Stock Price prediction using Prophet

```
#Importing all the necessary libraries

import pandas as pd
import plotly.express as px
from prophet import Prophet
```

```
#Initializing Plotly
import plotly.io as pio
pio.renderers.default='colab'
```

```
FVRR_df=pd.read_csv("FVRR.csv")
QFIN_df=pd.read_csv("QFIN.csv")
ROKU_df=pd.read_csv("ROKU.csv")
SAVA_df=pd.read_csv("SAVA.csv")

TWLO_df=pd.read_csv("TWLO.csv")
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
#read_csv function from pandas
FVRR_df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-06-13	26.000000	41.680000	25.549999	39.900002	39.900002	22046000
1	2019-06-14	41.639999	44.250000	31.490000	31.490000	31.490000	10275800
2	2019-06-17	32.810001	34.880001	31.541000	34.380001	34.380001	3789100
3	2019-06-18	35.259998	35.919998	31.040001	31.150000	31.150000	3274000
4	2019-06-19	31.799999	31.799999	28.250000	28.670000	28.670000	1902000
...	...	...	...	...	...	...	...
960	2023-04-05	34.389999	34.639999	33.365002	34.029999	34.029999	1394900
961	2023-04-06	33.849998	34.889999	32.880001	34.529999	34.529999	481100
962	2023-04-10	35.110001	35.369999	34.310001	35.049999	35.049999	434900
963	2023-04-11	35.119999	35.790001	34.869999	35.430000	35.430000	527100
964	2023-04-12	36.230000	36.750000	34.650002	34.830002	34.830002	590500

965 rows × 7 columns

```
QFIN_df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-06-14	14.500000	15.500000	14.020000	14.350000	13.335065	229700
1	2019-06-17	14.520000	14.790000	13.590000	13.670000	12.703159	108200

```
ROKU_df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-06-13	104.849998	105.400002	102.540001	104.970001	104.970001	7629200
1	2019-06-14	104.330002	106.120003	101.980003	102.019997	102.019997	7326900
2	2019-06-17	101.550003	104.330002	101.010002	103.839996	103.839996	5528300
3	2019-06-18	104.750000	104.879997	101.169998	104.389999	104.389999	6591700
4	2019-06-19	104.070000	106.550003	102.732002	106.489998	106.489998	5663000
...	...	...	...	...	...	...	...
960	2023-04-05	65.099998	65.389999	61.119999	61.270000	61.270000	9049600
961	2023-04-06	60.950001	64.169998	59.459999	64.080002	64.080002	9170100
962	2023-04-10	63.040001	64.449997	61.790001	64.320000	64.320000	7988400
963	2023-04-11	64.320000	64.980003	62.639999	63.970001	63.970001	5914800
964	2023-04-12	65.470001	65.470001	60.419998	60.470001	60.470001	9032800

965 rows × 7 columns

```
SAVA_df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-06-13	1.110000	1.12	1.100000	1.110000	1.110000	77100
1	2019-06-14	1.100000	1.22	1.100000	1.200000	1.200000	281600
2	2019-06-17	1.230000	1.23	1.150000	1.190000	1.190000	70600
3	2019-06-18	1.180000	1.22	1.110000	1.140000	1.140000	324900
4	2019-06-19	1.130000	1.16	1.130000	1.160000	1.160000	104500
...	...	...	...	...	...	...	...
960	2023-04-05	23.879999	24.08	23.309999	23.840000	23.840000	411200
961	2023-04-06	23.930000	24.18	23.410000	23.969999	23.969999	426100
962	2023-04-10	23.879999	23.93	22.420000	23.110001	23.110001	950300
963	2023-04-11	23.100000	23.82	23.049999	23.400000	23.400000	571000
964	2023-04-12	23.510000	23.84	22.850000	22.900000	22.900000	432500

965 rows × 7 columns

```
TWLO_df
```

```
            Date        Open        High        Low       Close  Adj Close   Volume
```

```
FVRR_df.info()



QFIN_df.info()

ROKU_df.info()

SAVA_df.info()
TWLO_df.info()
```

```
dtypes: float64(5), int64(1), object(1)
memory usage: 52.9+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 964 entries, 0 to 963
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       964 non-null    object 
 1   Open       964 non-null    float64
 2   High       964 non-null    float64
 3   Low        964 non-null    float64
 4   Close      964 non-null    float64
 5   Adj Close  964 non-null    float64
 6   Volume     964 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 52.8+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 965 entries, 0 to 964
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       965 non-null    object 
 1   Open       965 non-null    float64
 2   High       965 non-null    float64
 3   Low        965 non-null    float64
 4   Close      965 non-null    float64
 5   Adj Close  965 non-null    float64
 6   Volume     965 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 52.9+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 965 entries, 0 to 964
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       965 non-null    object 
 1   Open       965 non-null    float64
 2   High       965 non-null    float64
 3   Low        965 non-null    float64
 4   Close      965 non-null    float64
 5   Adj Close  965 non-null    float64
 6   Volume     965 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 52.9+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 965 entries, 0 to 964
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       965 non-null    object 
 1   Open       965 non-null    float64
 2   High       965 non-null    float64
 3   Low        965 non-null    float64
 4   Close      965 non-null    float64
 5   Adj Close  965 non-null    float64
 6   Volume     965 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 52.9+ KB
```

```
FVRR_df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	965.000000	965.000000	965.000000	965.000000	965.000000	9.650000e+02
mean	95.570161	98.453501	92.323667	95.448663	95.448663	8.063907e+05
std	78.452421	80.577269	75.831312	78.323589	78.323589	9.852608e+05
min	17.500000	18.170000	17.110001	17.680000	17.680000	3.170000e+04

```
QFIN_df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	964.000000	964.000000	964.000000	964.000000	964.000000	9.640000e+02
mean	16.266308	16.782992	15.718861	16.260830	15.347037	1.477851e+06
std	6.956765	7.235050	6.614239	6.953779	6.526019	1.620863e+06
min	6.500000	7.030000	6.370000	6.510000	6.049567	4.560000e+04
25%	10.760000	11.065000	10.527500	10.787500	10.045439	5.647250e+05
50%	14.330000	14.775000	13.915000	14.340000	13.765850	9.627000e+05
75%	20.925000	21.492500	20.205001	20.830000	19.601273	1.764275e+06
max	44.285000	45.000000	42.610001	44.049999	40.934464	2.344910e+07

```
ROKU_df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	965.000000	965.000000	965.000000	965.000000	965.000000	9.650000e+02
mean	180.436633	185.262047	175.461893	180.299166	180.299166	8.852012e+06
std	116.726871	119.095466	114.048885	116.507743	116.507743	7.345221e+06
min	39.169998	39.889999	38.259998	38.799999	38.799999	1.443700e+06
25%	92.730003	96.764999	89.720001	92.900002	92.900002	4.396200e+06
50%	134.509995	138.509995	131.119995	134.639999	134.639999	6.796000e+06
75%	276.029999	280.600006	271.339996	276.459991	276.459991	1.052060e+07
max	477.200012	490.760986	468.779999	479.500000	479.500000	6.658520e+07

```
SAVA_df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	965.000000	965.000000	965.000000	965.000000	965.000000	9.650000e+02
mean	27.206373	28.855047	25.849959	27.257389	27.257389	3.387860e+06
std	25.493375	27.271539	23.935321	25.481802	25.481802	9.279942e+06
min	1.050000	1.100000	1.030000	1.040000	1.040000	1.620000e+04
25%	4.850000	5.410000	4.630000	4.860000	4.860000	7.577000e+05
50%	24.500000	25.410000	23.879999	24.610001	24.610001	1.312700e+06
75%	42.070000	44.599998	40.799999	42.349998	42.349998	2.777500e+06
max	145.000000	146.160004	126.029999	135.300003	135.300003	1.750972e+08

