```
!pip install pystan==2.19.1.1 --quiet
!pip install fbprophet --quiet
!pip install yfinance --quiet
```

```
     |████████████████████████████████| 6.3 MB 40.0 MB/s
  Building wheel for yfinance (setup.py) ... done
```

```
import pandas as pd
import yfinance as yf
from datetime import datetime
from datetime import timedelta
import plotly.graph_objects as go
from fbprophet import Prophet
from fbprophet.plot import plot_plotly, plot_components_plotly
import warnings

warnings.filterwarnings('ignore')

pd.options.display.float_format = '${:,.2f}'.format
```

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

YFinance not only downloads the Stock Price data it also allows us to download all the financial data of a Company since its listing in the stock market.

```
today = datetime.today().strftime('%Y-%m-%d')
initial_date = '2019-01-01'

df = yf.download('BTC-USD',initial_date, today)

df.tail()
#df.info()
```

```
[**********100%**********]  1 of 1 completed
```

	Open	High	Low	Close	Adj Close	Volume

```
print(df.columns)
```

```
Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

2021-09-02	$48.807.85	$50.343.42	$48.652.32	$49.327.72	$49.327.72	39508070319

```
df.reset_index(inplace=True)
print(df.columns)
```

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

```
new_df = df[["Date","Open"]]
new_names = {
    "Date": "ds",
    "Open": "y",
}

new_df.rename(columns=new_names, inplace=True)
print(new_df.tail())
```

```
           ds          y
969 2021-08-31 $47,024.34
970 2021-09-01 $47,099.77
971 2021-09-02 $48,807.85
972 2021-09-03 $49,288.25
973 2021-09-04 $49,922.36
```

```
x = new_df["ds"]
y = new_df["y"]

fig = go.Figure()

fig.add_trace(go.Scatter(x=x, y=y))

# Set title
fig.update_layout(
    title_text="Time series plot of BTC Open Price",
)

fig.update_layout(
    xaxis=dict(
        rangeselector=dict(
            buttons=list(
                [
                    dict(count=1, label="1m", step="month", stepmode="backward"),
                    dict(count=6, label="6m", step="month", stepmode="backward"),
                    dict(count=1, label="YTD", step="year", stepmode="todate"),
                    dict(count=1, label="1y", step="year", stepmode="backward"),
```

```
                dict(step="all"),
            ]
        )
    ),
    rangeslider=dict(visible=True),
    type="date",
)
)
```


```
m = Prophet(
    seasonality_mode="multiplicative"
)

m.fit(new_df)
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to c
<fbprophet.forecaster.Prophet at 0x7f0a496090d0>
```

```
future = m.make_future_dataframe(periods = 365)
future.tail()
```

	ds
1334	2022-08-31
1335	2022-09-01
1336	2022-09-02
1337	2022-09-03
1338	2022-09-04

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

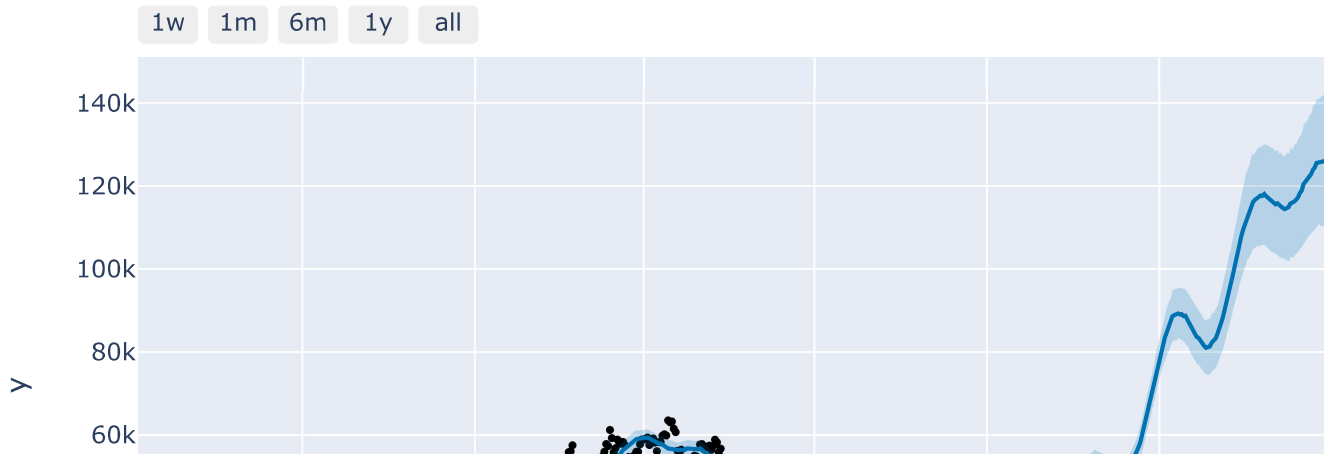
	ds	yhat	yhat_lower	yhat_upper
1334	2022-08-31	$82,967.83	$62,401.34	$104,445.29
1335	2022-09-01	$82,993.93	$63,319.22	$104,194.60
1336	2022-09-02	$82,438.56	$62,806.70	$103,202.67
1337	2022-09-03	$82,748.57	$63,146.70	$103,384.46
1338	2022-09-04	$81,955.94	$62,544.60	$102,628.90

```
next_day = (datetime.today() + timedelta(days=1)).strftime('%Y-%m-%d')

forecast[forecast['ds'] == next_day]['yhat'].item()
```

```
46290.504051758704
```

```
plot_plotly(m, forecast)
```




```
plot_components_plotly(m, forecast)
```