Stock Forecast

Using Facebook's Prophet forecasting library to predicate stock prices

Environment Requirements

Python >= 3.7.3

Python libraries:

- streamlit
- yfinance
- pystan
- Prophet
- fbprophet
- plotly

Execution

In [ ]:
```
$ python3 .src/main.py
```

Import Modules

In [ ]:
```
import streamlit as st
from datetime import date

import yfinance as yf
from fbprophet import Prophet
from fbprophet.plot import plot_plotly
from plotly import graph_objs as go
```

Data capture points

We declare the point in time using a string value when we want to start running the forecast from and also todays date

In [ ]:
```
START = "2000-01-01"
TODAY = date.today().strftime("%Y-%m-%d")
```

Application data

- Construct an application title
- A tuple of stocks, using a the standard abbrevaied stock name, that we want to analyse
- A select box widget to choose a sepefic stock
- Slider widget to select the number of year that we want to get a predication for

In [ ]:
```
st.title('Stock Forecast App')

stocks = ('TSLA', 'AAPL')
```

```
selected_stock = st.selectbox('Select dataset for prediction', stocks)

n_years = st.slider('Years of prediction:', 1, 5)
period = n_years * 365
```

Fetching data

A function to analyse stock data using Yahoo Finance

- The fuction will download all the data from the date as decalred by START varibale and the TODAY varibale
- The data that is returned is already in Pandas data set
- Using inplace=True ensures that the date is returned in the first column
- Cache the data for each specfic stock

Function

In [ ]:
```
@st.cache
def load_data(ticker):
    data = yf.download(ticker, START, TODAY)
    data.reset_index(inplace=True)
    return data
```

User Notification

Call the above function giving sufficient user feedback

- Subtext widet indicating that data is being fetched
- Subtext widget indicating that fecting the data has completed

In [ ]:
```
data_load_state = st.text('Loading data...')
data = load_data(selected_stock)
data_load_state.text('Loading data... done!')
```

Plot the data

- Streamlit can handle Pandas dataframe

The data table that is produced should be in the following columated format:

Today's date | Open Price | High Price | Low Price | Close Price |

Raw data frame

In [ ]:
```
st.subheader('Raw data')
st.write(data.tail())
```

Graph

- Using a ploty graph object
- Create a graph object figure
- Scatter using named X and Y axis for a stocks opening and close price
- Provide a slider on the graph to select specfic time series

In [ ]:
```
# Plot raw data
def plot_raw_data():
        fig = go.Figure()
        fig.add_trace(go.Scatter(x=data['Date'], y=data['Open'], name="stock_
        fig.add_trace(go.Scatter(x=data['Date'], y=data['Close'], name="stock
        fig.layout.update(title_text='Time Series data with Rangeslider', xax
        st.plotly_chart(fig)

plot_raw_data()
```

Forecast using ML Library

Before executing the training model Facebook Prophet expects the dataframe to be in a specific format:

Dataframe with two columns: ds and y. The ds (datestamp) column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The y column must be numeric, and represents the measurement we wish to forecast.

In this example:

- ds = Date
- y = Close (Stock closing price)

Data Training

In [ ]:
```
# Predict forecast with Prophet.
df_train = data[['Date','Close']]
df_train = df_train.rename(columns={"Date": "ds", "Close": "y"})
```

Facebook Prophet Model

- Fit training data
- For the forecase we need a date into the future. Using the previously declared period varibale which is decalred as an integer value based on number of days per year
- Forecase data is held in a data frame

In [ ]:
```
m = Prophet()
m.fit(df_train)
future = m.make_future_dataframe(periods=period)
forecast = m.predict(future)
```

Forecast data

- Using previous logic to show a raw dataframe we can display the Forecast data via a table

In [ ]:
```
# Show and plot forecast
st.subheader('Forecast data')
st.write(forecast.tail())
```

Plotting the forecast

- Using Ploty function

- Using previously saved varibales containing the forecast model and the forecast dataframe
- Using streamlit to plot the

In [ ]:
```
st.write(f'Forecast plot for {n_years} years')
fig1 = plot_plotly(m, forecast)
st.plotly_chart(fig1)
```

Plotting individual components

- Using Steamlit to display forecasted components

In [ ]:
```
st.write("Forecast components")
fig2 = m.plot_components(forecast)
st.write(fig2)
```