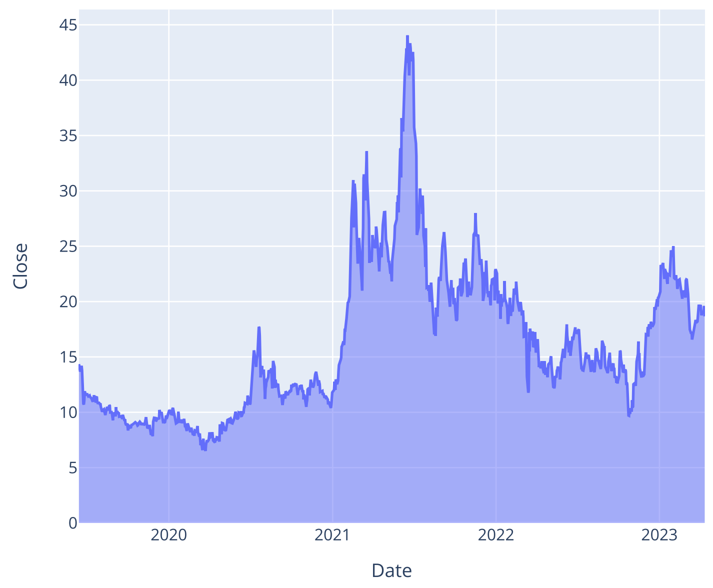
```
TWLO_df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	965.000000	965.000000	965.000000	965.000000	965.000000	9.650000e+02
mean	194.543392	198.973826	189.756916	194.367616	194.367616	3.345260e+06
min	43.290001	44.160000	41.000000	42.740002	42.740002	6.775000e+05
max	441.000000	457.299988	437.000000	443.489990	443.489990	4.484080e+07

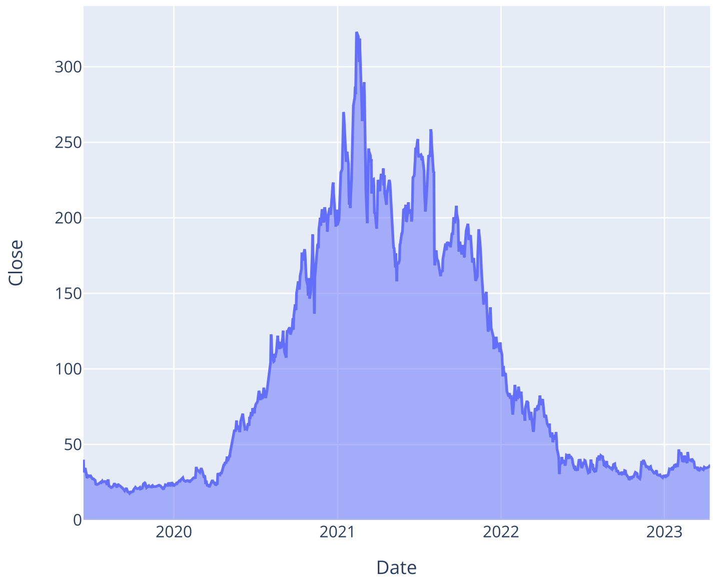
Data Visualization using plotly express- Visualizing the historical performance of the stocks

```
#Line graph, Area graph , box plot (Analyzing price and volume)
```

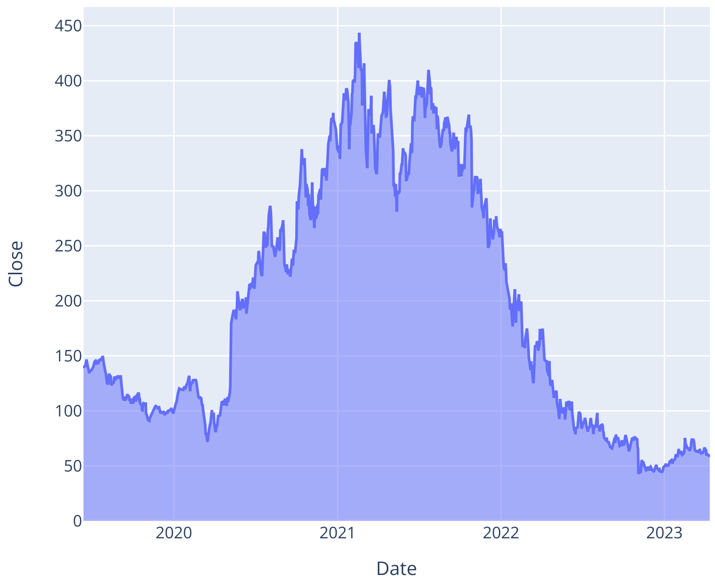
```
px.area(QFIN_df, x="Date", y="Close")
```



```
px.area(ROKU_df, x="Date", y="Close")
px.area(FVRR_df, x="Date", y="Close")
```

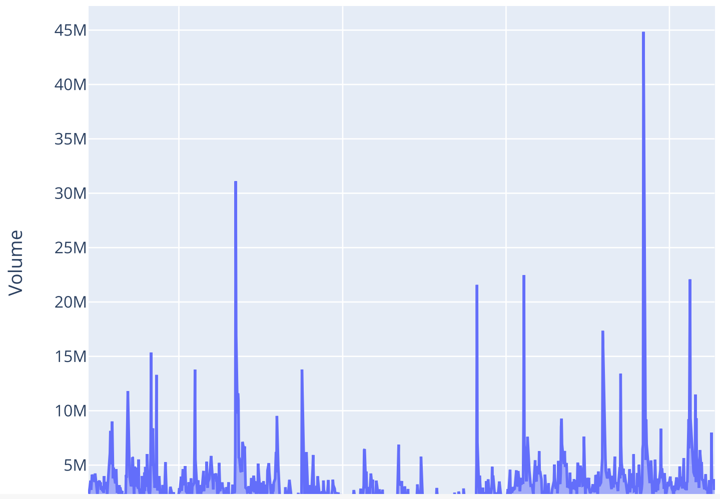


