In [1]:
```
from matplotlib import pyplot as plt
from matplotlib.dates import MonthLocator, num2date
from matplotlib.ticker import FuncFormatter
from prophet import Prophet
from prophet.diagnostics import cross_validation, performance_metrics
from prophet.plot import add_changepoints_to_plot

import pandas as pd
import numpy as np
import datetime as dt
from collections import defaultdict
import time
import datetime as dt
from pytz import timezone
tz = timezone('EST')
from tqdm import tqdm

from sklearn.metrics import mean_absolute_error, mean_absolute_percentage_error,

import seaborn as sns
%config InlineBackend.figure_format = 'retina'
%matplotlib inline
from matplotlib import pyplot as plt
from matplotlib import style
sns.set()
```

In [2]:
```
vt2011 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2011 = pd.read_excel(vt2011, 'VT')
vt2012 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2012 = pd.read_excel(vt2012, 'VT')
vt2013 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2013 = pd.read_excel(vt2013, 'VT')
vt2014 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2014 = pd.read_excel(vt2014, 'VT')
vt2015 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2015 = pd.read_excel(vt2015, 'VT')
vt2016 = pd.ExcelFile(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_2011
vt2016 = pd.read_excel(vt2016, 'VT')
```

In [3]:
```
vt2017 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
vt2018 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
vt2019 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
vt2020 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
vt2021 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
vt2022 = pd.read_excel(r"C:\Users\Rohan\Desktop\Big Data\Load Data\smd_hourly_201
```

In [4]:
```
vt2022
```

Out[4]:

	Date	Hr_End	DA_Demand	RT_Demand	DA_LMP	DA_EC	DA_CC	DA_MLC	RT_LMP	RT_
0	2022-01-01	1	572.4	531.140	32.20	32.35	-0.55	0.40	25.56	25.
1	2022-01-01	2	555.1	508.084	31.93	32.31	-0.68	0.30	25.54	25.
2	2022-01-01	3	496.6	490.019	31.01	30.85	0.00	0.16	27.41	27.
3	2022-01-01	4	480.6	483.029	29.82	29.69	0.00	0.13	25.07	25.
4	2022-01-01	5	481.5	485.721	31.04	30.86	0.00	0.18	29.20	29.
...	...	...	...	...	...	...	...	...	...	
739	2022-01-31	20	778.7	822.082	227.58	226.40	0.00	1.18	296.06	295.
740	2022-01-31	21	705.3	793.781	204.16	202.82	0.00	1.34	265.22	264.
741	2022-01-31	22	629.3	764.808	182.45	183.42	0.00	-0.97	252.46	251.
742	2022-01-31	23	628.3	720.894	179.47	179.93	0.00	-0.46	191.26	190.
743	2022-01-31	24	540.5	681.452	188.36	190.54	0.00	-2.18	189.68	189.

744 rows × 14 columns

In [5]:
```
val2011 = vt2011['DEMAND']
val2012 = vt2012['DEMAND']
val2013 = vt2013['DEMAND']
val2014 = vt2014['DEMAND']
val2015 = vt2015['DEMAND']
val2016 = vt2016['RT_Demand']
val2017 = vt2017['RT_Demand']
val2018 = vt2018['RT_Demand']
val2019 = vt2019['RT_Demand']
val2020 = vt2020['RT_Demand']
val2021 = vt2021['RT_Demand']
val2022 = vt2022['RT_Demand']
```

In [6]:
```
values = [val2011, val2012, val2013, val2014, val2015, val2016, val2017, val2018,
values_df = pd.concat(values, axis=0, ignore_index=False)
values_df = values_df.reset_index()
period = len(values_df)
```

In [7]:
```
rng = pd.date_range('2011-01-01', periods=period, freq='1H')
date_df = pd.DataFrame({ 'ds': rng})
date_df = date_df.reset_index()
```

In [8]:
```
frames = [date_df, values_df]
vt_load = pd.concat(frames, axis=1, ignore_index=False)
vt_load = vt_load.rename(columns={vt_load.columns[1]: 'ds', vt_load.columns[3]: '
frames2 = [vt_load['ds'], vt_load['y']]
vt_load = pd.concat(frames2, axis=1, ignore_index=False)
vt_load
```

Out[8]:

	ds	y
0	2011-01-01 00:00:00	575.000
1	2011-01-01 01:00:00	540.000
2	2011-01-01 02:00:00	516.000
3	2011-01-01 03:00:00	505.000
4	2011-01-01 04:00:00	503.000
...	...	...
97171	2022-01-31 19:00:00	822.082
97172	2022-01-31 20:00:00	793.781
97173	2022-01-31 21:00:00	764.808
97174	2022-01-31 22:00:00	720.894
97175	2022-01-31 23:00:00	681.452

