_prophet_df)
```

```
DEBUG:cmdstanpy:input tempfile: /tmp/tmpr_2cmt13/8mqhz39m.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpr_2cmt13/dhg5u6bj.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=30511', 'data', 'file=/tmp/tmpr_2cmt13/8
mqhz39m.json', 'init=/tmp/tmpr_2cmt13/dhg5u6bj.json', 'output', 'file=/tmp/tmpr_2cm
t13/prophet_modelb4fe7or2/prophet_model-20230422131016.csv', 'method=optimize', 'al
gorithm=lbfgs', 'iter=10000']
13:10:16 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
13:10:48 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```

Out[72]:
```
<prophet.forecaster.Prophet at 0x7f5eb59c63d0>
```

In [73]:
```
# 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, fr

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

| | ds |
|---|---|
| **0** | 2016-06-01 00:00:00 |
| **1** | 2016-06-01 01:00:00 |
| **2** | 2016-06-01 02:00:00 |
| **3** | 2016-06-01 03:00:00 |
| **4** | 2016-06-01 04:00:00 |

| | 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 |

In [85]:
```
# 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
display(forecast_mercado_trends.head())
display(forecast_mercado_trends.tail())
```

| | ds | trend | yhat_lower | yhat_upper | trend_lower | trend_upper | additive_terms | additi |
|---|---|---|---|---|---|---|---|---|
| **0** | 2016-06-01 00:00:00 | 44.273500 | 81.351485 | 97.788804 | 44.273500 | 44.273500 | 45.290820 | |
| **1** | 2016-06-01 01:00:00 | 44.274505 | 77.058148 | 94.298979 | 44.274505 | 44.274505 | 41.736647 | |
| **2** | 2016-06-01 02:00:00 | 44.275511 | 67.203060 | 84.755913 | 44.275511 | 44.275511 | 31.413187 | |
| **3** | 2016-06-01 03:00:00 | 44.276516 | 52.148689 | 68.480329 | 44.276516 | 44.276516 | 16.145999 | |
| **4** | 2016-06-01 04:00:00 | 44.277521 | 34.973064 | 51.801485 | 44.277521 | 44.277521 | -0.968848 | |

5 rows × 22 columns

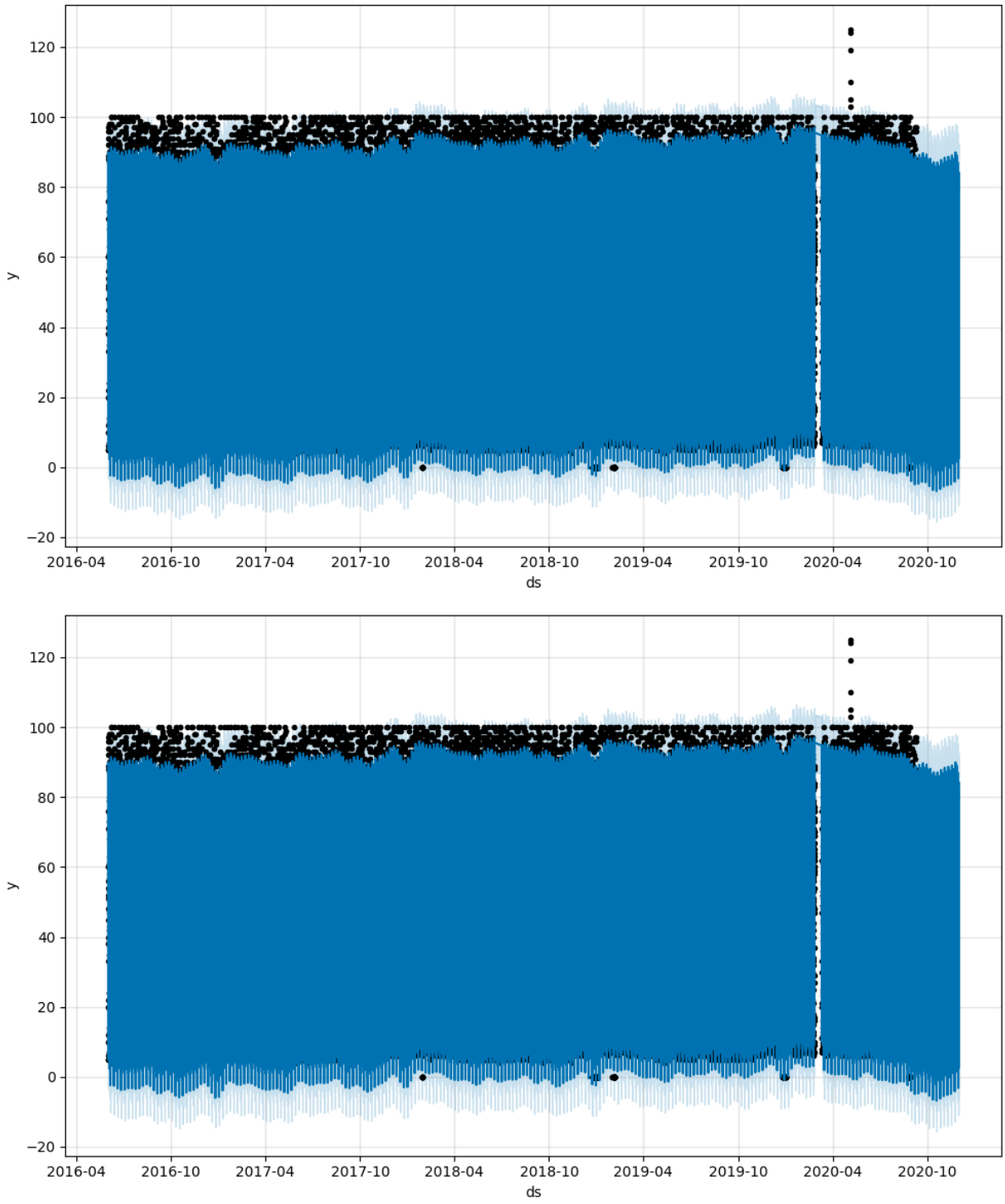
| | ds | trend | yhat_lower | yhat_upper | trend_lower | trend_upper | additive_terms | a |