```
px.area(TWLO_df, x="Date", y="Close")
```

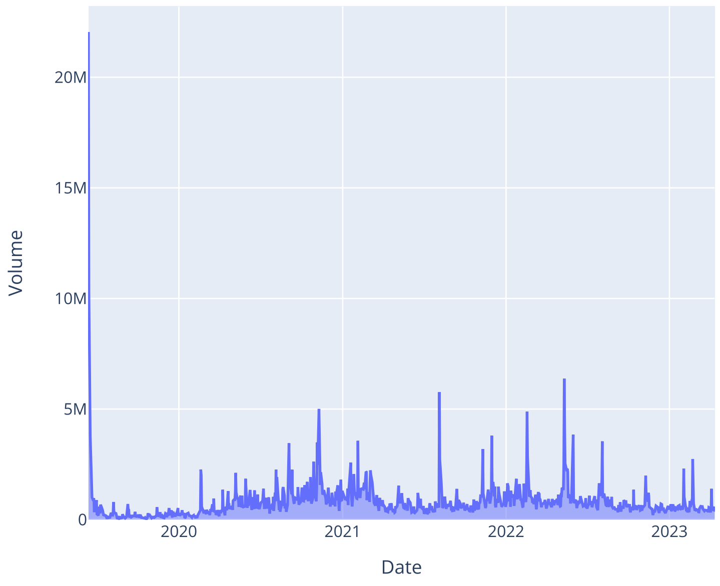


Double-click (or enter) to edit

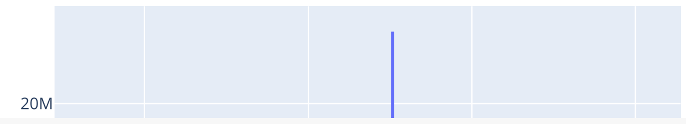
```
px.area(TWLO_df, x="Date", y="Volume")
```



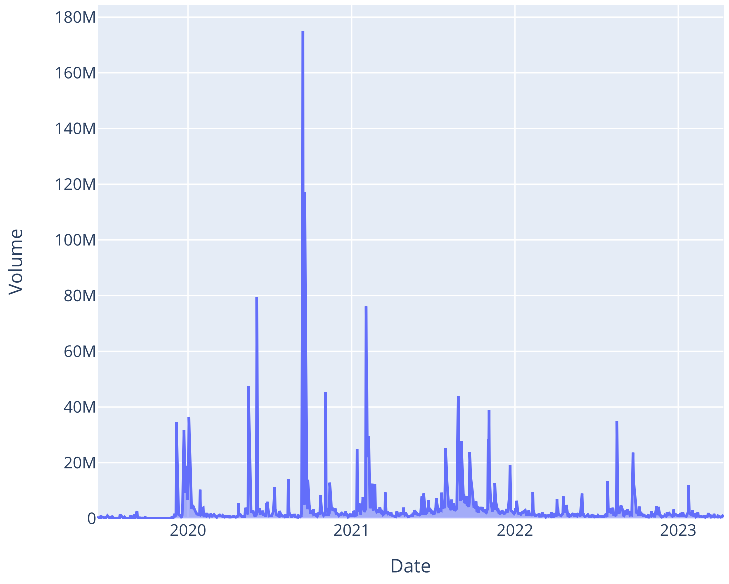
```
px.area(FVRR_df, x="Date", y="Volume")
```



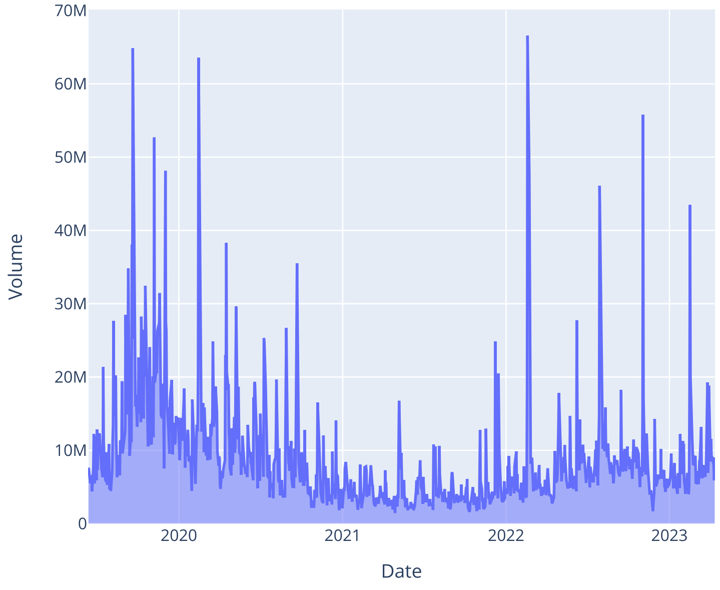
```
px.area(QFIN_df, x="Date", y="Volume")
```



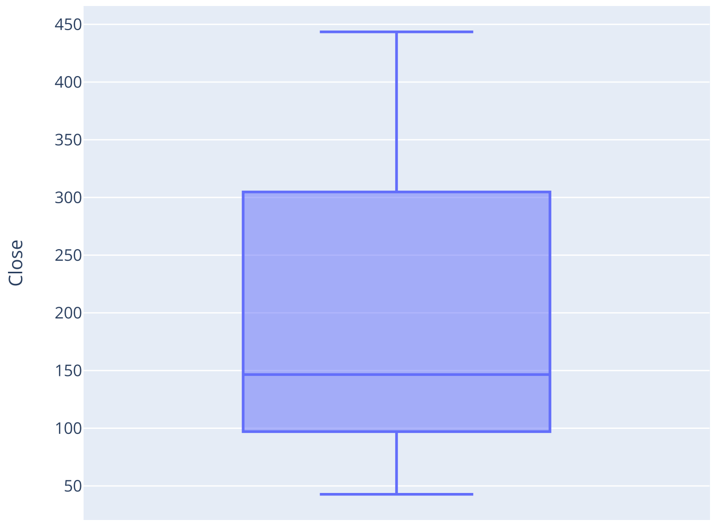
```
px.area(SAVA_df, x="Date", y="Volume")
```



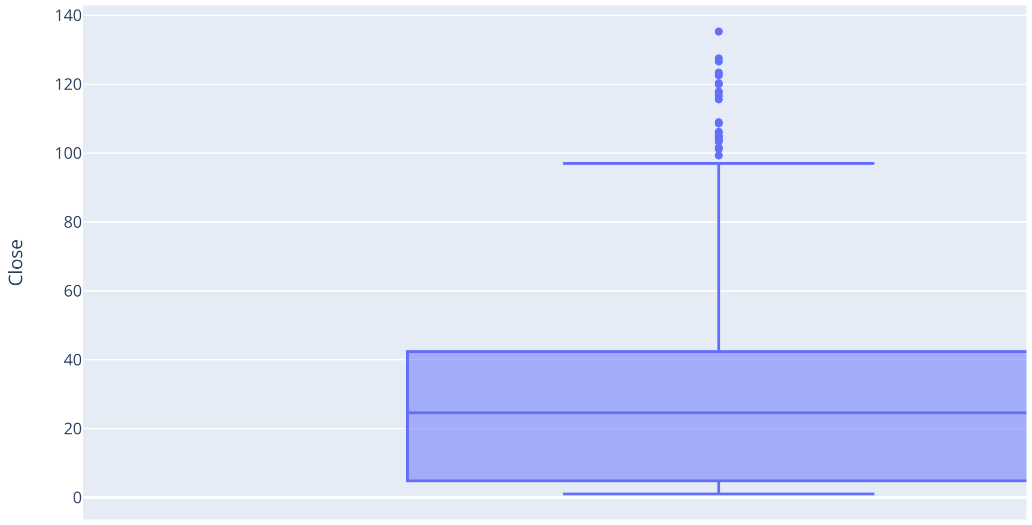
```
px.area(ROKU_df, x="Date", y="Volume")
```



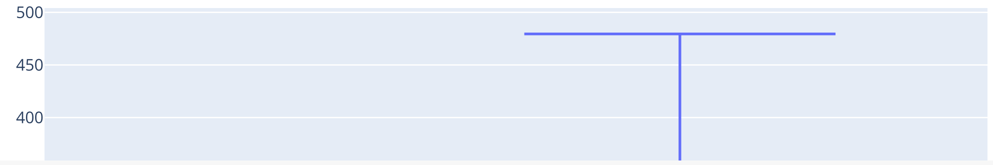
```
px.box(TWLO_df, y="Close")
```



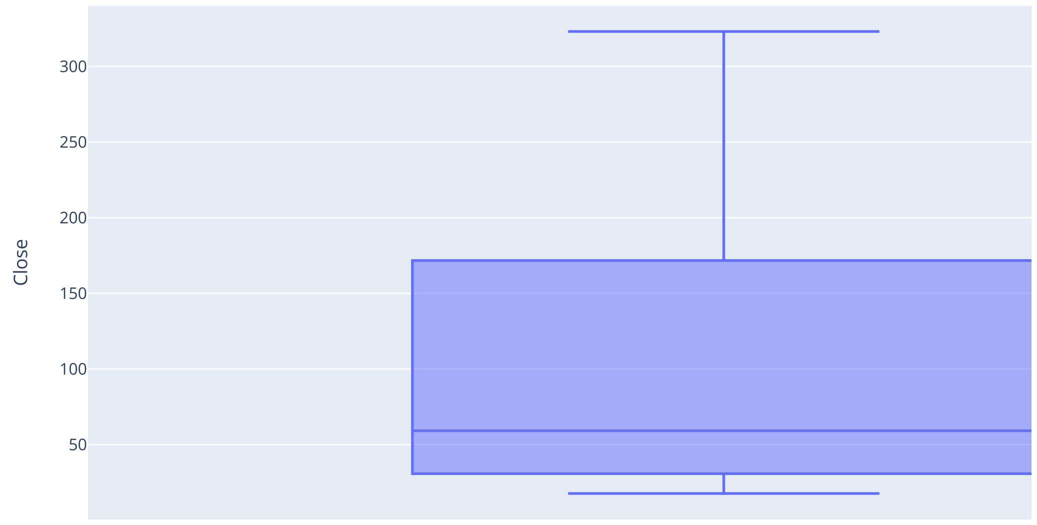
```
px.box(SAVA_df, y="Close")
```



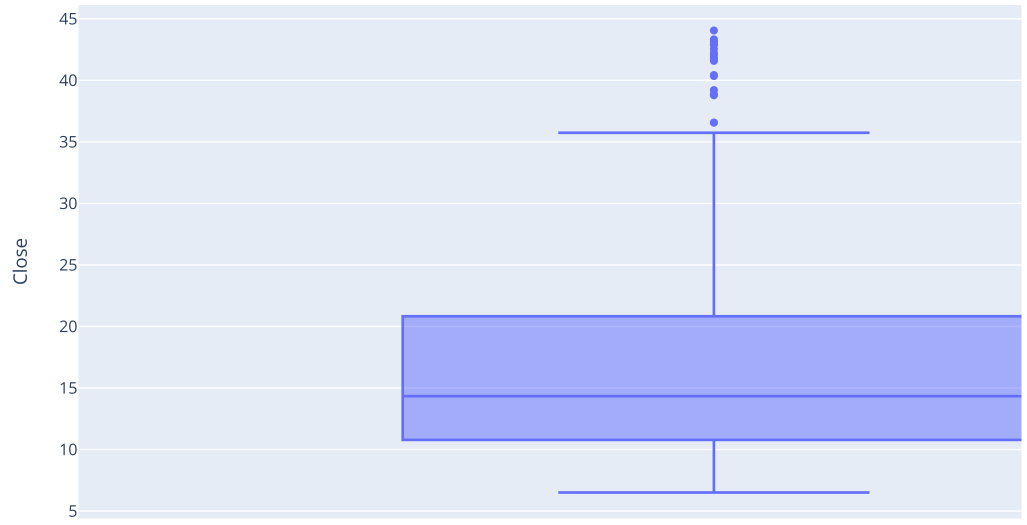
```
px.box(ROKU_df, y="Close")
```



```
px.box(FVRR_df, y="Close")
```



```
px.box(QFIN_df, y="Close")
```



```
## Data Preperation
```

```
columns=['Date', "Close"]
ndf = pd.DataFrame(TWLO_df, columns =columns)
```

```
ndf
```

	Date	Close
0	2019-06-13	141.059998
1	2019-06-14	140.169998
2	2019-06-17	140.710007
3	2019-06-18	142.289993
4	2019-06-19	146.500000
...	...	...
960	2023-04-05	59.320000
961	2023-04-06	60.759998
962	2023-04-10	59.639999
963	2023-04-11	59.040001
964	2023-04-12	58.130001

965 rows × 2 columns

```
prophet_df = ndf.rename (columns = {'Date':'ds','Close': 'y' })
```

```
prophet_df
```

	ds	y
0	2019-06-13	141.059998
1	2019-06-14	140.169998
2	2019-06-17	140.710007
3	2019-06-18	142.289993
4	2019-06-19	146.500000
...	...	...
960	2023-04-05	59.320000
961	2023-04-06	60.759998
962	2023-04-10	59.639999
963	2023-04-11	59.040001
964	2023-04-12	58.130001

965 rows × 2 columns

```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_df)
```

```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/gjapk1od.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/9svs_r1u.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=61893', '
17:15:51 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:15:51 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9ab3895970>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_TWLO= m.predict(future)
```

