Share Price Forecasting Using Facebook Prophet

Read Discuss Courses Practice

Time series forecast can be used in a wide variety of applications such as Budget Forecasting, Stock Market Analysis, etc. But as useful it is also challenging to forecast the correct projections, Thus can’t be easily automated because of the underlying assumptions and factors. The analysts who produced accurate forecasts are also rare, and there is a big market available for them because it requires a substantial understanding of statistics and data analysis and has prior experience of producing time series forecasting.

Facebook open-sourced its time-series forecasting tool called Prophet in 2017 which produced accurate forecasts as produced by skilled analysts with a minimum amount of human efforts. The Facebook prophet is available in the form of API in Python and R/

How Prophet Works:

Facebook Prophet using Additive Regressive models using the following four components:

y(t) = g(t) + s(t) + h(t) + \epsilon_t

- **g(t):** A piecewise linear or logistic growth curve trend. Prophet automatically detects changes in trends by selecting change points from the data.
- **s(t):** A yearly seasonal component modeled using the Fourier series and weekly seasonal component using dummy variable
- **h(t):** A user-provided list of important holidays.
- **e_t:** Error term used by the prophet.

Advantages of Facebook Prophet:

the prophet is optimized for business-related problems that are encountered at Facebook, it has the following characteristics:

- The Facebook prophet is as accurate as a skilled analyst and can generate results in seconds
- Facebook prophet requires minimal data processing and can deal with several outliers and null values.
- User can add seasonality and holidays values manually, this can help easily integrate the particular domain knowledge.

In this post, we will use Facebook prophet with Python. We try to forecast the share price of Amazon Stock (from 2019-2020) using the share price data from (2015-2019).

Implementation:

- For this post, we will be using Amazon Stock Price data, it can be downloaded from yahoo finance website.
- First, we need to install the fbprophet tool, it can be installed with the following command in python.

```
Requirement already satisfied: fbprophet in /usr/local/lib/python3.6/dist-packages (0.6)
Requirement already satisfied: Cython>=0.22 in /usr/local/lib/python3.6/dist-packages (from
fbprophet) (0.29.21)
Requirement already satisfied: cmdstanpy==0.4 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (0.4.0)
Requirement already satisfied: pystan>=2.14 in /usr/local/lib/python3.6/dist-packages (from
fbprophet) (2.19.1.1)
Requirement already satisfied: numpy>=1.10.0 in /usr/local/lib/python3.6/dist-packages (from
fbprophet) (1.18.5)
Requirement already satisfied: pandas>=0.23.4 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (1.0.5)
Requirement already satisfied: matplotlib>=2.0.0 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (3.2.2)
Requirement already satisfied: LunarCalendar>=0.0.9 in /usr/local/lib/python3.6/dist-
packages (from fbprophet) (0.0.9)
Requirement already satisfied: convertdate>=2.1.2 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (2.2.1)
Requirement already satisfied: holidays>=0.9.5 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (0.9.12)
Requirement already satisfied: setuptools-git>=1.2 in /usr/local/lib/python3.6/dist-packages
(from fbprophet) (1.2)
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.6/dist-
packages (from fbprophet) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from
pandas>=0.23.4->fbprophet) (2018.9)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-packages (from
(from matplotlib>=2.0.0->fbprophet) (1.2.0)
Requirement already satisfied: pyparsing!=2.0.4, !=2.1.2, !=2.1.6, >=2.0.1 in
/usr/local/lib/python3.6/dist-packages (from matplotlib>=2.0.0->fbprophet) (2.4.7)
Requirement already satisfied: ephem>=3.7.5.3 in /usr/local/lib/python3.6/dist-packages
(from LunarCalendar>=0.0.9->fbprophet) (3.7.7.1)
Requirement already satisfied: pymeeus<=1, >=0.3.6 in /usr/local/lib/python3.6/dist-packages
(from convertdate>=2.1.2->fbprophet) (0.3.7)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from
holidays>=0.9.5->fbprophet) (1.12.0)
```

- Now, we need to import fbprophet and some other modules for data processing and plotting. We will use mean squared error and mean absolute error as our metrics.

