1Final_Exam_Fall2021

December 15, 2021

1 Final exam - IST 652 - Notebook

Submitted by:

Date:

[1]:
```
%matplotlib inline

import pandas as pd
import numpy as np
import requests
from io import StringIO
from io import BytesIO
from zipfile import ZipFile
import matplotlib.pyplot as plt
np.set_printoptions(precision=4)
pd.options.display.max_rows = 20

!pip install plotly
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

```
Collecting plotly
  Downloading plotly-5.4.0-py2.py3-none-any.whl (25.3 MB)
     |                | 25.3 MB 5.0 MB/s eta 0:00:01
Collecting tenacity>=6.2.0
  Downloading tenacity-8.0.1-py3-none-any.whl (24 kB)
Requirement already satisfied: six in /opt/conda/lib/python3.9/site-packages
(from plotly) (1.16.0)
Installing collected packages: tenacity, plotly
Successfully installed plotly-5.4.0 tenacity-8.0.1
```

2 Loading data sets

2.1 Bus ridership data sets

[2]:
```
#Loading 2019 bus route SY36 dataset into Jupyter environment - a security␣
 ↪warning will appear. You can ignore it.
#Be patient - it could take up to 2 minutes for the dataset to become available
urldata19="https://gitlab.gitlab.svc.cent-su.org/ccaicedo/652public/-/raw/
 ↪master/datasets/busdata/BusActivity_SY36_2019.zip"
csvdata=requests.get(urldata19,verify=False).content

zf2019 = ZipFile(BytesIO(csvdata),'r')  #The dataset is being accessed from a␣
 ↪zip file so this step is needed.
```

```
/opt/conda/lib/python3.9/site-packages/urllib3/connectionpool.py:1013:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'gitlab.gitlab.svc.cent-su.org'. Adding certificate verification is strongly
advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-
warnings
  warnings.warn(
```

[3]:
```
#Dataframe with bus activity data for 2019 for route SY36 is named data2019
data2019=pd.read_csv(zf2019.open("Preprocessed_SY36_2019.csv"))
```

```
/opt/conda/lib/python3.9/site-packages/IPython/core/interactiveshell.py:3441:
DtypeWarning: Columns (23) have mixed types.Specify dtype option on import or
set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

[4]:
```
#Loading 2020 bus route SY36 dataset into Jupyter environment - a security␣
 ↪warning will appear. You can ignore it.
#Be patient - it could take up to 2 minutes for the dataset to become available
urldata20="https://gitlab.gitlab.svc.cent-su.org/ccaicedo/652public/-/raw/
 ↪master/datasets/busdata/BusActivity_SY36_2020.zip"
csvdata=requests.get(urldata20,verify=False).content

zf2020 = ZipFile(BytesIO(csvdata),'r')  #The dataset is being accessed from a␣
 ↪zip file so this step is needed.
```

```
/opt/conda/lib/python3.9/site-packages/urllib3/connectionpool.py:1013:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'gitlab.gitlab.svc.cent-su.org'. Adding certificate verification is strongly
advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-
warnings
  warnings.warn(
```

[5]:
```
#Dataframe with bus activity data for 2020 for route SY36 is named data2020
data2020=pd.read_csv(zf2020.open("Preprocessed_SY36_2020.csv"))
```

[6]:
```
#Drop some columns that won't be needed
data2019.
 ↪drop(['SURVEY_DATE','VEHICLE_DESCRIPTION','GARAGE_NAME','DIVISION_NAME','COMMENTS','WHEELCH
data2020.
 ↪drop(['SURVEY_DATE','VEHICLE_DESCRIPTION','GARAGE_NAME','DIVISION_NAME','COMMENTS','WHEELCH
```

[7]:
```
data2019.head()
```

[7]:

	SERIAL_NUMBER	SCHEDULE_ID	SCHEDULE_NAME	PATTERN_ID	ROUTE_NUMBER	\
0	2604260	295	Sep18 (Holiday)	180900591	371	
1	2604260	295	Sep18 (Holiday)	180900591	371	
2	2604260	295	Sep18 (Holiday)	180900591	371	
3	2604260	295	Sep18 (Holiday)	180900591	371	
4	2604260	295	Sep18 (Holiday)	180900591	371	

	ROUTE_NAME	DIRECTION_NAME	BRANCH	\
0	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
1	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
2	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
3	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
4	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	

	TRIP_START_TIME	TIME_PERIOD	…	DWELL_TIME	\
0	2019-01-01 07:40:00.000000	AM Peak	…	NaN	
1	2019-01-01 07:40:00.000000	AM Peak	…	0.0	
2	2019-01-01 07:40:00.000000	AM Peak	…	0.0	
3	2019-01-01 07:40:00.000000	AM Peak	…	0.0	
4	2019-01-01 07:40:00.000000	AM Peak	…	0.0	

	RUNNING_TIME_ACTUAL	PASSENGERS_ON	PASSENGERS_OFF	PASSENGERS_IN	\
0	5.483	8	0	8	
1	NaN	0	0	8	
2	2.550	0	0	8	
3	NaN	0	0	8	
4	NaN	0	0	8	

	TIMEPOINT_MILES	FIRST_LAST_STOP	UNIQUE_ID	stop_lat	stop_lon
0	0.413	1	37100000002	43.043656	-76.150963
1	NaN	2	37100000003	43.044280	-76.147495
2	0.716	2	37100000005	43.045336	-76.147419
3	NaN	2	37100000006	43.047959	-76.147440
4	NaN	2	37100000007	43.049554	-76.148697

[5 rows x 40 columns]

[8]:
```
data2020.head()
```

[8]:
```
   SERIAL_NUMBER  SCHEDULE_ID    SCHEDULE_NAME  PATTERN_ID  ROUTE_NUMBER  \
0        3286134          314  Dec19 (Holiday)   191200591           371
1        3286134          314  Dec19 (Holiday)   191200591           371
2        3286134          314  Dec19 (Holiday)   191200591           371
3        3286134          314  Dec19 (Holiday)   191200591           371
4        3286134          314  Dec19 (Holiday)   191200591           371

  ROUTE_NAME DIRECTION_NAME                        BRANCH  \
0       SY36       FROM HUB  [sy36]Outbound 136 no plazas
1       SY36       FROM HUB  [sy36]Outbound 136 no plazas
2       SY36       FROM HUB  [sy36]Outbound 136 no plazas
3       SY36       FROM HUB  [sy36]Outbound 136 no plazas
4       SY36       FROM HUB  [sy36]Outbound 136 no plazas

              TRIP_START_TIME TIME_PERIOD  … DWELL_TIME  \
0  2020-01-01 07:40:00.000000     AM Peak  …        NaN
1  2020-01-01 07:40:00.000000     AM Peak  …       0.00
2  2020-01-01 07:40:00.000000     AM Peak  …       0.00
3  2020-01-01 07:40:00.000000     AM Peak  …       0.00
4  2020-01-01 07:40:00.000000     AM Peak  …       0.12

   RUNNING_TIME_ACTUAL  PASSENGERS_ON  PASSENGERS_OFF  PASSENGERS_IN  \
0                5.533              9               0              9
1                  NaN              0               0              9
2                4.200              0               0              9
3                  NaN              0               0              9
4                  NaN              1               0             10

  TIMEPOINT_MILES  FIRST_LAST_STOP    UNIQUE_ID   stop_lat   stop_lon
0           0.417                1  37100000002  43.043656 -76.150963
1             NaN                2  37100000003  43.044280 -76.147495
2           0.705                2  37100000005  43.045336 -76.147419
3             NaN                2  37100000006  43.047959 -76.147440
4             NaN                2  37100000007  43.049554 -76.148697

[5 rows x 40 columns]
```