```
forecast_TWLO
```

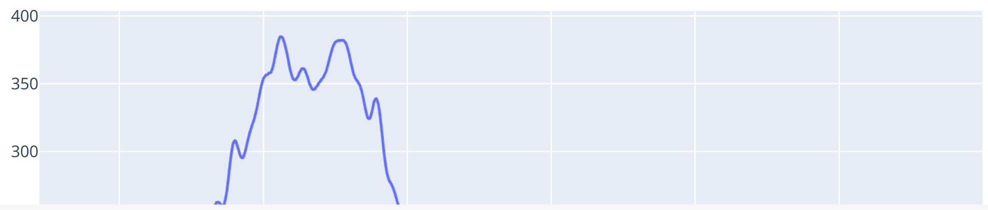
	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_te
0	2019-06-13	118.093845	96.476678	141.976335	118.093845	118.093845	1.423845	
1	2019-06-14	118.075798	100.318063	142.224496	118.075798	118.075798	2.582881	
2	2019-06-17	118.021659	102.307748	146.829930	118.021659	118.021659	6.382362	
3	2019-06-18	118.003612	103.340396	148.284609	118.003612	118.003612	7.895676	
4	2019-06-19	117.985565	106.403811	148.992951	117.985565	117.985565	8.977375	
...	...	...	...	...	...	...	...	
2785	2028-04-06	79.466711	-1502.214041	1735.614311	-1489.846815	1764.378874	-15.090059	
2786	2028-04-07	79.471694	-1505.397939	1747.772583	-1491.432313	1765.004340	-14.643960	
2787	2028-04-08	79.476676	-1488.747977	1763.607902	-1493.017811	1765.629807	0.979862	
2788	2028-04-09	79.481658	-1486.464217	1764.877773	-1494.603310	1766.255273	1.374563	
2789	2028-04-10	79.486641	-1512.587154	1741.265443	-1496.188808	1766.880739	-13.681702	

2790 rows × 19 columns

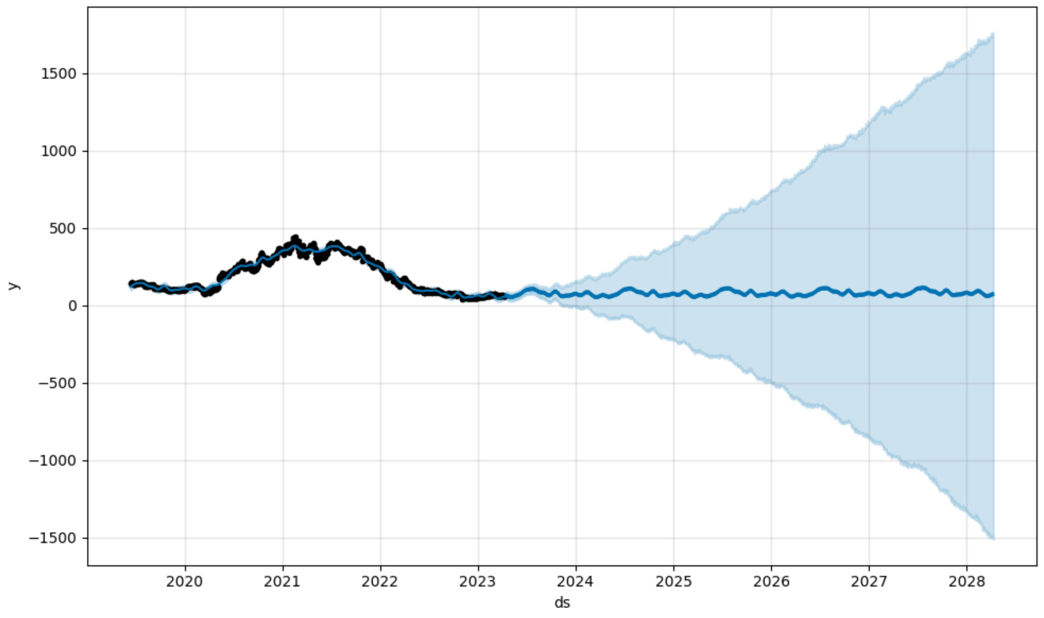
```
 # Display the underlying forecast dataframe (tail)
forecast_TWLO[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

	ds	yhat	yhat_lower	yhat_upper
2785	2028-04-06	64.376652	-1502.214041	1735.614311
2786	2028-04-07	64.827734	-1505.397939	1747.772583
2787	2028-04-08	80.456538	-1488.747977	1763.607902
2788	2028-04-09	80.856221	-1486.464217	1764.877773
2789	2028-04-10	65.804938	-1512.587154	1741.265443

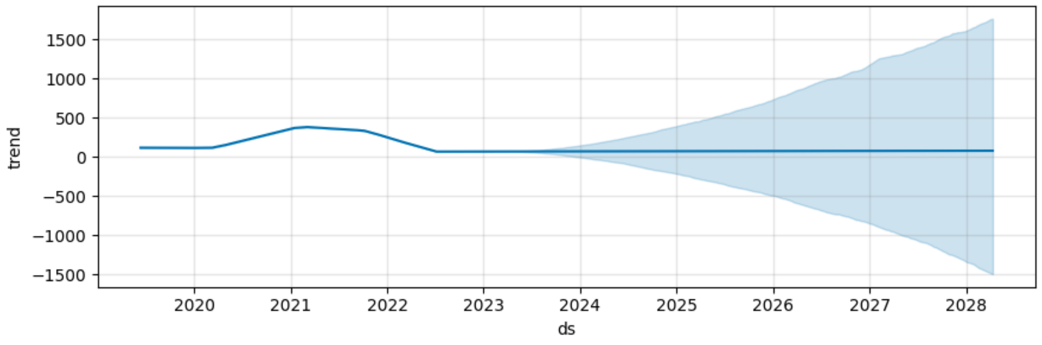
```
px.line(forecast_TWLO, x='ds',y='yhat')
```

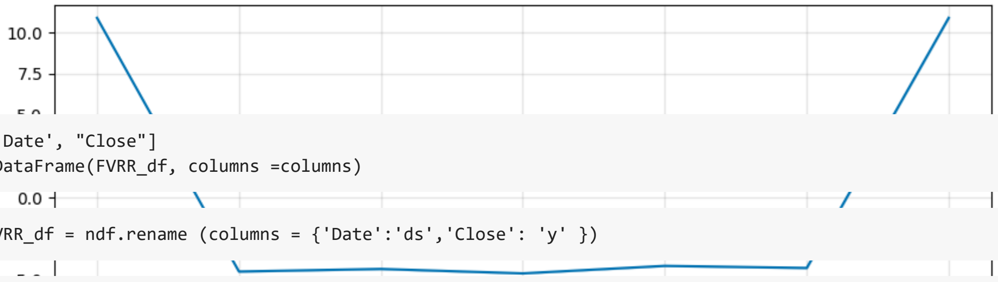


```
figure = m.plot(forecast_TWLO, xlabel = 'ds', ylabel = 'y')
```



```
figure2=m.plot_components(forecast_TWLO)
```





```
columns=['Date', "Close"]
ndf = pd.DataFrame(FVRR_df, columns =columns)
```

```
prophet_FVRR_df = ndf.rename (columns = {'Date':'ds','Close': 'y' })
```

```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_FVRR_df)
```

```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/fafddmtu.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/s0i1jfiv.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=96934', '
17:15:54 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:15:55 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9ab039aee0>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_FVRR= m.predict(future)
```

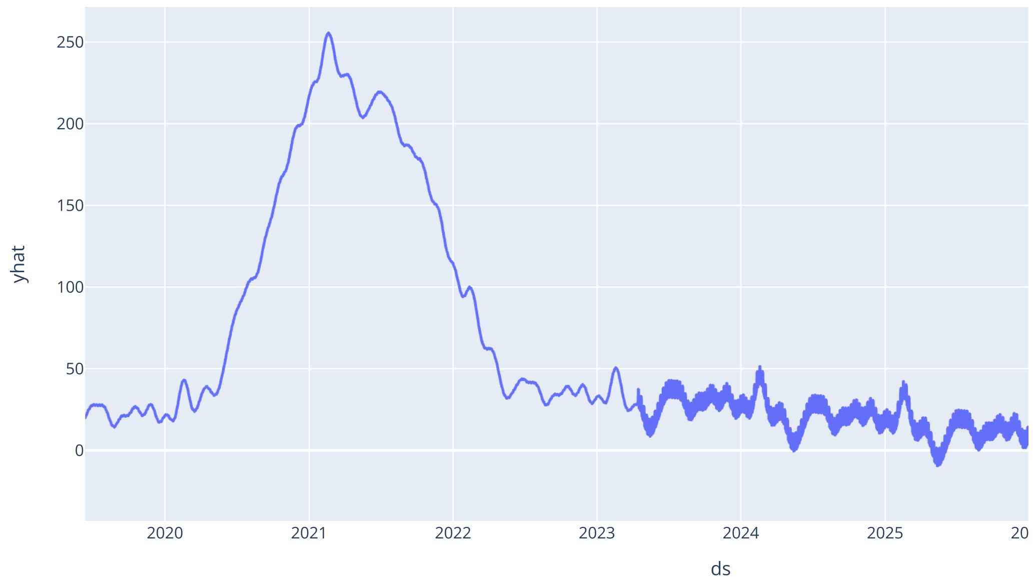
```
forecast_FVRR
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_te
0	2019-06-13	25.175468	3.201889	37.567580	25.175468	25.175468	-5.338177	
1	2019-06-14	25.165478	3.502947	37.274032	25.165478	25.165478	-4.372060	

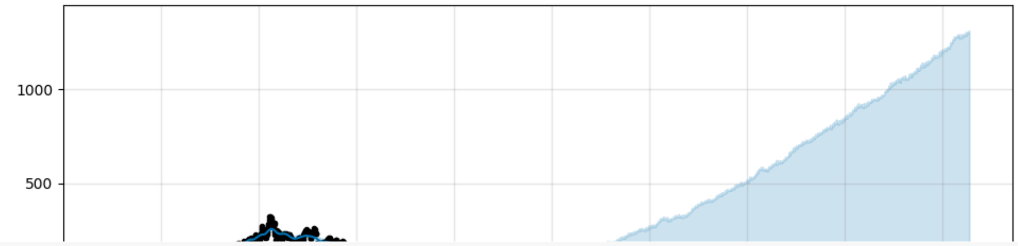
```
 # Display the underlying forecast dataframe (tail)
forecast_FVRR[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