97176 rows × 2 columns

In [9]:
```
model = Prophet(
        changepoint_prior_scale=0.5,
        seasonality_mode='multiplicative',
        interval_width=0.95,
    )
model.add_country_holidays(country_name='US')
```

Out[9]: <prophet.forecaster.Prophet at 0x2c302a09f70>

In [10]:
```
model.fit(vt_load)
```

Out[10]: <prophet.forecaster.Prophet at 0x2c302a09f70>

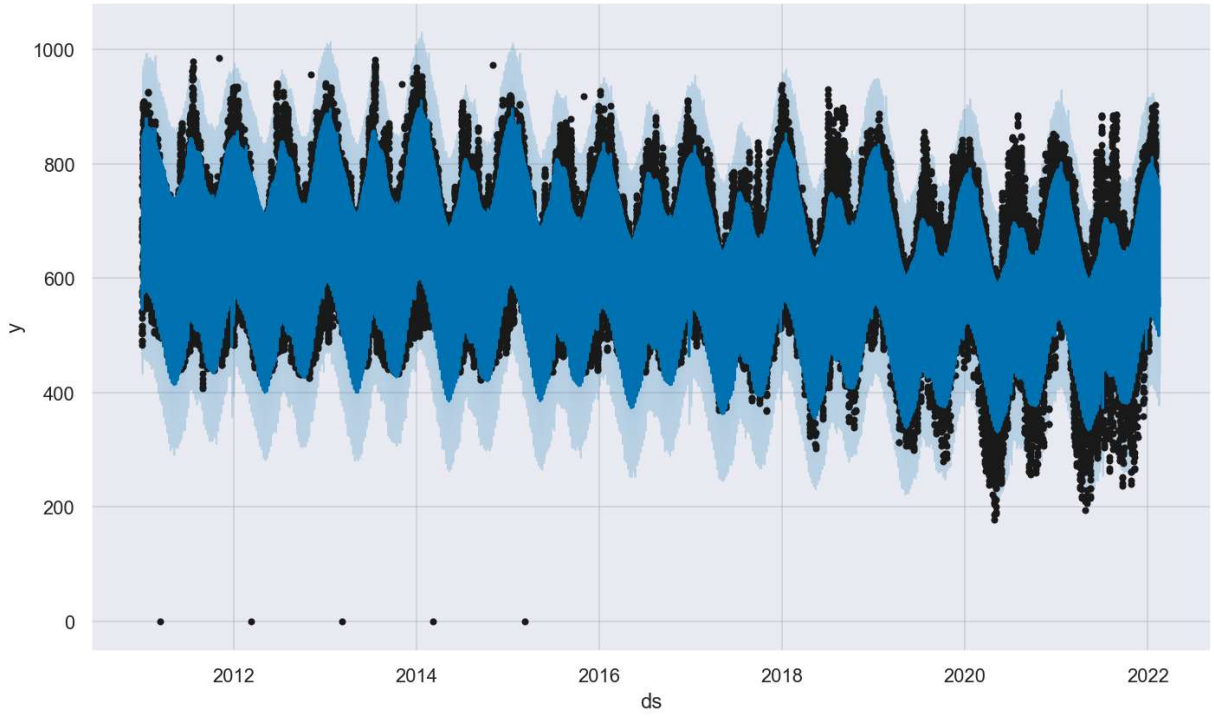
In [11]:
```
future_pd = model.make_future_dataframe(
        periods=365,
        freq='1H',
        include_history=True
    )

# make predictions
forecast_pd = model.predict(future_pd)
```