[9]:
```
data2019.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 980551 entries, 0 to 980550
Data columns (total 40 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   SERIAL_NUMBER        980551 non-null  int64
 1   SCHEDULE_ID          980551 non-null  int64
 2   SCHEDULE_NAME        980551 non-null  object
```

```
 3   PATTERN_ID           980551 non-null  int64
 4   ROUTE_NUMBER         980551 non-null  int64
 5   ROUTE_NAME           980551 non-null  object
 6   DIRECTION_NAME       980551 non-null  object
 7   BRANCH               980551 non-null  object
 8   TRIP_START_TIME      980551 non-null  object
 9   TIME_PERIOD          980551 non-null  object
 10  SERVICE_PERIOD       980551 non-null  object
 11  TRIP_NUMBER          980551 non-null  int64
 12  TRIP_KEY             980551 non-null  int64
 13  BLOCK_NUMBER         980551 non-null  int64
 14  BLOCK_KEY            980551 non-null  int64
 15  BLOCK_NAME           980551 non-null  object
 16  RUN_NUMBER           980551 non-null  int64
 17  RUN_KEY              980551 non-null  int64
 18  VEHICLE_NUMBER       980551 non-null  int64
 19  VEHICLE_SEATS        980551 non-null  int64
 20  OPERATOR_ID          980551 non-null  int64
 21  SORT_ORDER           980551 non-null  int64
 22  STOP_ID              980551 non-null  int64
 23  MAIN_CROSS_STREET    980551 non-null  object
 24  TRAVEL_DIRECTION     980551 non-null  object
 25  TIMEPOINT            980551 non-null  int64
 26  SEGMENT_MILES        980551 non-null  float64
 27  TIME_SCHEDULED       99147 non-null   object
 28  TIME_ACTUAL_ARRIVE   980551 non-null  object
 29  TIME_ACTUAL_DEPART   980551 non-null  object
 30  DWELL_TIME           947450 non-null  float64
 31  RUNNING_TIME_ACTUAL  83881 non-null   float64
 32  PASSENGERS_ON        980551 non-null  int64
 33  PASSENGERS_OFF       980551 non-null  int64
 34  PASSENGERS_IN        980551 non-null  int64
 35  TIMEPOINT_MILES      94690 non-null   float64
 36  FIRST_LAST_STOP      980551 non-null  int64
 37  UNIQUE_ID            980551 non-null  int64
 38  stop_lat             980259 non-null  float64
 39  stop_lon             980259 non-null  float64
dtypes: float64(6), int64(21), object(13)
memory usage: 299.2+ MB
```

[10]:
```
data2020.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 962566 entries, 0 to 962565
Data columns (total 40 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   SERIAL_NUMBER        962566 non-null  int64
```

```
 1   SCHEDULE_ID          962566 non-null  int64
 2   SCHEDULE_NAME        962566 non-null  object
 3   PATTERN_ID           962566 non-null  int64
 4   ROUTE_NUMBER         962566 non-null  int64
 5   ROUTE_NAME           962566 non-null  object
 6   DIRECTION_NAME       962566 non-null  object
 7   BRANCH               962566 non-null  object
 8   TRIP_START_TIME      962566 non-null  object
 9   TIME_PERIOD          962566 non-null  object
 10  SERVICE_PERIOD       962566 non-null  object
 11  TRIP_NUMBER          962566 non-null  int64
 12  TRIP_KEY             962566 non-null  int64
 13  BLOCK_NUMBER         962566 non-null  int64
 14  BLOCK_KEY            962566 non-null  int64
 15  BLOCK_NAME           962566 non-null  object
 16  RUN_NUMBER           962566 non-null  int64
 17  RUN_KEY              962566 non-null  int64
 18  VEHICLE_NUMBER       962566 non-null  int64
 19  VEHICLE_SEATS        962566 non-null  int64
 20  OPERATOR_ID          962566 non-null  int64
 21  SORT_ORDER           962566 non-null  int64
 22  STOP_ID              962566 non-null  int64
 23  MAIN_CROSS_STREET    962566 non-null  object
 24  TRAVEL_DIRECTION     962566 non-null  object
 25  TIMEPOINT            962566 non-null  int64
 26  SEGMENT_MILES        962566 non-null  float64
 27  TIME_SCHEDULED       97912 non-null   object
 28  TIME_ACTUAL_ARRIVE   962566 non-null  object
 29  TIME_ACTUAL_DEPART   962566 non-null  object
 30  DWELL_TIME           915859 non-null  float64
 31  RUNNING_TIME_ACTUAL  77090 non-null   float64
 32  PASSENGERS_ON        962566 non-null  int64
 33  PASSENGERS_OFF       962566 non-null  int64
 34  PASSENGERS_IN        962566 non-null  int64
 35  TIMEPOINT_MILES      97912 non-null   float64
 36  FIRST_LAST_STOP      962566 non-null  int64
 37  UNIQUE_ID            962566 non-null  int64
 38  stop_lat             161343 non-null  float64
 39  stop_lon             161343 non-null  float64
dtypes: float64(6), int64(21), object(13)
memory usage: 293.8+ MB
```

2.2 2019 Syracuse weather data

[11]:
```
#Loading Syracuse Weather dataset into Jupyter environment - a security warning␣
↪will appear. You can ignore it.
```

```
url_weatherdata="https://gitlab.gitlab.svc.cent-su.org/ccaicedo/652public/-/raw/
 ↪master/syracuse_2019_weather.csv"
csvweatherdata=requests.get(url_weatherdata,verify=False).text  #this will␣
 ↪generate a warning but you can proceed
```

```
/opt/conda/lib/python3.9/site-packages/urllib3/connectionpool.py:1013:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'gitlab.gitlab.svc.cent-su.org'. Adding certificate verification is strongly
advised. See: https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-
warnings
  warnings.warn(
```

[12]:
```
#Setup the weather_2019 dataframe with the data from the weather dataset
#You still need to set the column that will be the index
weather_2019=pd.read_csv(StringIO(csvweatherdata))
```

[13]:
```
weather_2019.head()
```

[13]:

	STATION	NAME	DATE	AWND	PRCP	SNOW	TAVG	TMAX	TMIN
0	USW00014771	SYRACUSE HANCOCK INTERNATIONAL AIRPORT, NY US	1/1/2019	15.66	0.02	0.0	40	53	21
1	USW00014771	SYRACUSE HANCOCK INTERNATIONAL AIRPORT, NY US	1/2/2019	5.14	0.00	0.0	24	31	18
2	USW00014771	SYRACUSE HANCOCK INTERNATIONAL AIRPORT, NY US	1/3/2019	10.74	0.09	0.3	33	37	30
3	USW00014771	SYRACUSE HANCOCK INTERNATIONAL AIRPORT, NY US	1/4/2019	4.70	0.00	0.0	36	49	25
4	USW00014771	SYRACUSE HANCOCK INTERNATIONAL AIRPORT, NY US	1/5/2019	5.59	0.00	0.0	33	44	25

3 Exam task solutions

Add the text/code/visualizations for your exam tasks solutions from this point onwards. Use as many additional cells as required. Please place long textual explanations or analysis in their own markdown cells, not as comments inside your code cells.