Code:

python3

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import fbprophet as fbp
from sklearn.metrics import mean_squared_error, mean_absolute_error
# Use fivethirtyeight plot style
plt.style.use('fivethirtyeight')
```

- Now, we will read data from CSV file and put them into a pandas data frame.

Code:

```python
# We will be using amazon share price data which can be downloaded from YAHOO finance website.
df = pd.read_csv('sample_data / AMZN.csv')
df.head()
```

Output:

```
Date     Open     High     Low     Close     Adj Close     Volume
0     2015-07-21     487.899994     488.880005     482.549988     488.000000     488.000000
3181800
1     2015-07-22     485.989990     492.500000     484.899994     488.269989     488.269989
3114900
2     2015-07-23     491.660004     491.660004     475.700012     482.179993     482.179993
9374400
3     2015-07-24     578.989990     580.570007     529.349976     529.419983     529.419983
21909400
4     2015-07-27     527.750000     544.950012     526.599976     531.409973     531.409973
7491000
```

- Since we need only two columns date and adjusted close price, so, we subset the original dataset to get these columns. Since the data is required in a prophet in the form of two columns named ds (for date column) and y (for data column).

Code:

```
# add two columnsin dataframe having values as Date and Adj Close
df[['ds', 'y']] = df[['Date', 'Adj Close']]
# Subset two columns from data frame
df = df[['ds', 'y']]

df.head()
```

Output:

```
ds    y
0    2015-07-21    488.000000
1    2015-07-22    488.269989
2    2015-07-23    482.179993
3    2015-07-24    529.419983
4    2015-07-27    531.409973
```

- Now, we split the data frame into train and test data, we will be using 4 years of data for training and a year of data for test purpose.

Code:

python3

```
# split data frame  into two parts train and test
split_date = "2019-07-21"
df_train = df.loc[df.ds <= split_date].copy()
df_test = df.loc[df.ds > split_date].copy()
```

- Now, we instantiate the Facebook prophet API, this prophet API works similar to scikit-learn. It uses the fit function to fit the dataset into the model and predict function to forecast future values.

Code:

python3

```python
# Instantiate prophet
model = fbp.Prophet()
# fit the training data
model.fit(df_train)
```

- Now, we use predict function to forecast the share price for next 1 year.

Code:

python3

```python
forecast = model.predict(df_test)
forecast.tail()
```

Output:

```
yearly    yearly_lower    yearly_upper    multiplicative_terms    multiplicative_terms_lower
multiplicative_terms_upper    yhat
247    2020-07-14    1992.862925    1479.553875    2566.925238    1403.962381    2483.045869
93.536964    93.536964    93.536964    -25.535936    -25.535936    -25.535936    119.072900
119.072900    119.072900    0.0    0.0    0.0    2086.399889
248    2020-07-15    1993.215324    1485.368711    2575.314593    1401.835761    2485.386736
97.405883    97.405883    97.405883    -25.138654    -25.138654    -25.138654    122.544537
122.544537    122.544537    0.0    0.0    0.0    2090.621207
249    2020-07-16    1993.567723    1484.197262    2589.201052    1399.740456    2487.727602
100.236350    100.236350    100.236350    -25.549805    -25.549805    -25.549805
125.786155    125.786155    125.786155    0.0    0.0    0.0    2093.804073
250    2020-07-17    1993.920121    1478.807958    2617.093500    1397.645151    2490.068469
99.309824    99.309824    99.309824    -29.445843    -29.445843    -29.445843    128.755666
128.755666    128.755666    0.0    0.0    0.0    2093.229945
251    2020-07-20    1994.977318    1475.034301    2618.609494    1389.089958    2497.091069
104.649308    104.649308    104.649308    -31.050560    -31.050560    -31.050560
135.699868    135.699868    135.699868    0.0    0.0    0.0    2099.626626
```

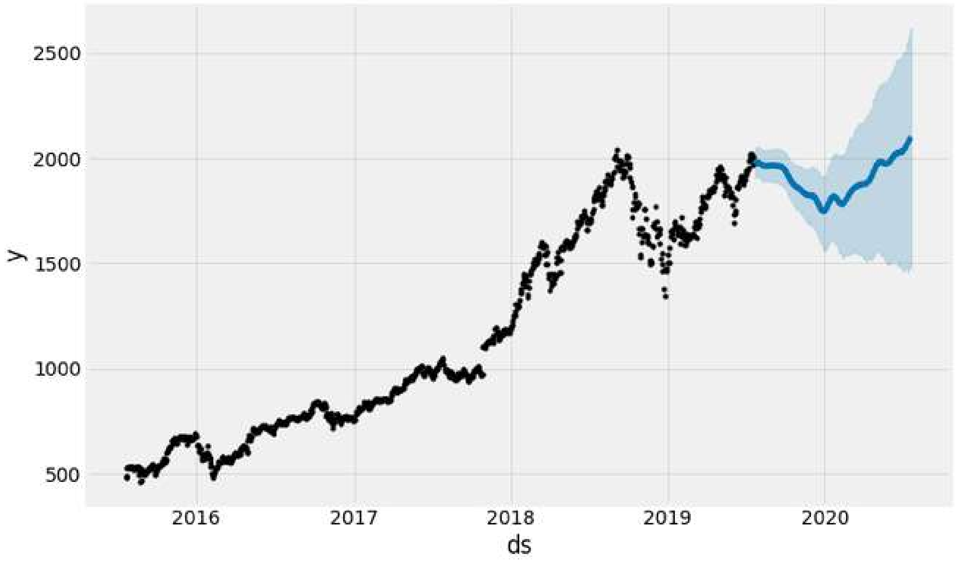
- As we can see this column contains the date column, predict share price (y_hat), lower and upper estimates of it, trend components, seasonal components (weekly and yearly).

Code:

python3

```
model.plot(forecast)
```

Output:



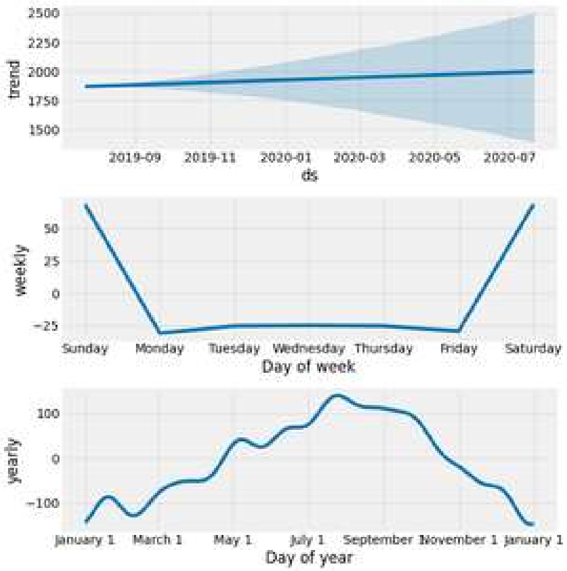
- The command will plot the components of the prophet such as: trend line, weekly and yearly seasonality.

Code:

python3

```
# plot graphs of different components:
model.plot_components(forecast)
```

Output:



- Now, we calculate the mean square error and mean absolute error for the forecasted data.

Code:

python3

```
# code
print("Mean Squared Error (MSE):", mean_squared_error(y_true = df_test["y"], y_pred = forecast['yhat
print("Mean Absolute Error (MAE):", mean_absolute_error(y_true = df_test["y"], y_pred = forecast['yh
```

Output:

```
Mean Squared Error (MSE): 121417.80253038534
Mean Absolute Error (MAE): 246.57694290710793
```

- Now, we calculate the mean absolute percentage error of our forecast, because it gives a better idea about how accurate our prediction is

Code:

python3

```python
def mean_abs_perc_err(y_true, y_pred):
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

print("Mean Absolute % Error (MAPE): ", mean_abs_perc_err(y_true = np.asarray(df_test["y"]), y_pred =
```

Output:

```
Mean Absolute % Error (MAPE):  10.693787212532687
```