	ds	yhat	yhat_lower	yhat_upper
2785	2028-04-06	-17.282245	-1393.897618	1305.706130
2786	2028-04-07	-16.884033	-1386.912969	1301.914866
2787	2028-04-08	-7.049663	-1377.089628	1310.823308
2788	2028-04-09	-7.044627	-1384.909096	1314.678813
2789	2028-04-10	-17.672504	-1400.725390	1309.141639

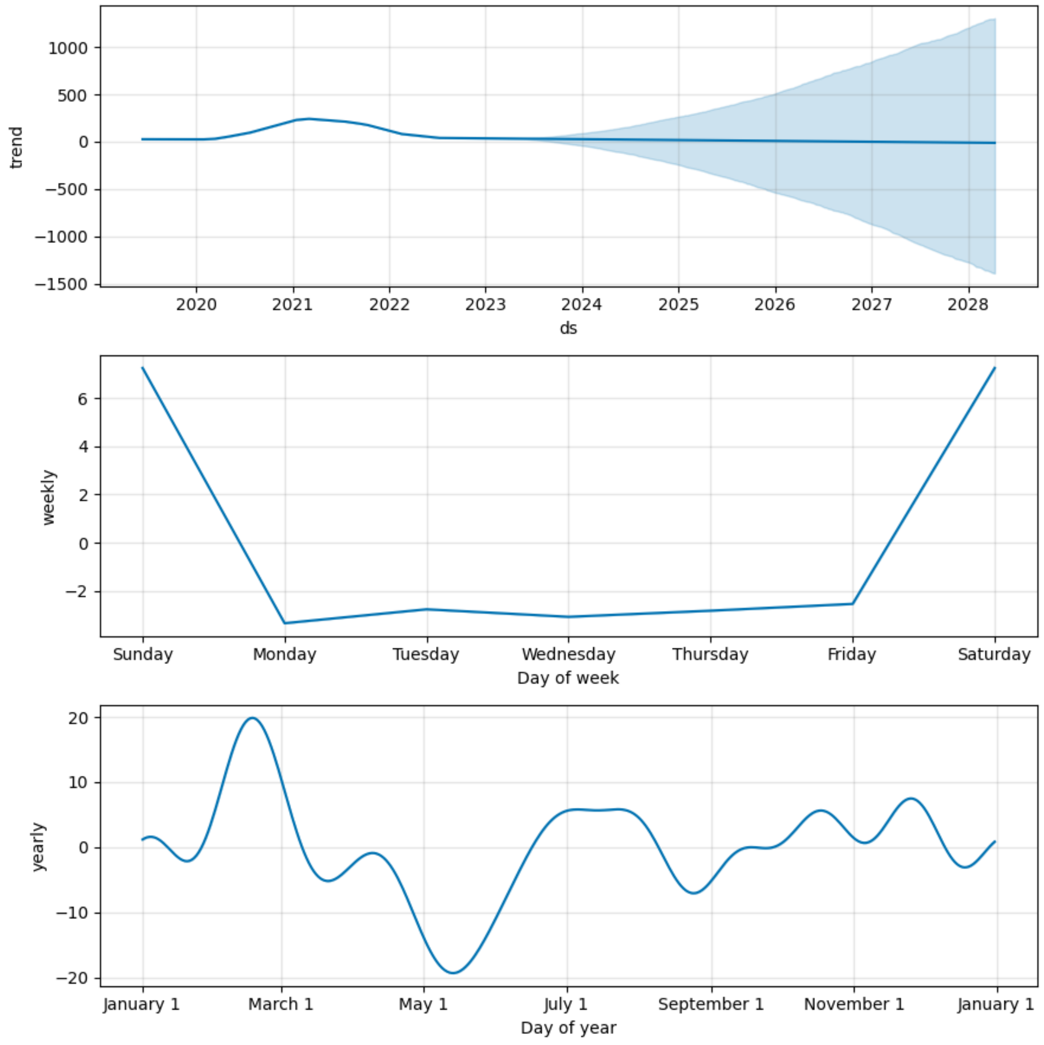
```
px.line(forecast_FVRR, x='ds',y='yhat')
```



```
figure = m.plot(forecast_FVRR, xlabel = 'ds', ylabel = 'y')
```



```
figure2=m.plot_components(forecast_FVRR)
```



```
columns=['Date', "Close"]
ndf = pd.DataFrame(ROKU_df, columns =columns)
```

```
prophet_ROKU_df = ndf.rename (columns = {'Date':'ds','Close': 'y' })
```

```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_ROKU_df)
```

```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/jgtv66hw.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/beq39rm2.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=74627', '
17:15:59 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:15:59 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9ab00d23d0>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_ROKU= m.predict(future)
```

```
forecast_ROKU
```

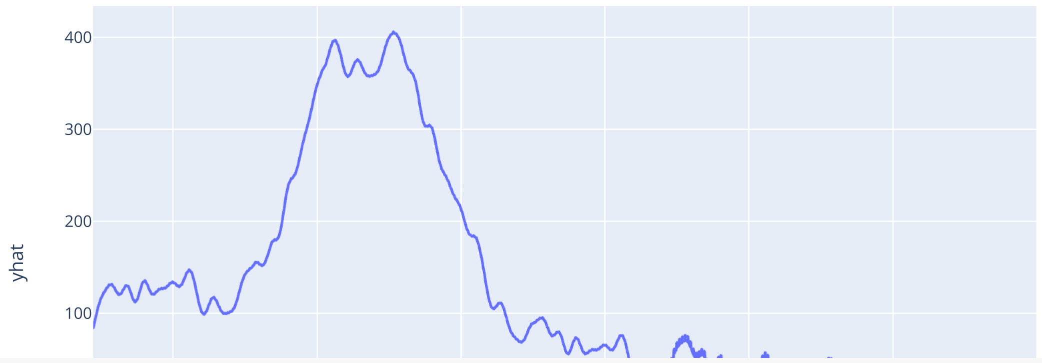
	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_te
0	2019-06-13	95.568665	54.815878	110.998125	95.568665	95.568665	-11.421010	
1	2019-06-14	95.841312	57.121079	111.278049	95.841312	95.841312	-10.434995	
2	2019-06-17	96.659251	63.111600	119.585030	96.659251	96.659251	-5.038165	
3	2019-06-18	96.931897	66.674354	121.115988	96.931897	96.931897	-2.854840	
4	2019-06-19	97.204543	67.202674	121.007794	97.204543	97.204543	-1.770651	
...	...	...	...	...	...	...	...	
2785	2028-04-06	-69.117646	-2570.760262	2136.060850	-2558.184390	2144.822144	-13.386239	
2786	2028-04-07	-69.184856	-2555.111452	2133.083051	-2562.812904	2147.515832	-13.119558	
2787	2028-04-08	-69.252065	-2546.477668	2137.460434	-2566.112496	2150.209520	-6.213885	
2788	2028-04-09	-69.319275	-2562.704950	2146.668549	-2567.946195	2152.903208	-5.627422	
2789	2028-04-10	-69.386484	-2545.200436	2143.270452	-2569.745345	2155.596895	-10.811091	