3.0.1 Task 1 (30 points): For the year of 2019, determine the number of passengers that board thebus (PASSENGERS_ON) at a particular STOP_ID per day. Use this data to understand how the changes in weather affect the ridership at your selected Bus Stop. Select a bus stop with a daily annual average of at least 5 passengers using it (This means that for any service day of the year, on average, at least 5 passengers boarded the bus from that bus stop).

[14]:
```
## converting the object TRIP_START_TIME into datetime.
data2019['TRIP_START_TIME'] = pd.to_datetime(data2019['TRIP_START_TIME'])
```

[15]:
```
data2019['TRIP_START_TIME_YYYY_DD_MM'] = data2019['TRIP_START_TIME'].dt.date
data2019['TRIP_START_TIME_YYYY_DD_MM'] = pd.
 ↪to_datetime(data2019['TRIP_START_TIME_YYYY_DD_MM'])
```

[16]:
```
## Number of passengers boarding at all stop ids per day
cols = ['TRIP_START_TIME_YYYY_DD_MM', 'STOP_ID']
df_pass_count = data2019.groupby(cols)[["PASSENGERS_ON"]].sum()
df_pass_count.head(1000)
```

[16]:

		PASSENGERS_ON
TRIP_START_TIME_YYYY_DD_MM	STOP_ID	
2019-01-01	100	0
	611	8
	612	6
	619	0
	621	0
…		…
2019-01-07	740	1
	747	0
	750	0
	751	1
	757	0

[1000 rows x 1 columns]

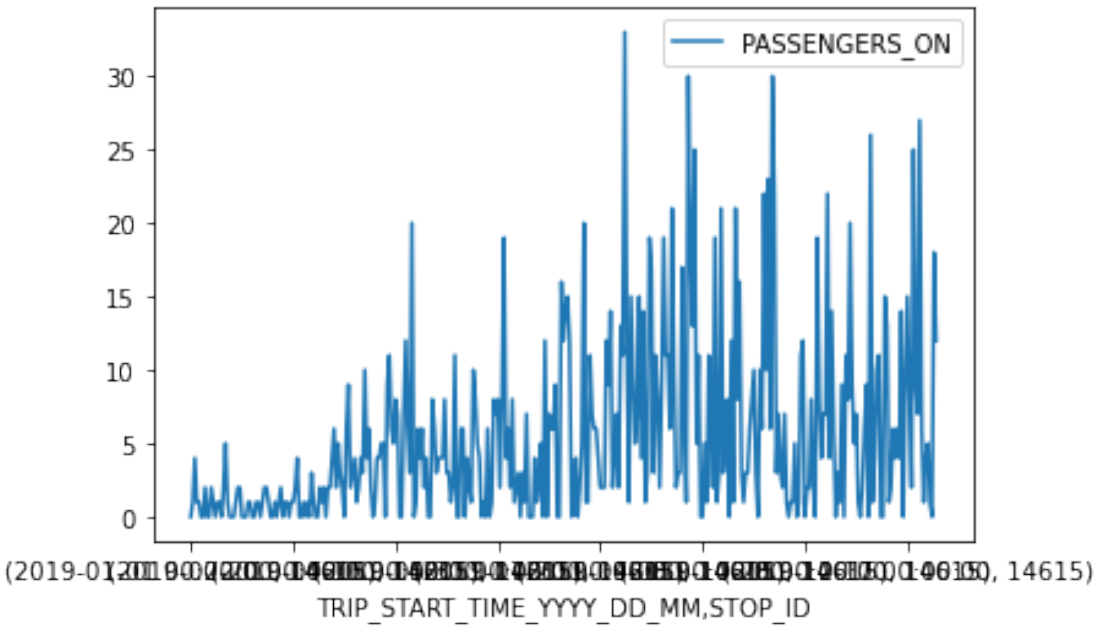
[17]:
```
df_particular_14615_passengers = data2019.loc[(data2019['STOP_ID'] == 14615) &␣
 ↪(data2019['PASSENGERS_ON'] >= 0)]
df_particular_14615_passengers_1 = df_particular_14615_passengers.
 ↪groupby(cols)[["PASSENGERS_ON"]].sum()
df_particular_14615_passengers_1
```

[17]:

		PASSENGERS_ON
TRIP_START_TIME_YYYY_DD_MM	STOP_ID	
2019-01-01	14615	0
2019-01-02	14615	1
2019-01-03	14615	4
2019-01-04	14615	1

```
2019-01-05                  14615                  1
…                                                  …
2019-12-27                  14615                  5
2019-12-28                  14615                  1
2019-12-29                  14615                  0
2019-12-30                  14615                 18
2019-12-31                  14615                 12

[365 rows x 1 columns]
```

[19]:
```
df_particular_14615_passengers_1.plot.line()
plt.show()
```



[21]:
```
# monthly
df_particular_stop_id_passengers_m = df_particular_14615_passengers.groupby(pd.
 ↪Grouper(freq='M', key='TRIP_START_TIME_YYYY_DD_MM')).mean()
# daily
davg_df2 = df_particular_14615_passengers.groupby(pd.Grouper(freq='D',␣
 ↪key='TRIP_START_TIME_YYYY_DD_MM')).mean()
df_gt3 = davg_df2.loc[(davg_df2['PASSENGERS_ON'] > 3) ]
df_gt3['STOP_ID']
```

[21]:
```
TRIP_START_TIME_YYYY_DD_MM
2019-09-01    14615.0
2019-10-12    14615.0
Freq: 41D, Name: STOP_ID, dtype: float64
```