|---|---|---|---|---|---|---|---|---|
| **39101** | 2020-11-30 04:00:00 | 44.992190 | 30.390874 | 48.135032 | 43.832561 | 45.863174 | -5.446901 | |
| **39102** | 2020-11-30 05:00:00 | 44.991427 | 15.408158 | 32.525066 | 43.830488 | 45.863722 | -20.917300 | |
| **39103** | 2020-11-30 06:00:00 | 44.990663 | 3.234884 | 20.627130 | 43.828337 | 45.864271 | -32.882237 | |
| **39104** | 2020-11-30 07:00:00 | 44.989899 | -4.210793 | 13.054987 | 43.826185 | 45.864819 | -40.153661 | |
| **39105** | 2020-11-30 08:00:00 | 44.989136 | -5.682468 | 11.413799 | 43.824033 | 45.865368 | -42.347823 | |

5 rows × 22 columns

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

In [77]:
```
# Plot the Prophet predictions for the Mercado trends data
model_mercado_trends.plot(forecast_mercado_trends)
```

Out[77]:







Answer the following question:

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

**Answer:** Plot indicates that the near-term forecast for the popularity of MercadoLibre will be stable same as in the past.

## 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?

In [88]:
```
# 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.head()
```

Out[88]:

| ds | trend | yhat_lower | yhat_upper | trend_lower | trend_upper | additive_terms | additive |
|---|---|---|---|---|---|---|---|
| **2016-06-01 00:00:00** | 44.273500 | 81.351485 | 97.788804 | 44.273500 | 44.273500 | 45.290820 | |
| **2016-06-01 01:00:00** | 44.274505 | 77.058148 | 94.298979 | 44.274505 | 44.274505 | 41.736647 | |
| **2016-06-01 02:00:00** | 44.275511 | 67.203060 | 84.755913 | 44.275511 | 44.275511 | 31.413187 | |
| **2016-06-01 03:00:00** | 44.276516 | 52.148689 | 68.480329 | 44.276516 | 44.276516 | 16.145999 | |
| **2016-06-01 04:00:00** | 44.277521 | 34.973064 | 51.801485 | 44.277521 | 44.277521 | -0.968848 | |

5 rows × 21 columns

In [90]:
```
# View the only the yhat,yhat_lower and yhat_upper columns from the DataFrame
forecast_mercado_trends[['yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[90]:

| ds | yhat | yhat_lower | yhat_upper |
|---|---|---|---|
| **2020-11-30 04:00:00** | 39.545289 | 30.390874 | 48.135032 |
| **2020-11-30 05:00:00** | 24.074127 | 15.408158 | 32.525066 |
| **2020-11-30 06:00:00** | 12.108426 | 3.234884 | 20.627130 |
| **2020-11-30 07:00:00** | 4.836238 | -4.210793 | 13.054987 |
| **2020-11-30 08:00:00** | 2.641312 | -5.682468 | 11.413799 |

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).

In [91]:
```
# 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
```

Out[91]:

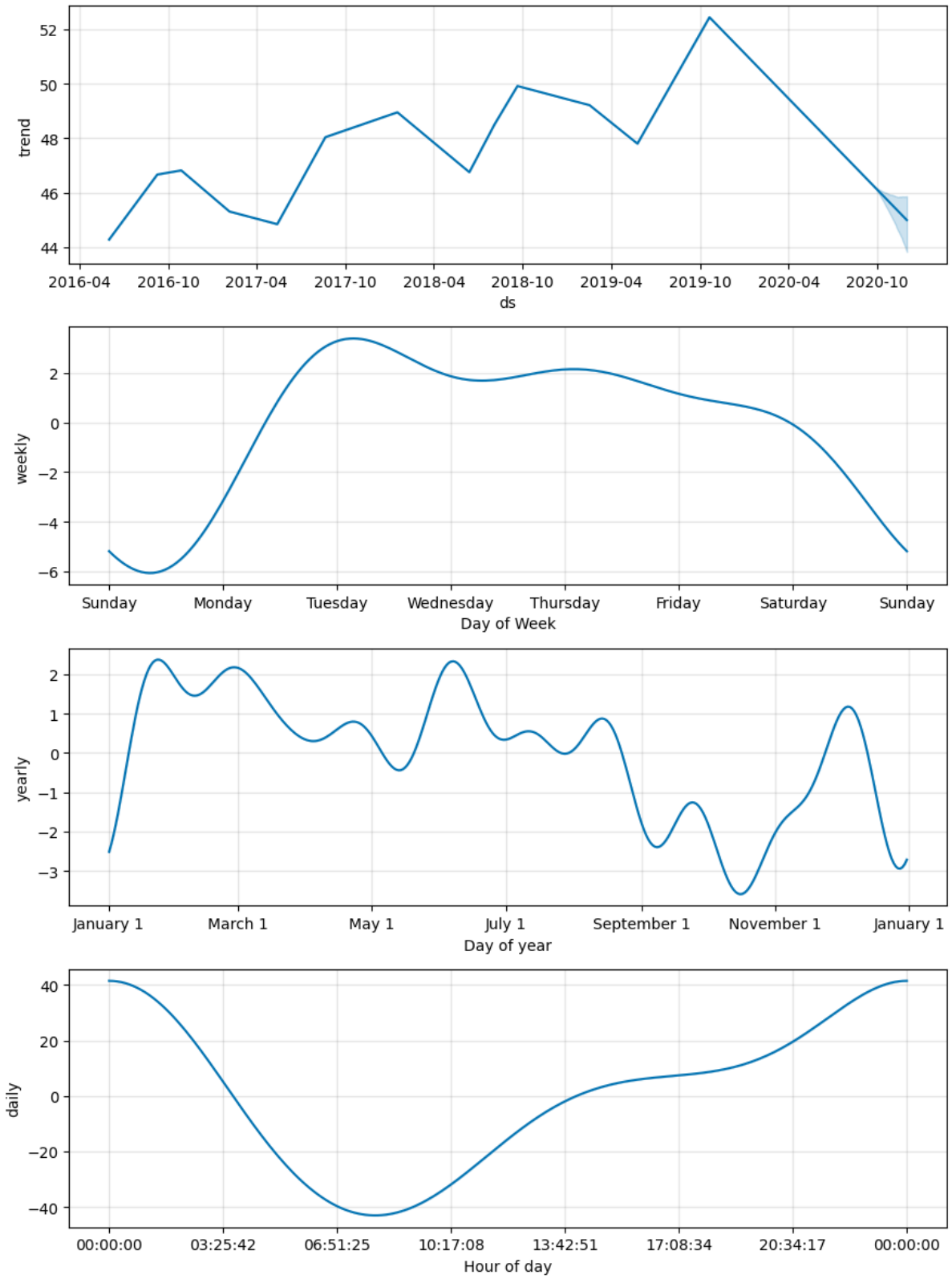
In [92]:
```
# Reset the index in the forecast_mercado_trends DataFrame
forecast_mercado_trends = forecast_mercado_trends.reset_index()
forecast_mercado_trends.head()
```

Out[92]:

| | ds | trend | yhat_lower | yhat_upper | trend_lower | trend_upper | additive_terms | additi |
|---|---|---|---|---|---|---|---|---|
| **0** | 2016-06-01 00:00:00 | 44.273500 | 81.351485 | 97.788804 | 44.273500 | 44.273500 | 45.290820 | |
| **1** | 2016-06-01 01:00:00 | 44.274505 | 77.058148 | 94.298979 | 44.274505 | 44.274505 | 41.736647 | |
| **2** | 2016-06-01 02:00:00 | 44.275511 | 67.203060 | 84.755913 | 44.275511 | 44.275511 | 31.413187 | |
| **3** | 2016-06-01 03:00:00 | 44.276516 | 52.148689 | 68.480329 | 44.276516 | 44.276516 | 16.145999 | |
| **4** | 2016-06-01 04:00:00 | 44.277521 | 34.973064 | 51.801485 | 44.277521 | 44.277521 | -0.968848 | |

5 rows × 22 columns

In [93]:
```
# 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_tren
```




Answer the following questions:

**Question:** What time of day exhibits the greatest popularity?

**Answer:** 00:00:00

**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:** November