2790 rows × 19 columns

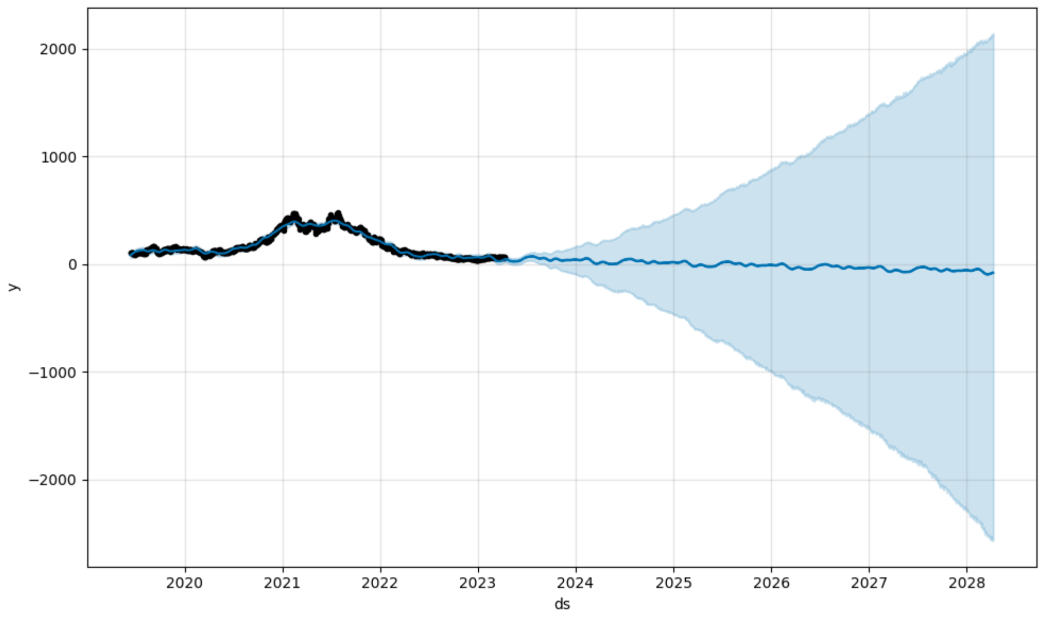
```
 # Display the underlying forecast dataframe (tail)
forecast_ROKU[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

	ds	yhat	yhat_lower	yhat_upper
2785	2028-04-06	-82.503884	-2570.760262	2136.060850
2786	2028-04-07	-82.304413	-2555.111452	2133.083051
2787	2028-04-08	-75.465950	-2546.477668	2137.460434
2788	2028-04-09	-74.946697	-2562.704950	2146.668549
2789	2028-04-10	-80.197575	-2545.200436	2143.270452

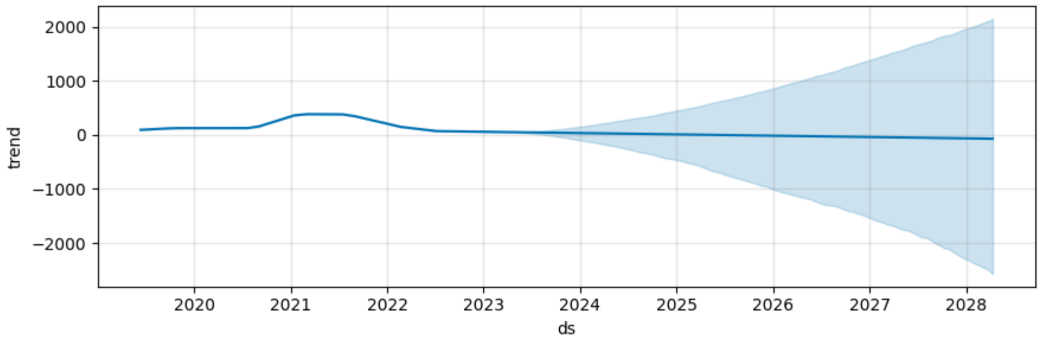
```
px.line(forecast_ROKU, x='ds',y='yhat')
```

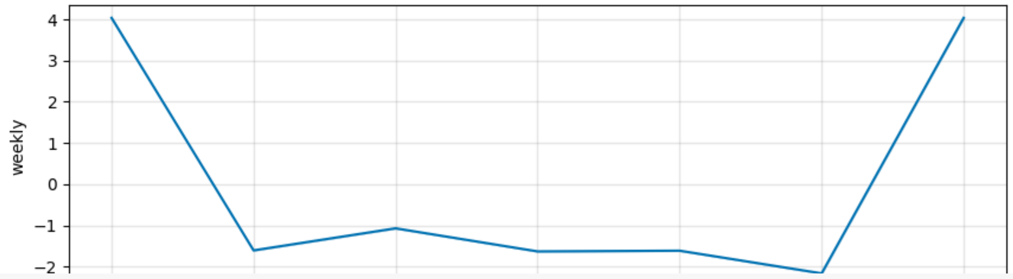


```
figure = m.plot(forecast_ROKU, xlabel = 'ds', ylabel = 'y')
```



```
figure2=m.plot_components(forecast_ROKU)
```





```
columns=['Date', "Close"]
ndf = pd.DataFrame(SAVA_df, columns =columns)
```

```
prophet_SAVA_df·=·ndf.rename·(columns·=·{'Date':'ds','Close':·'y'·})
```

```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_SAVA_df)
```

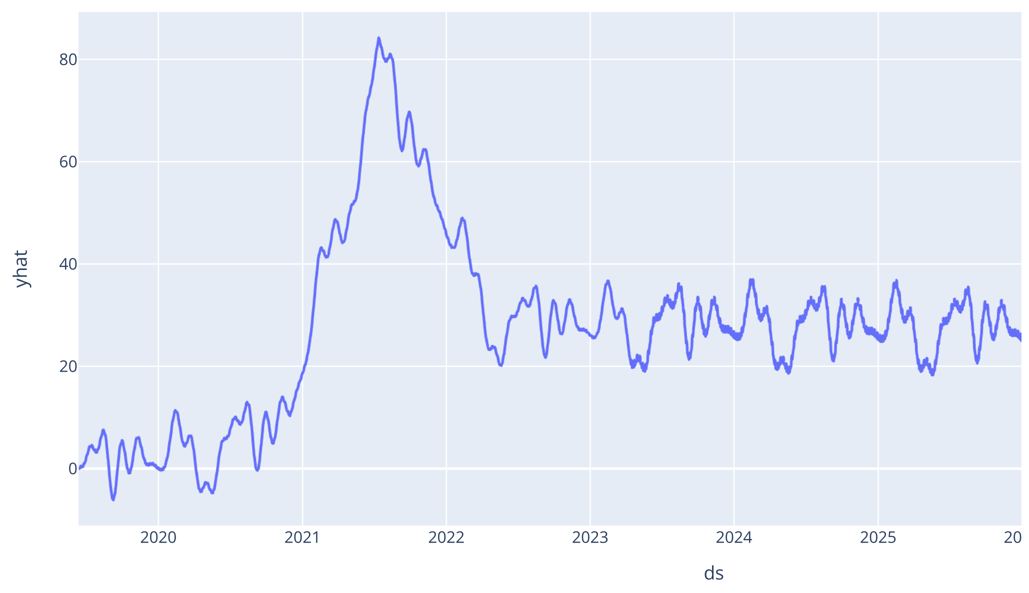
```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/c9o2s98n.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/b75as8mh.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.9/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=71424', '
17:16:02 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:16:02 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9aaaea1b20>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_SAVA= m.predict(future)
```

```
forecast_SAVA
```

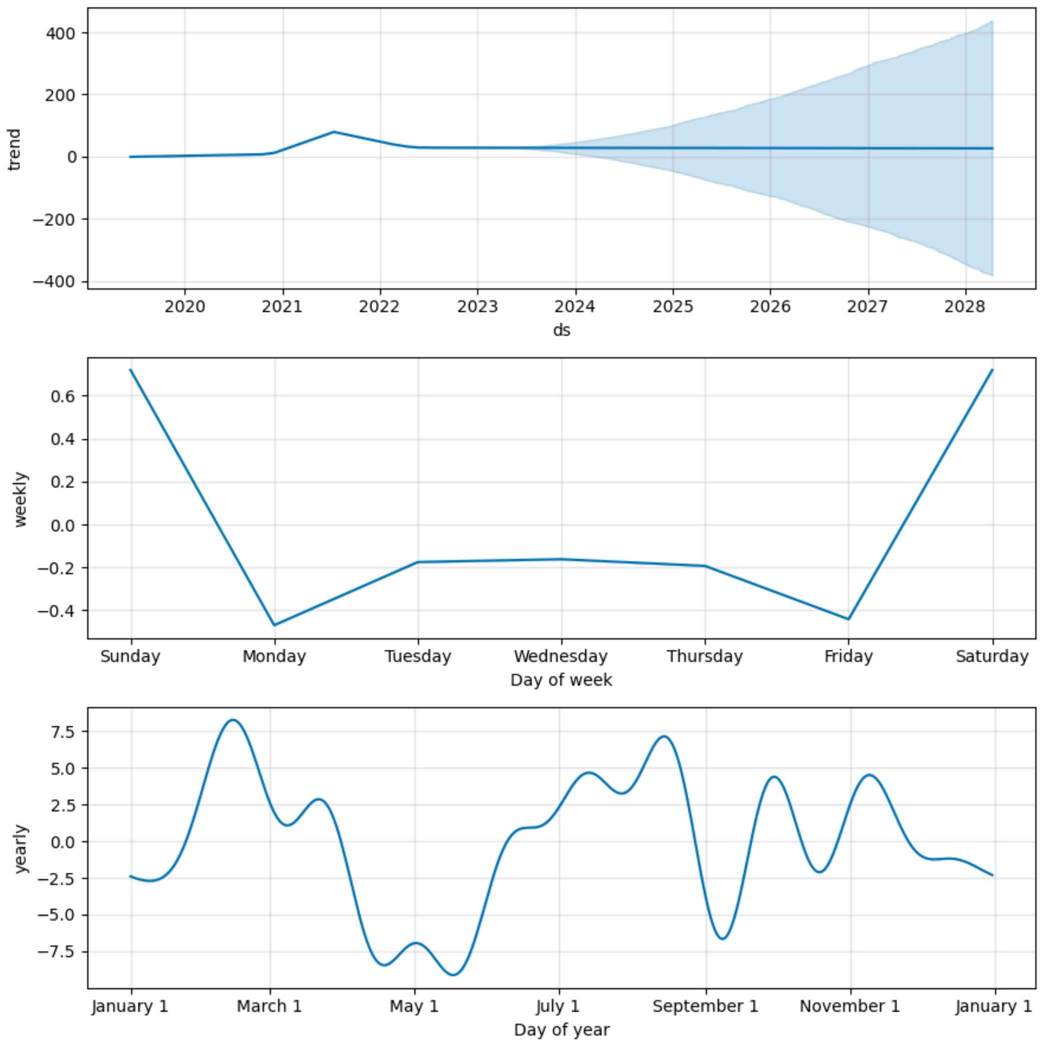
	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_term
0	2019-06-13	-0.313874	-10.751076	11.593996	-0.313874	-0.313874	0.464975	0
1	2019-06-14	-0.298771	-11.619794	11.311607	-0.298771	-0.298771	0.322636	0
2	2019-06-17	-0.253462	-12.274528	12.165609	-0.253462	-0.253462	0.438944	0
3	2019-06-18	-0.238359	-10.917909	11.929222	-0.238359	-0.238359	0.744279	0
	2019-							