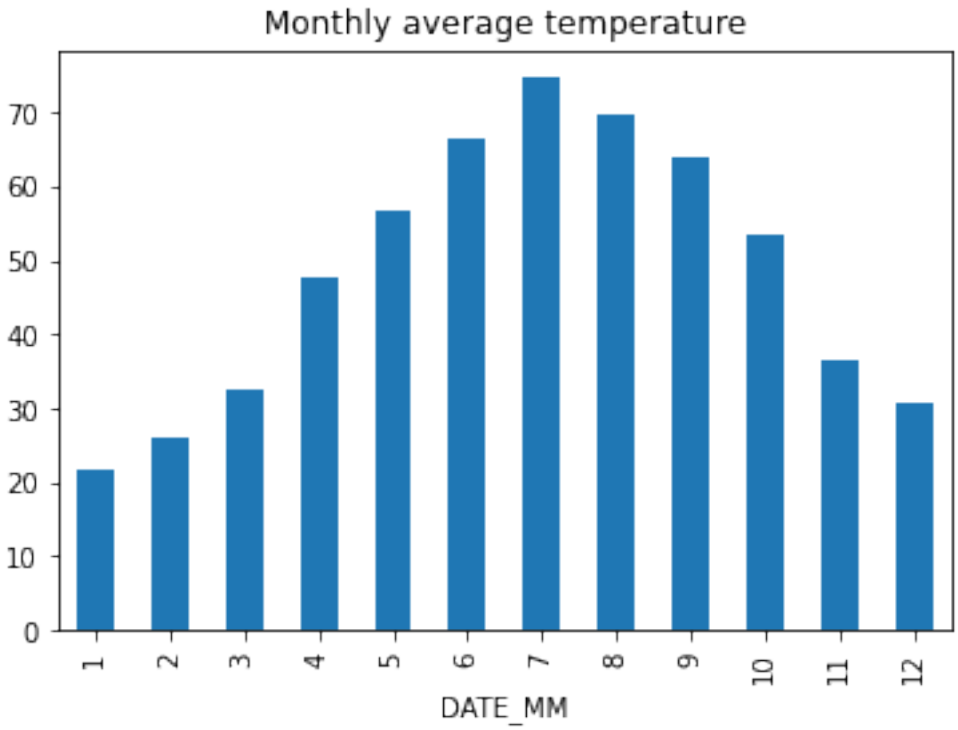
[23]:
```
fig=go.Figure()
fig.add_trace(go.Scatter(x=df_particular_14615_passengers.
 ↪TRIP_START_TIME_YYYY_DD_MM,␣
 ↪y=df_particular_14615_passengers["PASSENGERS_ON"],
                    mode='lines',
                    name=' Passenger count for stop id 14615'))
fig.update_layout(title=" Passenger count for stop id 14615",
                  xaxis_title="Date",yaxis_title=" Passenger count for stop id␣
 ↪14615",legend=dict(x=0,y=1,traceorder="normal"))

fig.show()
```

[24]:
```
#how the changes in weather affect the ridership at your selected Bus Stop
weather_2019['DATE'] = pd.to_datetime(weather_2019['DATE'])
weather_2019['DATE_dd'] = weather_2019['DATE'].dt.date


#finding average temperature monthly

DATE_dd = weather_2019['DATE_dd']
df1 = weather_2019.groupby(DATE_dd)['TAVG'].mean()


df1.index
```

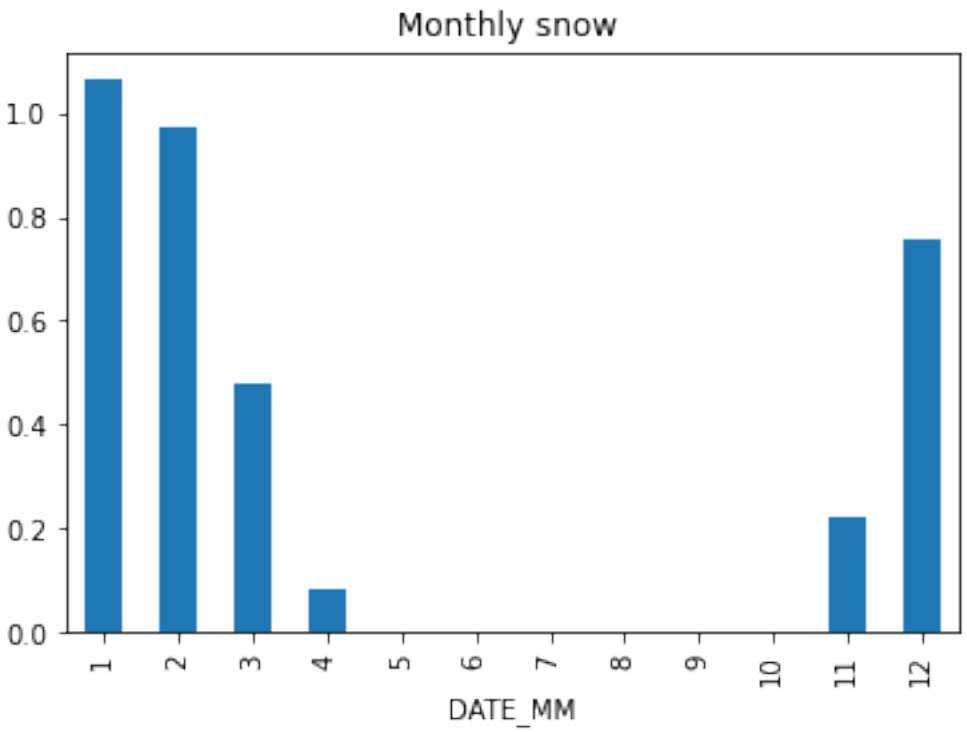
[24]:
```
Index([2019-01-01, 2019-01-02, 2019-01-03, 2019-01-04, 2019-01-05, 2019-01-06,
       2019-01-07, 2019-01-08, 2019-01-09, 2019-01-10,
       …
       2019-12-22, 2019-12-23, 2019-12-24, 2019-12-25, 2019-12-26, 2019-12-27,
       2019-12-28, 2019-12-29, 2019-12-30, 2019-12-31],
      dtype='object', name='DATE_dd', length=365)
```

[26]:
```
fig=go.Figure()
fig.add_trace(go.Scatter(x=df_particular_14615_passengers.
 ↪TRIP_START_TIME_YYYY_DD_MM,␣
 ↪y=df_particular_14615_passengers["PASSENGERS_ON"],
                    mode='lines',
                    name='Passenger count for stop id 14615'))
fig.add_trace(go.Scatter(x=df1.index, y=df1,mode='lines',
                    name=' Temperature AVG'))
fig.update_layout(title=" Temperature vs Passenger count for stop id 14615",
                  xaxis_title="Date",yaxis_title=" Temperature vs Passenger␣
 ↪count for stop id 14615",legend=dict(x=0,y=1,traceorder="normal"))

fig.show()
```

