In [3]:

```
import pandas as pd
from matplotlib import pyplot
from pandas import read_csv
from pandas import to_datetime
from pandas import DataFrame
from pandas import date_range
from pandas.tseries.offsets import MonthEnd
from fbprophet import Prophet
```

In [4]:

```
df = read_csv('data.csv', header=0)
# summarize shape
print(df.shape)
# show first few rows
print(df.head())
```

```
(654, 2)
       Date  Actual
0  1/2/2020       2
1  1/3/2020       5
2  1/6/2020       6
3  1/7/2020       7
4  1/8/2020       8
```

In [22]:

```
# plot the time series
#df.plot()
#pyplot.show()
```

In [5]:

```
df.columns = ['ds', 'y']
df['ds']= to_datetime(df['ds'])
print(df.tail())
```

```
            ds   y
649 2022-03-25   5
650 2022-03-28   4
651 2022-03-29  11
652 2022-03-30   6
653 2022-03-31   1
```

In [74]:

```
df['cap'] = 18
df['floor'] = 1
m = Prophet(growth = 'logistic',
            changepoint_prior_scale=0.100,
            daily_seasonality=False,
            weekly_seasonality=False,
            yearly_seasonality=20,
           seasonality_prior_scale=15,
           seasonality_mode = 'multiplicative',)
m.add_seasonality(name='daily', period=1, prior_scale=0.07, fourier_order=3)
m.add_seasonality(name='weekly', period=7, prior_scale=0.06, fourier_order=7)
m.add_seasonality(name='monthly', period=30.5, prior_scale=0.05, fourier_order=12)
m.fit(df)
```

Out[74]:

```
<fbprophet.forecaster.Prophet at 0x1fb98a0b670>
```

In [78]:

```
## Hyperparameter tuning
```

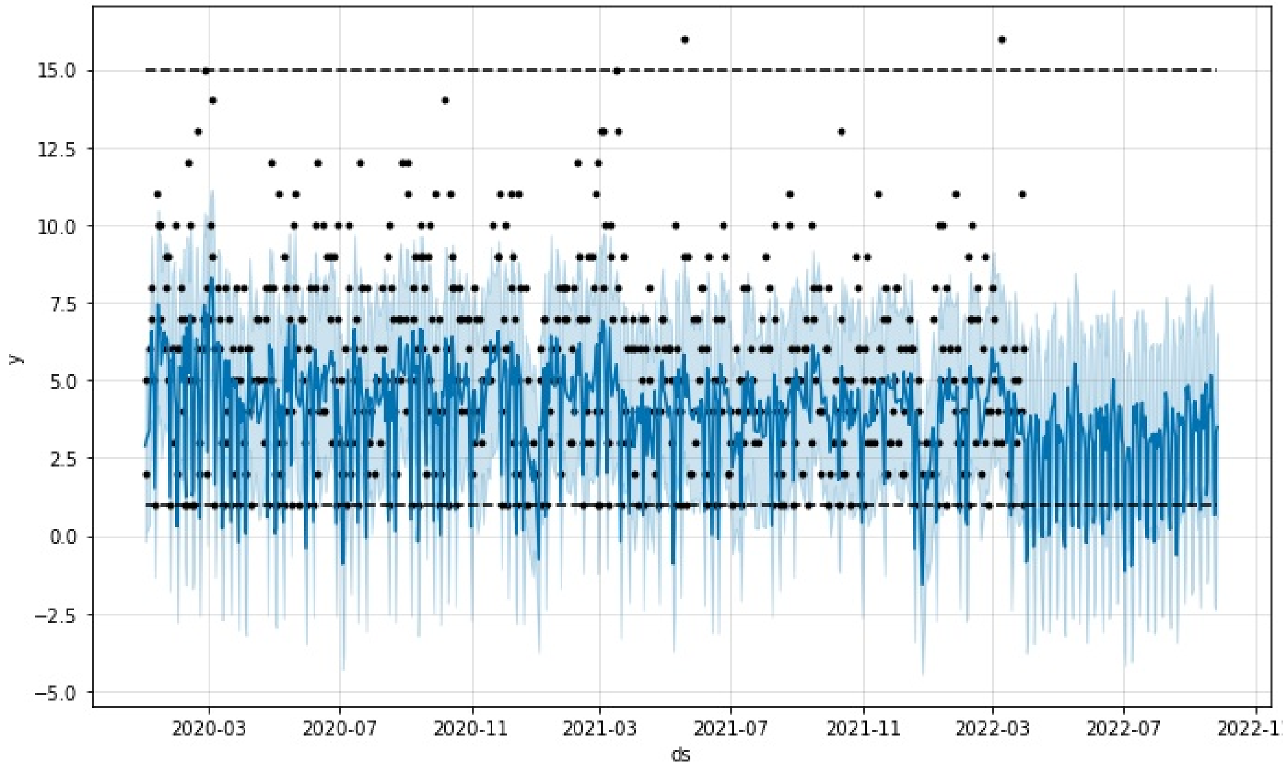
```
#import itertools
#import numpy as np

#param_grid = {
#    'changepoint_prior_scale': [0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5],
#    'seasonality_prior_scale': [0.01, 0.1, 1.0, 5.0, 10.0, 15, 20],
#}

# Generate all combinations of parameters
#all_params = [dict(zip(param_grid.keys(), v)) for v in itertools.product(*param_grid.val
ues())]
#maes = []  # Store the RMSEs for each params here

# Use cross validation to evaluate all parameters
#for params in all_params:
#    m2 = Prophet(**params).fit(df)  # Fit model with given params
#    df_cv = cross_validation(m2, initial='365.25 days', period='30 days', horizon = '30
0 days', parallel="processes")
#    df_p = performance_metrics(df_cv, rolling_window=1)
#    maes.append(df_p['mae'].values[0])

# Find the best parameters
#tuning_results = pd.DataFrame(all_params)
#tuning_results['mae'] = maes
#print(tuning_results)
```