```
px.line(forecast_SAVA, x='ds',y='yhat')
```



```
figure = m.plot(forecast_SAVA, xlabel = 'ds', ylabel = 'y')
```

```
figure2=m.plot_components(forecast_SAVA)
```



```
columns=['Date', "Close"]
ndf = pd.DataFrame(QFIN_df, columns =columns)
```

```
prophet_QFIN_df = ndf.rename (columns = {'Date':'ds','Close': 'y' })
```

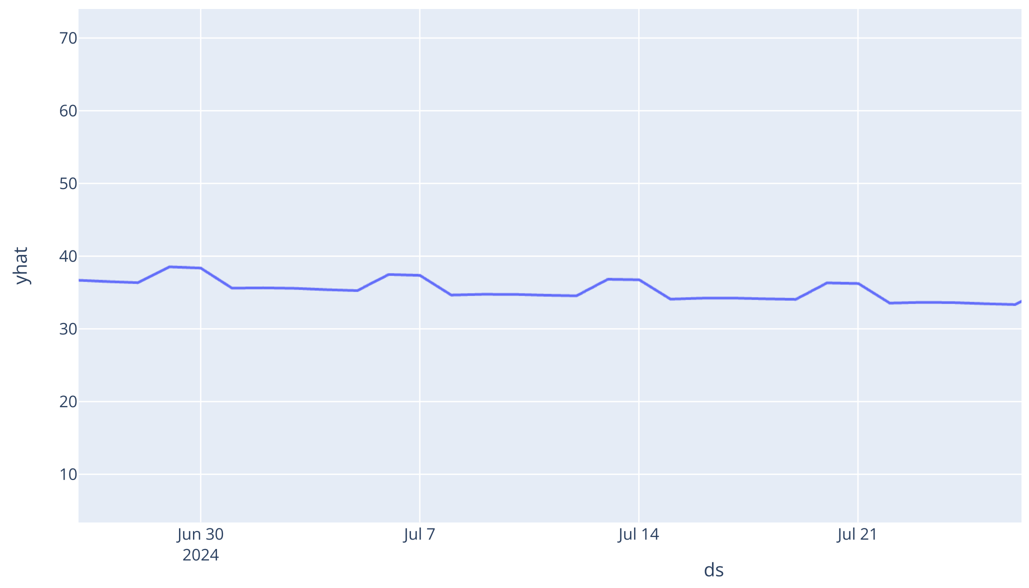
```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_QFIN_df)
```

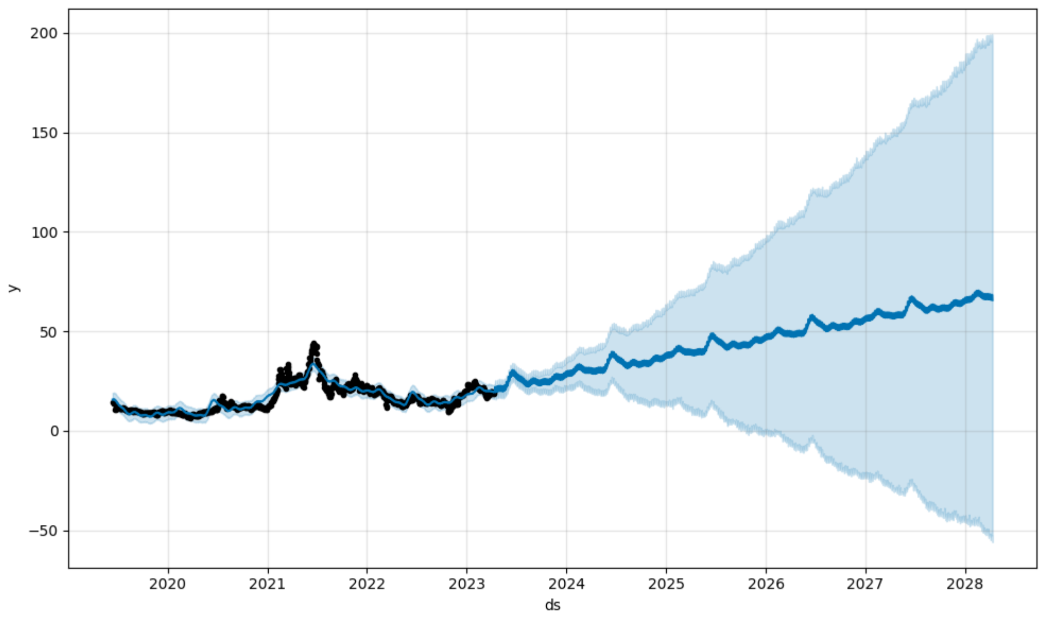
```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/3yc_pu9v.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/wog29t52.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.9/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=74700', '
17:16:05 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:16:05 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9aaac2dc10>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_QFIN= m.predict(future)
```

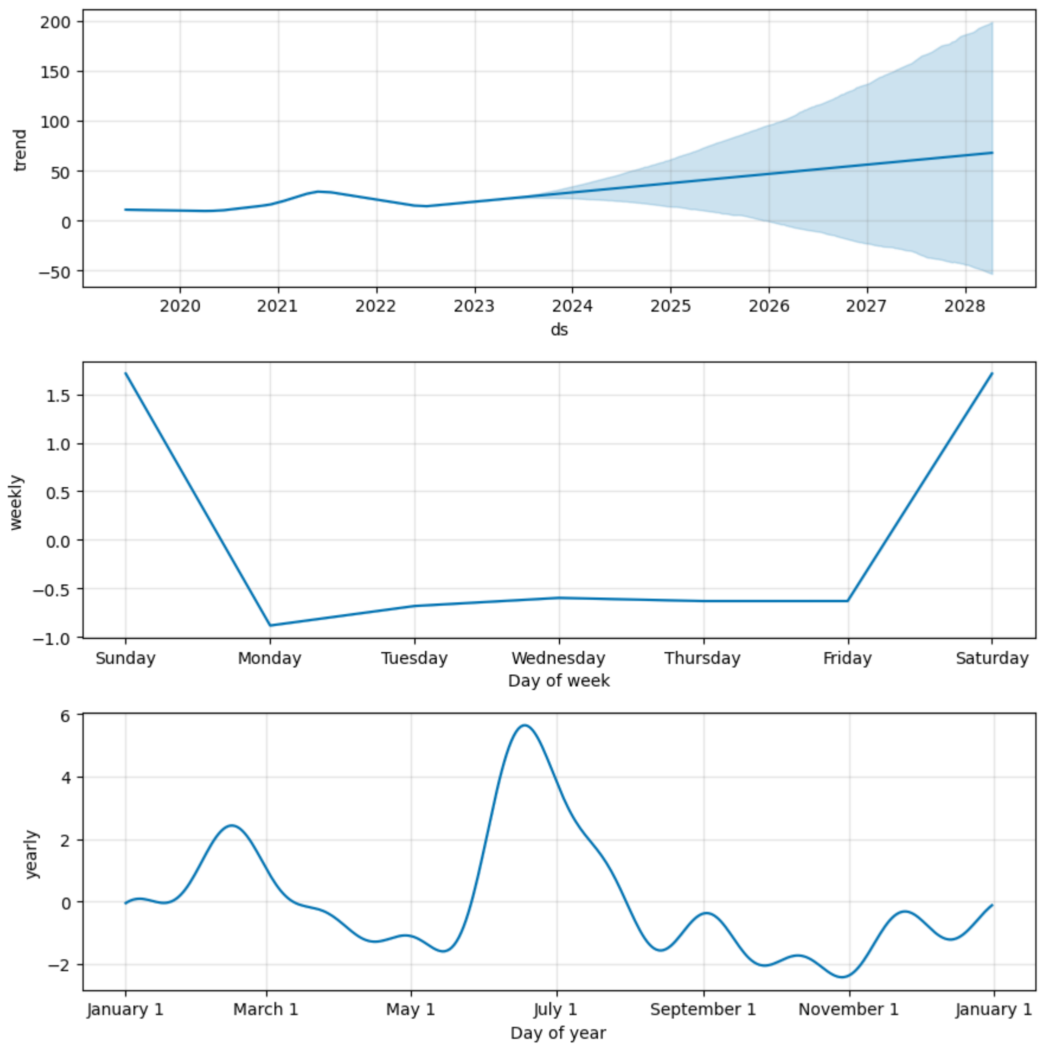
```
px.line(forecast_QFIN, x='ds',y='yhat')
```