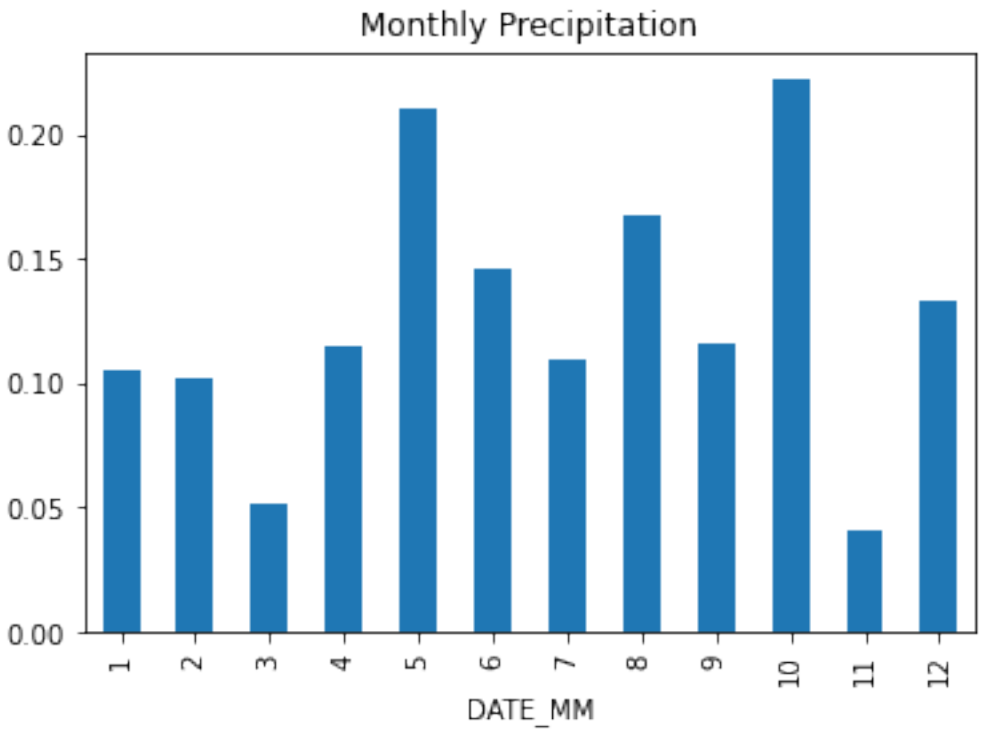
[27]:
```
#how the changes in weather affect the ridership at your selected Bus Stop
weather_2019['DATE'] = pd.to_datetime(weather_2019['DATE'])
weather_2019['DATE_MM'] = weather_2019['DATE'].dt.month


#finding average temperature monthly

DATE_MM = weather_2019['DATE_MM']
df1 = weather_2019.groupby(DATE_MM)['TAVG'].mean()


df1.plot.bar()
plt.title("Monthly average temperature")
plt.show()
```



[28]:
```
#finding average Snow monthly
DATE_MM = weather_2019['DATE_MM']
df2 = weather_2019.groupby(DATE_MM)['SNOW'].mean()

df2.plot.bar()
plt.title("Monthly snow ")
plt.show()
```



[29]:
```
#finding average precipitation monthly
DATE_MM = weather_2019['DATE_MM']
df2 = weather_2019.groupby(DATE_MM)['PRCP'].mean()

df2.plot.bar()
plt.title("Monthly Precipitation")
plt.show()
```



3.0.2 Weather and ridership relationship

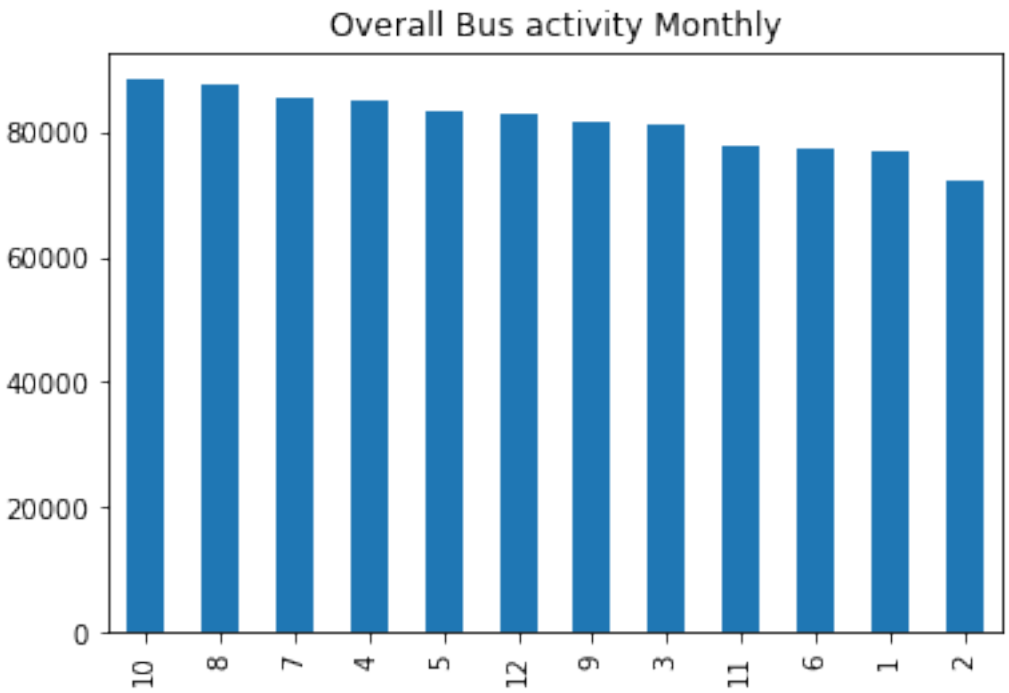
From the above analysis it is clear that, as the temperature decreases the bus ridership decreases. In january to march where the temperature was lowest, the bus ridership was the least.

As the snow increases, the bus ridership decreases. In the month of January to march where snow is maximum the bus ridership at 14615 is the least.

Also, one fact to consider is that the winter break might have also affected the bus ridership as many of the people go on holidays.

3.0.3 Subtask 1.1 (+10 points): Group the activity at the selected bus stop per month and compare against the average temperature for that month

[30]:
```
## Overall Bus activity Monthly

data2019['Date_MM'] = data2019['TRIP_START_TIME_YYYY_DD_MM'].dt.month
month_count = data2019['Date_MM'].value_counts()
month_count.plot.bar()
plt.title("Overall Bus activity Monthly")
plt.show()
```



[31]:
```
## Number of passengers boarding at all stop ids per day Monthly
cols = ['Date_MM', 'STOP_ID']
df_pass_count_monthly = data2019.groupby(cols)[["PASSENGERS_ON"]].sum()
df_pass_count_monthly
```

[31]:

Date_MM	STOP_ID	PASSENGERS_ON
1	100	11
	611	398
	612	295
	619	10
	621	40
…		…
12	17661	5264
	17676	146
	17677	311
	17823	39
	17824	6

[1933 rows x 1 columns]

[32]:
```
df_particular_14615_passengers_monthly = data2019.loc[(data2019['STOP_ID'] ==␣
↪14615) & (data2019['PASSENGERS_ON'] >= 0)]
```

```
df_particular_14615_passengers_monthly_1 =␣
 ↪df_particular_14615_passengers_monthly.groupby(cols)[["PASSENGERS_ON"]].sum()
df_particular_14615_passengers_monthly_1
```