In [12]:
```
forecast_pd[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

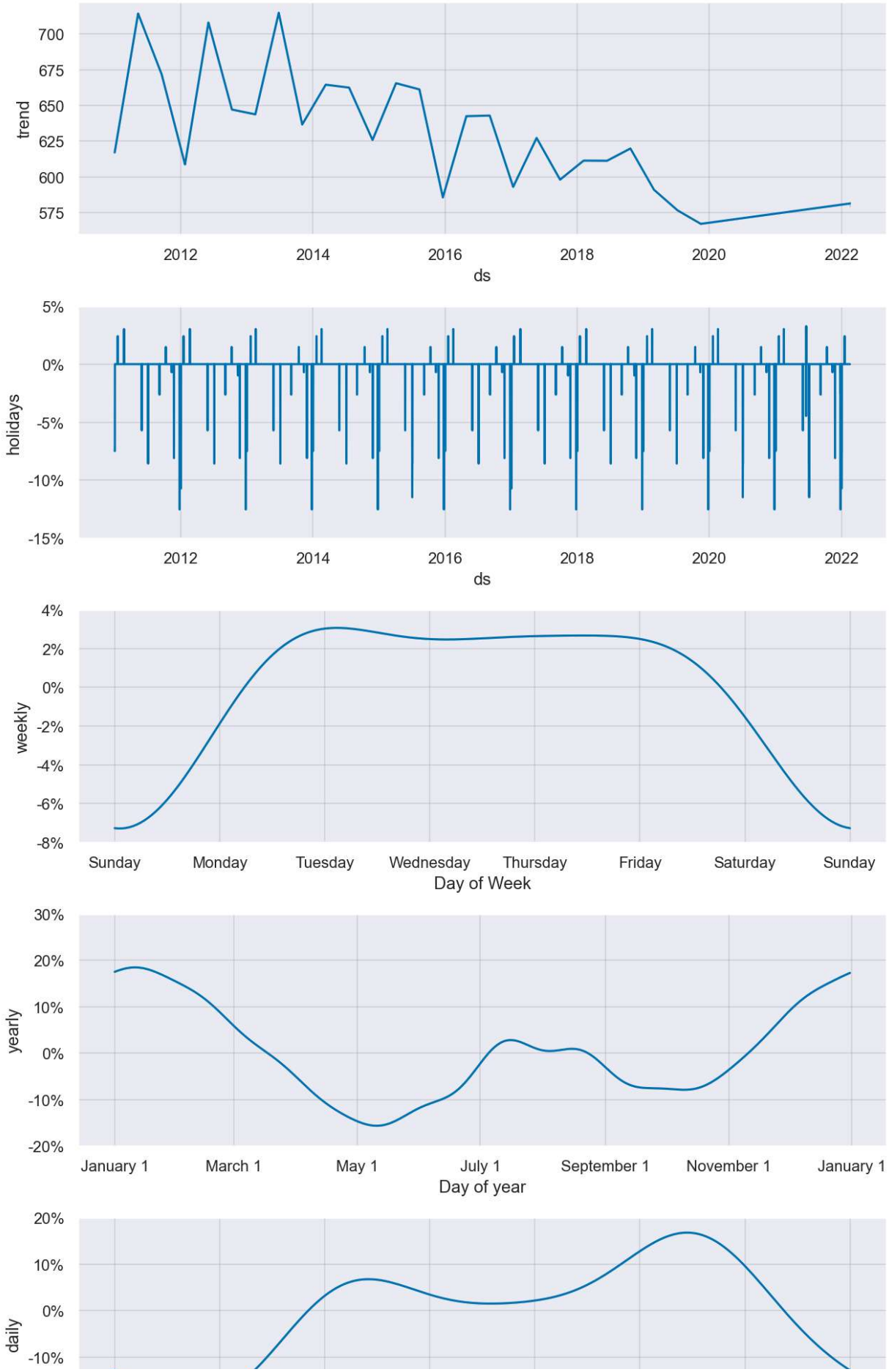
Out[12]:

	ds	yhat	yhat_lower	yhat_upper
97536	2022-02-16 00:00:00	586.604746	468.085362	699.835360
97537	2022-02-16 01:00:00	565.027689	451.839024	681.788991
97538	2022-02-16 02:00:00	551.093867	437.913377	673.666607
97539	2022-02-16 03:00:00	549.815510	431.901808	667.849935
97540	2022-02-16 04:00:00	567.513779	453.867082	687.185613

In [13]:
```
fig1 = model.plot(forecast_pd)
```



In [14]:
```
fig2 = model.plot_components(forecast_pd)
```



In [15]:
```
forecast_pd
```

Out[15]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	Christmas Day	Christm Day_lov
0	2011-01-01 00:00:00	617.098584	470.958485	702.035181	617.098584	617.098584	0.0	
1	2011-01-01 01:00:00	617.129849	450.164900	686.024280	617.129849	617.129849	0.0	
2	2011-01-01 02:00:00	617.161113	435.607291	664.157556	617.161113	617.161113	0.0	
3	2011-01-01 03:00:00	617.192377	433.516658	674.439339	617.192377	617.192377	0.0	
4	2011-01-01 04:00:00	617.223641	446.755465	676.421690	617.223641	617.223641	0.0	
...	...	...	...	...	...	...	...	
97536	2022-02-16 00:00:00	581.407820	468.085362	699.835360	579.415292	582.307830	0.0	
97537	2022-02-16 01:00:00	581.408541	451.839024	681.788991	579.407800	582.325518	0.0	
97538	2022-02-16 02:00:00	581.409262	437.913377	673.666607	579.400308	582.343207	0.0	
97539	2022-02-16 03:00:00	581.409982	431.901808	667.849935	579.392816	582.360895	0.0	
97540	2022-02-16 04:00:00	581.410703	453.867082	687.185613	579.385325	582.378583	0.0	

97541 rows × 73 columns

In [ ]:
```
```