```
figure·=·m.plot(forecast_QFIN,·xlabel·=·'ds',·ylabel·=·'y')
```



```
figure2=m.plot_components(forecast_QFIN)
```



```
columns=['Date', "Close"]
ndf = pd.DataFrame(TWLO_df, columns =columns)
```

```
prophet_TWLO_df = ndf.rename (columns = {'Date':'ds','Close': 'y' })
```

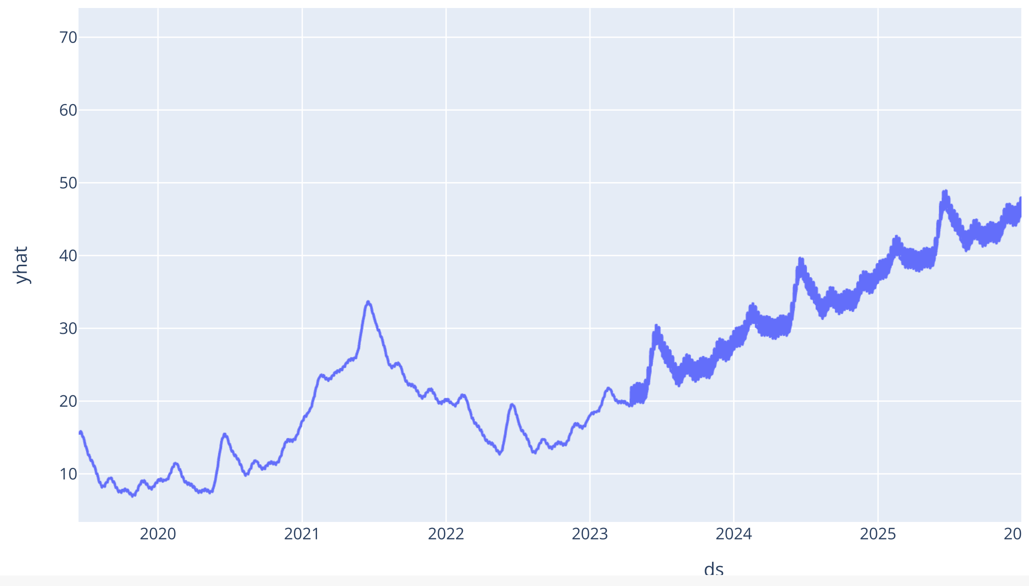
```
# Creating Prophet Model

m = Prophet()
m.fit(prophet_QFIN_df)
```

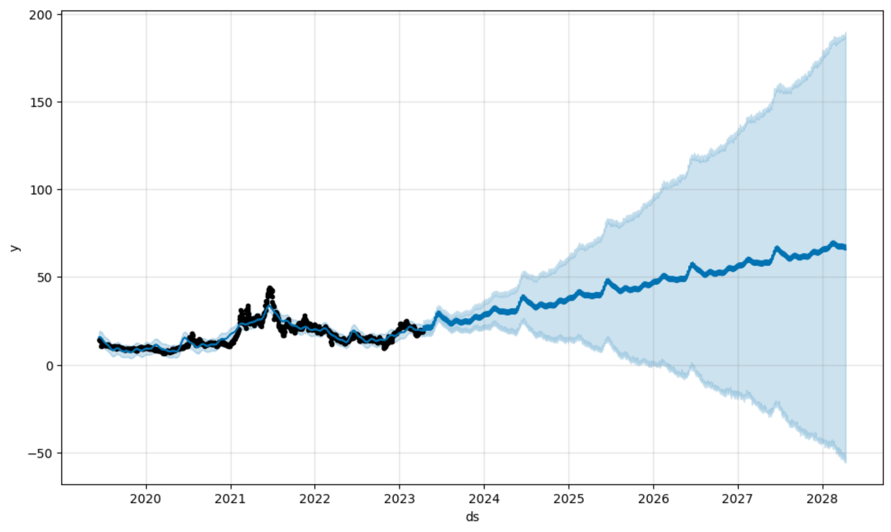
```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/ieostqp4.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpeqadd0kw/l6rh89i1.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.9/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=98522', '
17:30:36 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
17:30:36 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
<prophet.forecaster.Prophet at 0x7f9ab36a7d00>
```

```
future = m.make_future_dataframe(periods=365*5)
forecast_TWLO= m.predict(future)
```

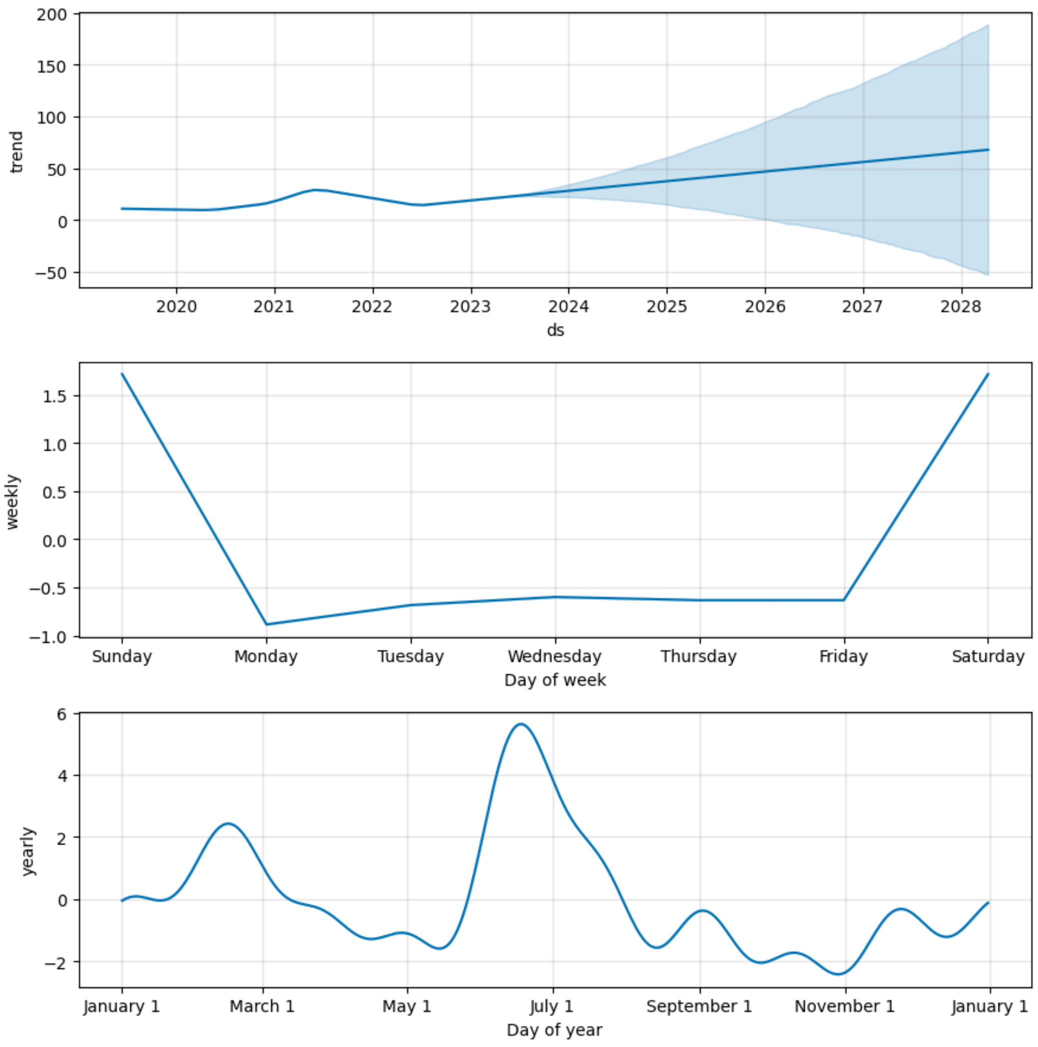
```
px.line(forecast_TWLO, x='ds',y='yhat')
```



```
figure = m.plot(forecast_TWLO, xlabel = 'ds', ylabel = 'y')
```



```
figure2=m.plot_components(forecast_TWLO)
```



✓ 1s completed at 1:34 PM