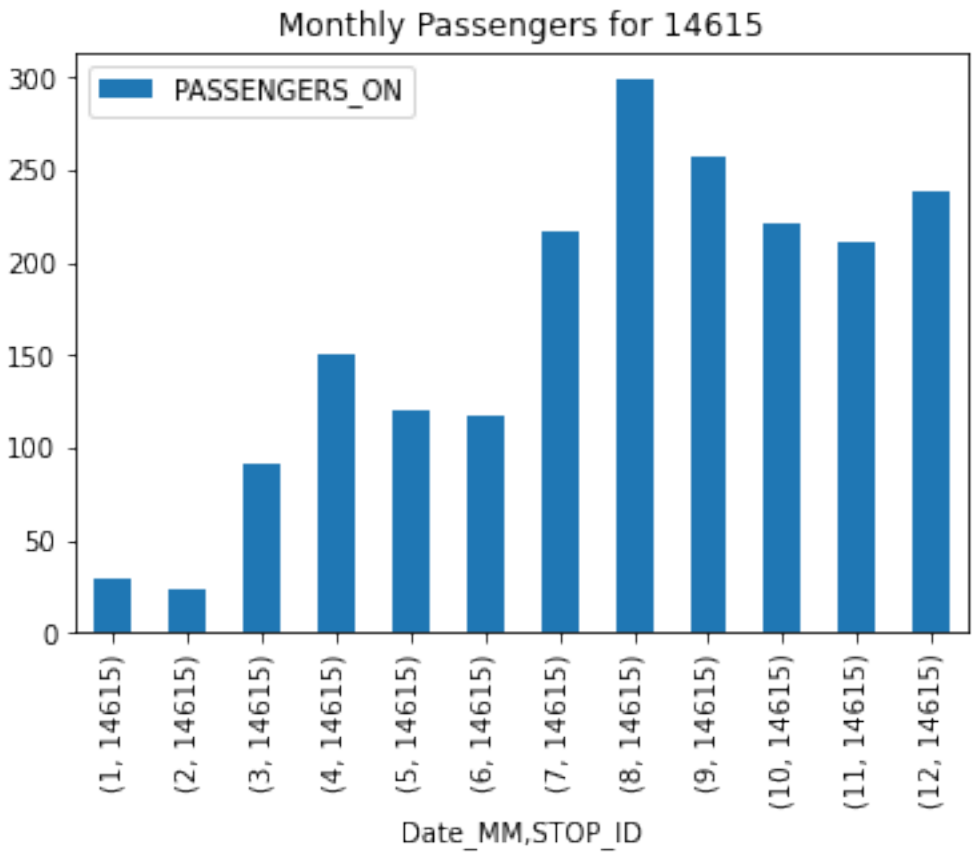
[32]:

Date_MM	STOP_ID	PASSENGERS_ON
1	14615	29
2	14615	24
3	14615	91
4	14615	150
5	14615	120
6	14615	118
7	14615	217
8	14615	299
9	14615	257
10	14615	222
11	14615	211
12	14615	239

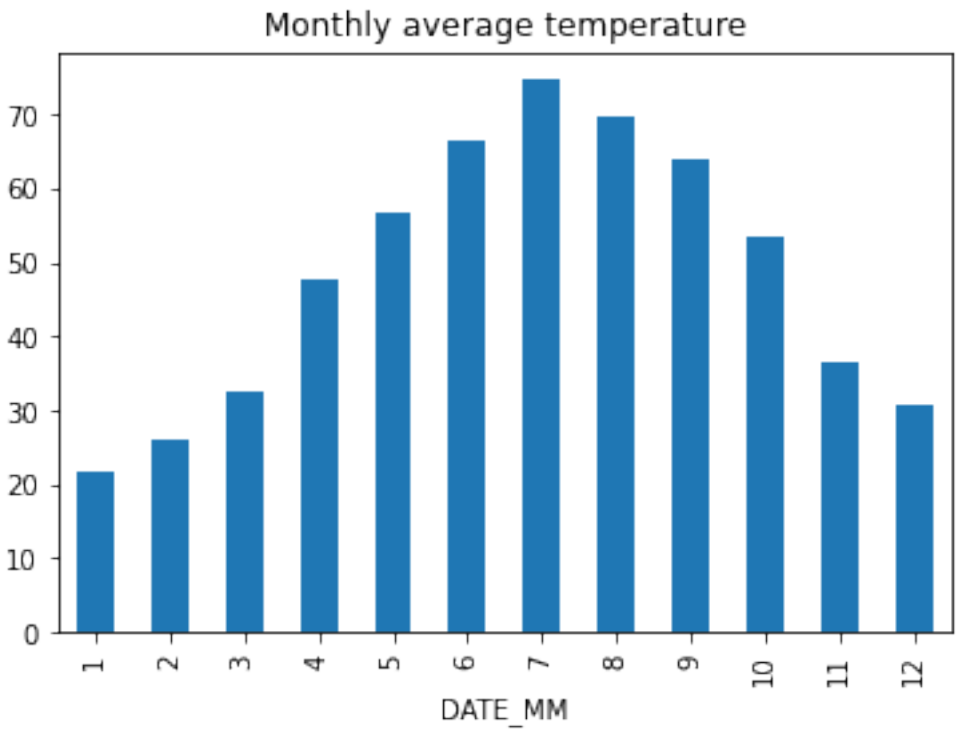
[33]:
```
df_particular_14615_passengers_monthly.columns
```

[33]:
```
Index(['SERIAL_NUMBER', 'SCHEDULE_ID', 'SCHEDULE_NAME', 'PATTERN_ID',
       'ROUTE_NUMBER', 'ROUTE_NAME', 'DIRECTION_NAME', 'BRANCH',
       'TRIP_START_TIME', 'TIME_PERIOD', 'SERVICE_PERIOD', 'TRIP_NUMBER',
       'TRIP_KEY', 'BLOCK_NUMBER', 'BLOCK_KEY', 'BLOCK_NAME', 'RUN_NUMBER',
       'RUN_KEY', 'VEHICLE_NUMBER', 'VEHICLE_SEATS', 'OPERATOR_ID',
       'SORT_ORDER', 'STOP_ID', 'MAIN_CROSS_STREET', 'TRAVEL_DIRECTION',
       'TIMEPOINT', 'SEGMENT_MILES', 'TIME_SCHEDULED', 'TIME_ACTUAL_ARRIVE',
       'TIME_ACTUAL_DEPART', 'DWELL_TIME', 'RUNNING_TIME_ACTUAL',
       'PASSENGERS_ON', 'PASSENGERS_OFF', 'PASSENGERS_IN', 'TIMEPOINT_MILES',
       'FIRST_LAST_STOP', 'UNIQUE_ID', 'stop_lat', 'stop_lon',
       'TRIP_START_TIME_YYYY_DD_MM', 'Date_MM'],
      dtype='object')
```

[34]:
```
df_particular_14615_passengers_monthly_1.plot.bar()
plt.title("Monthly Passengers for 14615")
plt.show()
```



[35]:
```
#how the changes in weather affect the ridership at your selected Bus Stop
weather_2019['DATE'] = pd.to_datetime(weather_2019['DATE'])
weather_2019['DATE_MM'] = weather_2019['DATE'].dt.month


#finding average temperature monthly

DATE_MM = weather_2019['DATE_MM']
df1 = weather_2019.groupby(DATE_MM)['TAVG'].mean()


df1.plot.bar()
plt.title("Monthly average temperature")
plt.show()
```