In [ ]:

```

```

In [75]:

```
future = m.make_future_dataframe(periods=180)
future['cap'] = 15
future['floor'] = 1


forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']]
fig1 = m.plot(forecast)
```



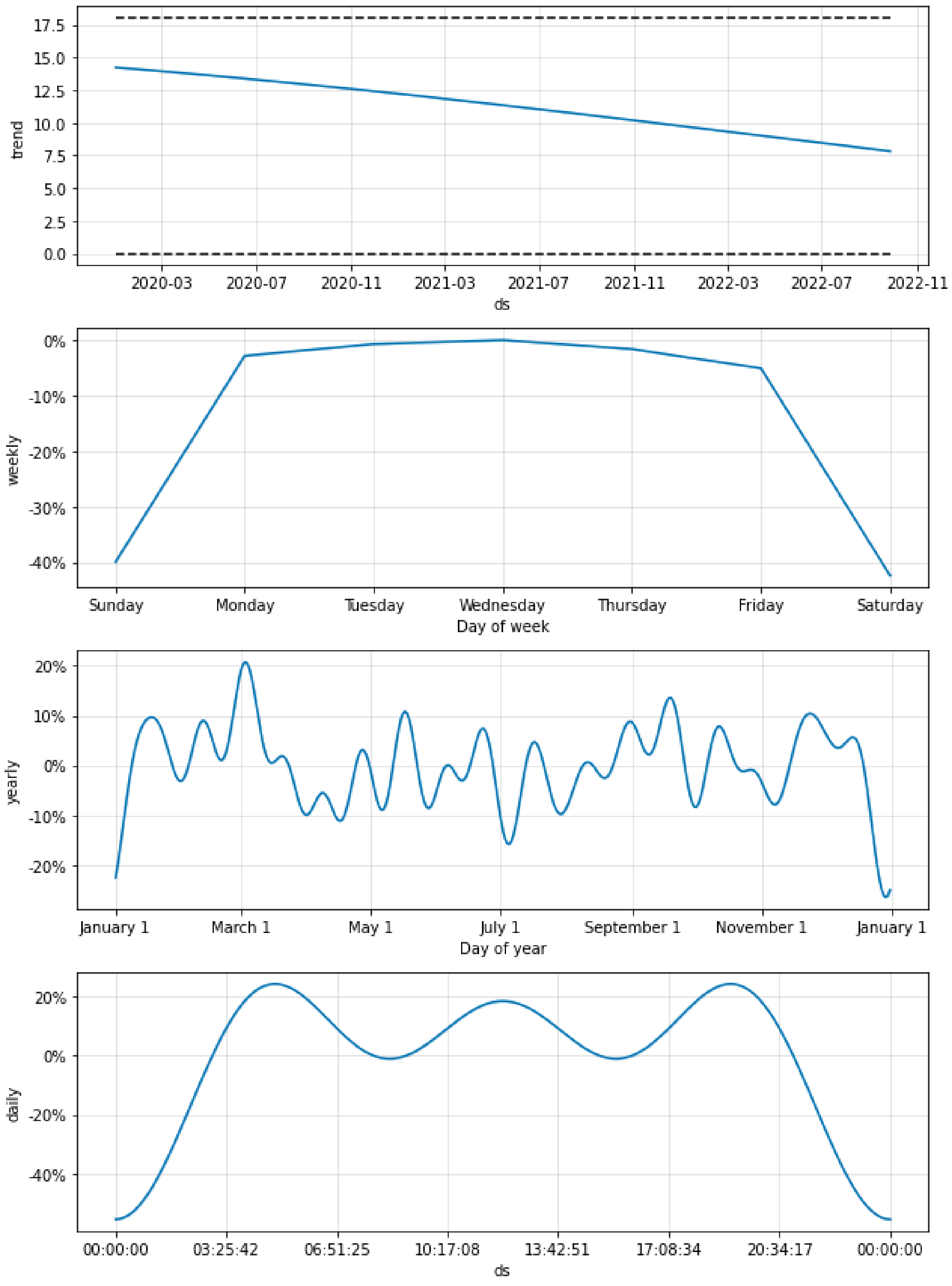
In [ ]:

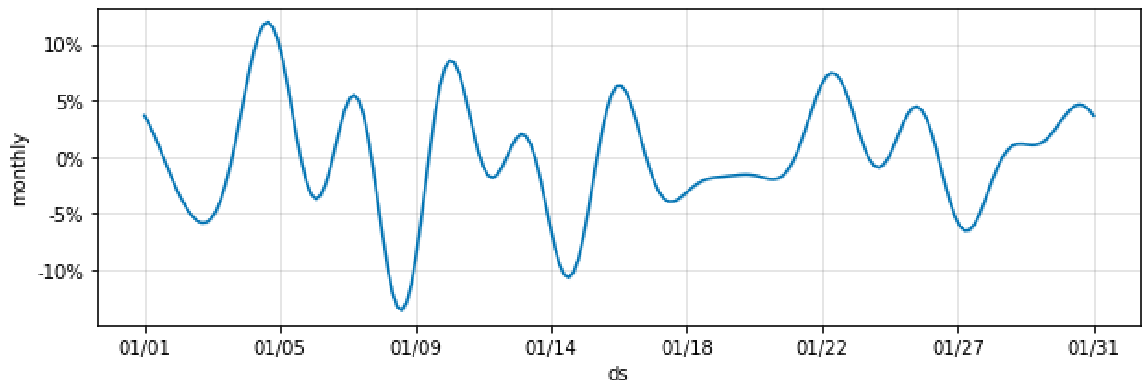
```

```

In [69]:

```
fig2 = m.plot_components(forecast)
```

```
C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator
```





In [70]:

```
from fbprophet.plot import plot_plotly, plot_components_plotly
plot_plotly(m, forecast)
```

In [76]:

```
from fbprophet.diagnostics import cross_validation
df_cv = cross_validation(m, initial='365.25 days', period='30 days', horizon = '300 days
')
```

```
INFO:fbprophet:Making 6 forecasts with cutoffs between 2021-01-05 00:00:00 and 2021-06-04
00:00:00
```

In [77]:

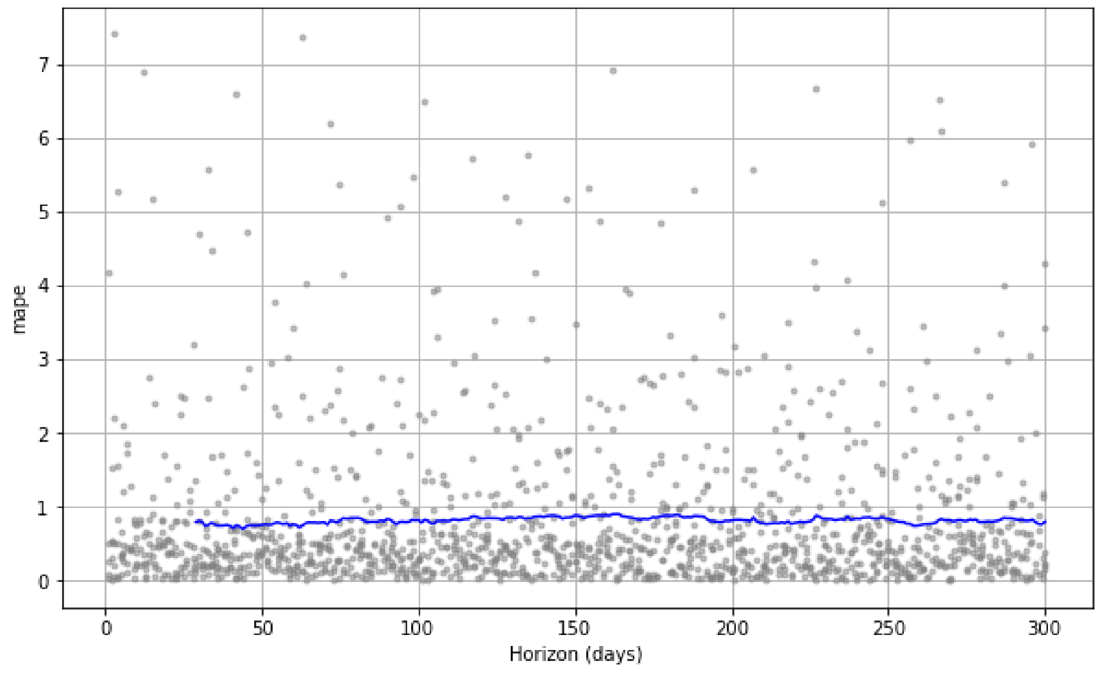
```
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
df_p = performance_metrics(df_cv)
```

```
fig = plot_cross_validation_metric(df_cv, metric='mape')
df_p.head()
```

```
C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:526: FutureWarning:

casting timedelta64[ns] values to int64 with .astype(...) is deprecated and will raise in
a future version. Use .view(...) instead.

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:527: FutureWarning:

casting timedelta64[ns] values to int64 with .astype(...) is deprecated and will raise in
a future version. Use .view(...) instead.
```

Out[77]:

	horizon	mse	rmse	mae	mape	mdape	coverage
0	29 days	10.172229	3.189393	2.517488	0.786439	0.449741	0.618841
1	30 days	10.270454	3.204755	2.547901	0.801047	0.451983	0.615942
2	31 days	10.128124	3.182471	2.522261	0.787870	0.440563	0.623188
3	32 days	9.703671	3.115072	2.475048	0.733260	0.440563	0.630435
4	33 days	9.627698	3.102853	2.474111	0.746377	0.436235	0.635266



In [ ]:

In [ ]:

In [ ]:

In [73]:

```
##To save predicted volumes to another csv file for validation
import os
df2 = pd.read_csv('data.csv', header=0)

df3 = pd.DataFrame()
df3['ds'] = forecast['ds'].copy()
df3['Actual'] = df2['Actual'].copy()
```

```
df3['Predicted'] = forecast['yhat'].copy()
df3['Predicted'] = df3['Predicted'].astype(float).round(0)
df3.head()


df3.set_index('ds', inplace=True)
df3.to_csv('data-val.csv')
```

In [ ]:

```

```

In [ ]:

```

```

In [7]:

```
from sklearn.metrics import mean_squared_error, r2_score,mean_absolute_error, mean_absolu
te_percentage_error
df_val = read_csv('UVW-val.csv', header=0)
#df_val
df_val["Actual"].astype(float)
df_val["Predicted"].astype(float)
#df_val.dtypes
r2_score(df_val['Actual'],df_val['Predicted'])
```

Out[7]:

```
0.48828436537313025
```

In [8]:

```
mean_squared_error(df_val['Actual'],df_val['Predicted'])
```

Out[8]:

```
25.988505747126435
```

In [9]:

```
mean_absolute_error(df_val['Actual'],df_val['Predicted'])
```

Out[9]:

```
3.9655172413793105
```

In [10]:

```
mean_absolute_percentage_error(df_val['Actual'],df_val['Predicted'])
```

Out[10]:

```
996486124446920.5
```

In [27]:

```
##To Save the Model
import json
from fbprophet.serialize import model_to_json, model_from_json
with open('serialized_model.json', 'w') as fout:
    json.dump(model_to_json(m), fout)
```

In [23]:

```
import pandas as pd
cutoffs = pd.date_range(start='2020-12-01', end='2021-03-01', freq='M')
print(cutoffs)
```

```
DatetimeIndex(['2020-12-31', '2021-01-31', '2021-02-28'], dtype='datetime64[ns]', freq='M
')
```