[36]:
```
fig=go.Figure()
fig.add_trace(go.Scatter(x=df1.index,␣
 ↪y=df_particular_14615_passengers_monthly_1["PASSENGERS_ON"],
                    mode='lines',
                    name='Passenger count for stop id 14615'))
fig.add_trace(go.Scatter(x=df1.index, y=df1,mode='lines',
                    name=' Temperature AVG in fahrenheit'))
fig.update_layout(title=" Temperature in fahrenheit vs Passenger count for stop␣
 ↪id 14615",
                  xaxis_title="Month of the year",yaxis_title=" Temperature vs␣
 ↪Passenger count for stop id 14615",legend=dict(x=0,y=1,traceorder="normal"))

fig.show()
```

3.0.4 as the temperature decreases, Passenger count also decreases and vice versa

[ ]:
```
```

3.0.5 Subtask 1.2 (+10 points): Compare the activity between 2 or more bus stops over each month of the year

[37]:
```
df_particular_14615_passengers_monthly_14615 = data2019.
 ↪loc[(data2019['STOP_ID'] == 14615) & (data2019['PASSENGERS_ON'] >= 0)]
df_particular_14615_passengers_monthly_14615_1 =␣
 ↪df_particular_14615_passengers_monthly_14615.
 ↪groupby(cols)[["PASSENGERS_ON"]].sum()
df_particular_14615_passengers_monthly_14615_1
```

[37]:

Date_MM	STOP_ID	PASSENGERS_ON
1	14615	29
2	14615	24
3	14615	91
4	14615	150
5	14615	120
6	14615	118
7	14615	217
8	14615	299
9	14615	257
10	14615	222
11	14615	211
12	14615	239

[38]:
```
df_particular_passengers_monthly_17661 = data2019.loc[(data2019['STOP_ID'] ==␣
 ↪17661) & (data2019['PASSENGERS_ON'] >= 0)]
df_particular_passengers_monthly_17661_1 =␣
 ↪df_particular_passengers_monthly_17661.groupby(cols)[["PASSENGERS_ON"]].sum()
df_particular_passengers_monthly_17661_1
```

[38]:

Date_MM	STOP_ID	PASSENGERS_ON
1	17661	4264
2	17661	4111
3	17661	4876
4	17661	5284
5	17661	5078
6	17661	4498
7	17661	5068
8	17661	5735
9	17661	5901
10	17661	6042
11	17661	5146
12	17661	5264

[39]:
```
fig=go.Figure()
fig.add_trace(go.Scatter(x=df1.index,␣
 ↪y=df_particular_14615_passengers_monthly_14615_1["PASSENGERS_ON"],
                    mode='lines',
                    name='Passenger count for stop id 14615'))
fig.add_trace(go.Scatter(x=df1.index,␣
 ↪y=df_particular_passengers_monthly_17661_1["PASSENGERS_ON"],mode='lines',
                    name='Passenger count for stop id 17661'))
fig.update_layout(title=" Passenger count for stop id 17661 vs Passenger count␣
 ↪for stop id 14615",
                  xaxis_title="Month of the year",yaxis_title="  Passenger count␣
 ↪for stop id 17661 vs 14615",legend=dict(x=0,y=1,traceorder="normal"))

fig.show()
```

[ ]:
```

```

3.0.6 Subtask 1.3 (+10 points): Determine the 5 bus stops that provide the highest average number of daily passengers during the year

[40]:
```
## Top 5 bus stops with highest average daily passengers
cols = ['TRIP_START_TIME_YYYY_DD_MM', 'STOP_ID']
df_pass_count = data2019.groupby(cols)[["PASSENGERS_ON"]].mean()
means = data2019.groupby('STOP_ID').mean()
df_pass_count.nlargest(10000,'PASSENGERS_ON')
```

[40]:

TRIP_START_TIME_YYYY_DD_MM	STOP_ID	PASSENGERS_ON
2019-08-31	17661	10.533333
2019-11-02	17661	10.333333
2019-11-09	17661	9.533333
2019-11-16	17661	9.363636
2019-09-02	17661	9.307692
…		…
2019-05-13	14616	0.225806
2019-06-18	3758	0.225806
2019-09-17	14616	0.225806
2019-10-08	3758	0.225806
2019-10-18	14615	0.225806

[10000 rows x 1 columns]

3.0.7 5 bus stops that provide the highest average number of daily passengers during the year are 3761, 1114, 17661, 14616, 14615

[ ]:

3.0.8 Task 2 (30 points): For the years of 2019 and 2020, determine the number of passengers that board the bus (PASSENGERS_ON) at a particular STOP_ID per week. Select a bus stop where you would expect a high number of users (i.e. Near a shopping mall, school, hospital, etc). Compare the ridership activity between the two years and mention any hypothesis as to why changes could have taken place.

[66]:
```
data2019['TRIP_START_TIME_WW'] = data2019['TRIP_START_TIME_YYYY_DD_MM'].dt.
↪strftime('%U')
data2019
```

[66]:

	SERIAL_NUMBER	SCHEDULE_ID	SCHEDULE_NAME	PATTERN_ID	ROUTE_NUMBER	\
0	2604260	295	Sep18 (Holiday)	180900591	371	
1	2604260	295	Sep18 (Holiday)	180900591	371	
2	2604260	295	Sep18 (Holiday)	180900591	371	
3	2604260	295	Sep18 (Holiday)	180900591	371	
4	2604260	295	Sep18 (Holiday)	180900591	371	
…	…	…	…	…	…	
980546	3290357	317	Dec19 (Weekday)	191201024	371	
980547	3290357	317	Dec19 (Weekday)	191201024	371	
980548	3290357	317	Dec19 (Weekday)	191201024	371	
980549	3290357	317	Dec19 (Weekday)	191201024	371	
980550	3290357	317	Dec19 (Weekday)	191201024	371	

	ROUTE_NAME	DIRECTION_NAME	BRANCH	\
0	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
1	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
2	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
3	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
4	SY36	FROM HUB	[Sy36]Outbound 136 no plazas	
…	…	…	…	
980546	SY36	TO HUB	[sy36]Inbound 136 to B18	
980547	SY36	TO HUB	[sy36]Inbound 136 to B18	
980548	SY36	TO HUB	[sy36]Inbound 136 to B18	
980549	SY36	TO HUB	[sy36]Inbound 136 to B18	
980550	SY36	TO HUB	[sy36]Inbound 136 to B18	

	TRIP_START_TIME	TIME_PERIOD	…	PASSENGERS_OFF	PASSENGERS_IN	\
0	2019-01-01 07:40:00	AM Peak	…	0	8	
1	2019-01-01 07:40:00	AM Peak	…	0	8	
2	2019-01-01 07:40:00	AM Peak	…	0	8	
3	2019-01-01 07:40:00	AM Peak	…	0	8	

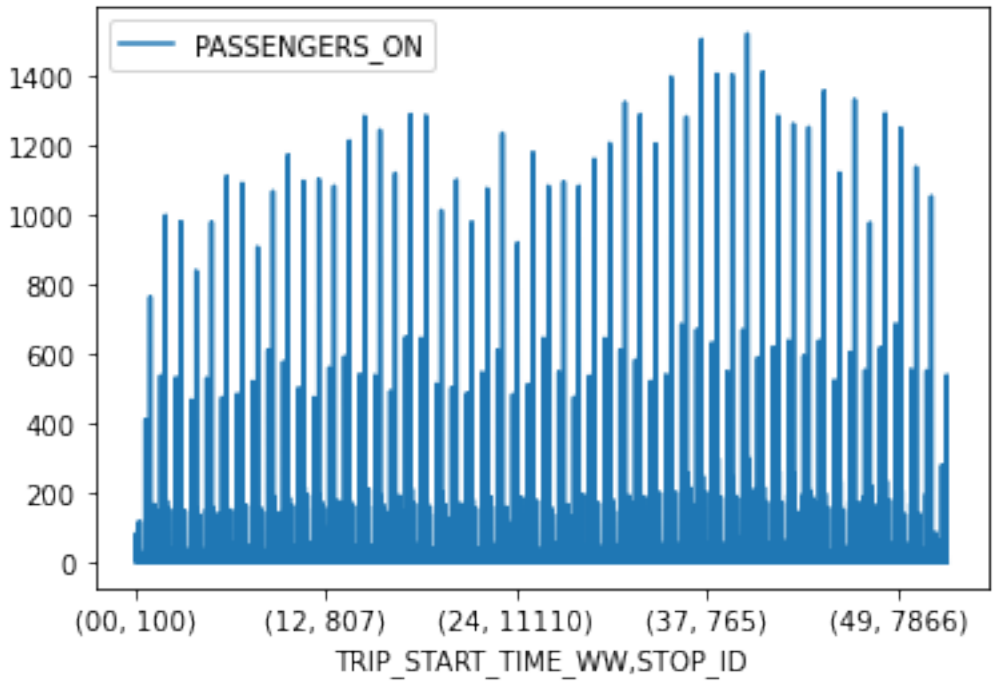
```
4      2019-01-01 07:40:00     AM Peak  …              0              8
…                      …         …  …            …            …
980546 2019-12-31 17:20:00     PM Peak  …              0             11
980547 2019-12-31 17:20:00     PM Peak  …              0             11
980548 2019-12-31 17:20:00     PM Peak  …              0             11
980549 2019-12-31 17:20:00     PM Peak  …              0             11
980550 2019-12-31 17:20:00     PM Peak  …             11              0

        TIMEPOINT_MILES  FIRST_LAST_STOP    UNIQUE_ID   stop_lat   stop_lon  \
0                 0.413                1  37100000002  43.043656 -76.150963
1                   NaN                2  37100000003  43.044280 -76.147495
2                 0.716                2  37100000005  43.045336 -76.147419
3                   NaN                2  37100000006  43.047959 -76.147440
4                   NaN                2  37100000007  43.049554 -76.148697
…                   …                …           …        …        …
980546              NaN                2  37101125029  43.047136 -76.147503
980547            0.349                2  37101125031  43.045623 -76.147629
980548              NaN                2  37101125032  43.044405 -76.147532
980549              NaN                2  37101125033  43.044344 -76.148716
980550            0.000                3  37101125035  43.043047 -76.151260

        TRIP_START_TIME_YYYY_DD_MM  Date_MM  TRIP_START_TIME_WW
0                       2019-01-01        1                  00
1                       2019-01-01        1                  00
2                       2019-01-01        1                  00
3                       2019-01-01        1                  00
4                       2019-01-01        1                  00
…                              …        …                   …
980546                  2019-12-31       12                  52
980547                  2019-12-31       12                  52
980548                  2019-12-31       12                  52
980549                  2019-12-31       12                  52
980550                  2019-12-31       12                  52

[980551 rows x 43 columns]
```

[67]:
```
## Number of passengers boarding at all stop ids per day
cols = ['TRIP_START_TIME_WW', 'STOP_ID']
df_pass_count = data2019.groupby(cols)[["PASSENGERS_ON"]].sum()
df_pass_count
```

[67]:
```
                                PASSENGERS_ON
TRIP_START_TIME_WW STOP_ID
00                 100                    4
                   611                   67
                   612                   52
                   619                    4
```

```
                    621                 3
...                                     ...
52                  17661               538
                    17676               14
                    17677               24
                    17823               7
                    17824               0

[8523 rows x 1 columns]
```

[68]:
```
df_pass_count.plot.line()
plt.show()
```

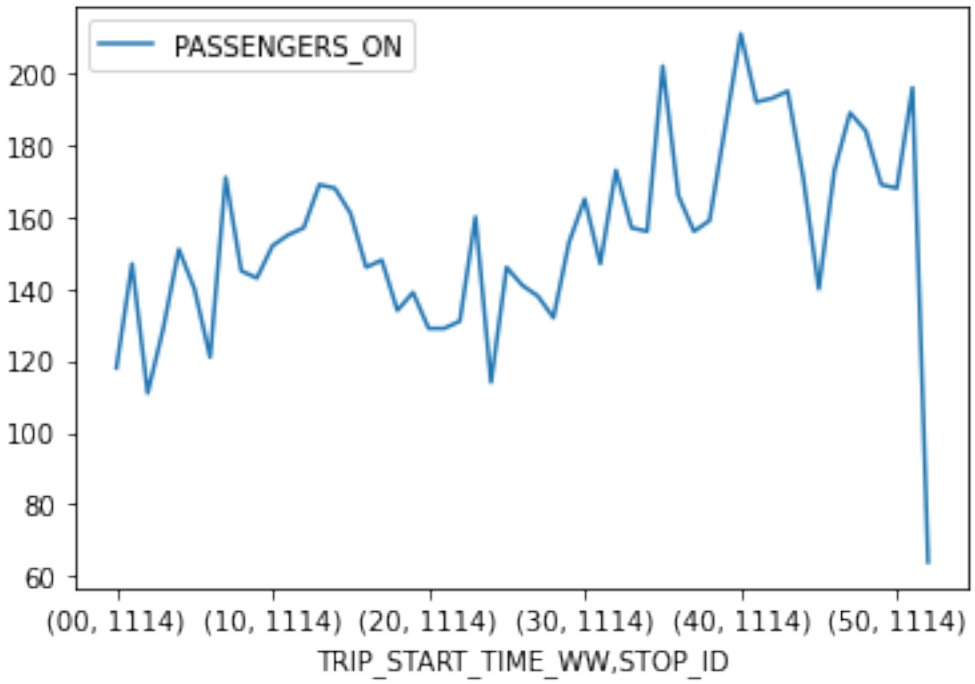


[69]:
```
df_weekly_1114 = data2019.loc[(data2019['STOP_ID'] == 1114) &␣
↪(data2019['PASSENGERS_ON'] >= 0)]
df_weekly_1114_1 = df_weekly_1114.groupby(cols)[["PASSENGERS_ON"]].sum()
df_weekly_1114_1
```

[69]:
```
                                PASSENGERS_ON
TRIP_START_TIME_WW STOP_ID
00                 1114                 118
01                 1114                 147
02                 1114                 111
03                 1114                 129
```

```
04                  1114                151
…                                       …
48                  1114                184
49                  1114                169
50                  1114                168
51                  1114                196
52                  1114                 64

[53 rows x 1 columns]
```

[78]:
```
df_weekly_1114_1.plot.line()
plt.show()
```



[ ]:

[ ]: