Financial Time Series Prediction

David Ocepek

October 2021

1 Introduction

Stock markets are one of the goto's data and computer scientists as they provide all of the essentials for fruitful research: they're highly non-linear therefore challenging, have large quantities of data easily accessible and improved predictions are of great importance to users and even more so to institutions.

Our project will be to use common ML models to predict the financial time-series of several stocks and analyze the results.

Some parts are marked as optional as they will be implemented depending on the time available.

2 Evaluated Models

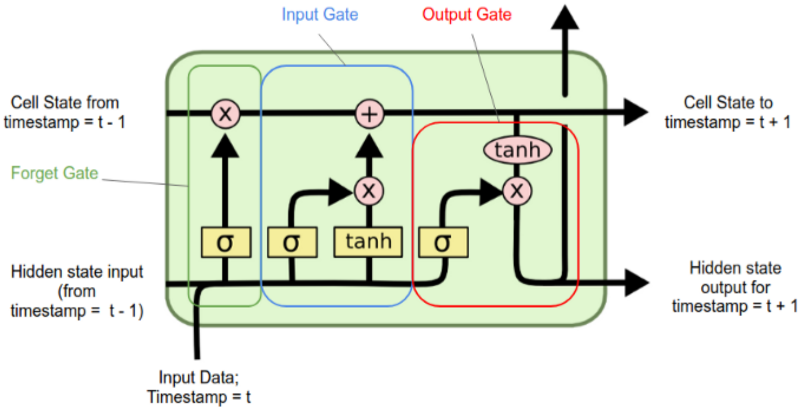
We have choosen to use ARIMA, LSTMs, CNNs, CNN-LSTM to model our time series.

2.1 ARIMA

Autoregresive Integrated Moving Average is a combination of Linear Regression and the moving average. It is one of the most commonly used algorithms for financial time series prediction. ARIMA mainly models the seasonality and trend of a series. This model will serve as the baseline for our models.

2.2 LSTM

LSTMs are most commonly associated with time series modeling(citation). They are a form of RNNs; their core building block being the LSTM cell.


Input Gate
Output Gate
Cell State from timestamp = t - 1
Forget Gate
Cell State to timestamp = t + 1
tanh
σ
σ
tanh
σ
Hidden state input (from timestamp = t - 1)
Hidden state output for timestamp = t + 1
Input Data; Timestamp = t

A LSTM layer takes as input a sequence of features from a time window up to but not including t and returns the value of the time series at time t. It is mainly used for modeling long-term dependencies.

In our work we decided to use a many-to-one stacked LSTM-RNN to model our time series. Currently our LSTM model uses only a one dimensional time series, however we intend to add hand-crafted features such moving averages, market volume, both opening and closing value as well as average value etc.

2.3 CNN

While slightly counter-intuitive CNNs have been shown to be quite effective in financial time series modeling. CNNs use convolutional filters to extract hierarchical from data sets. This is especially usefull when we have multiple features.

2.4 LSTM-CNN

These networks use LSTM for modeling long term dependencies and CNN for modeling hierarchical dependencies.

3 Datasets

For our datasets we will be using the 3-5 datasets of daily stock prices for different stocks.

All our stock data will be acquired from Yahoo! Finance historical database of stocks. Our training sets will most likely have a time period of 4 to 8 years while our test set will have a time period from 1 month to 1 year.

4 Implementation

4.1 Preprocessing of Data

4.1.1 Normalization

We have normalized our stacked LSTM with MinMaxScaler, however for CNN we shall use Standard Scaler.

4.1.2 Seasonal Decomposition(Optional)

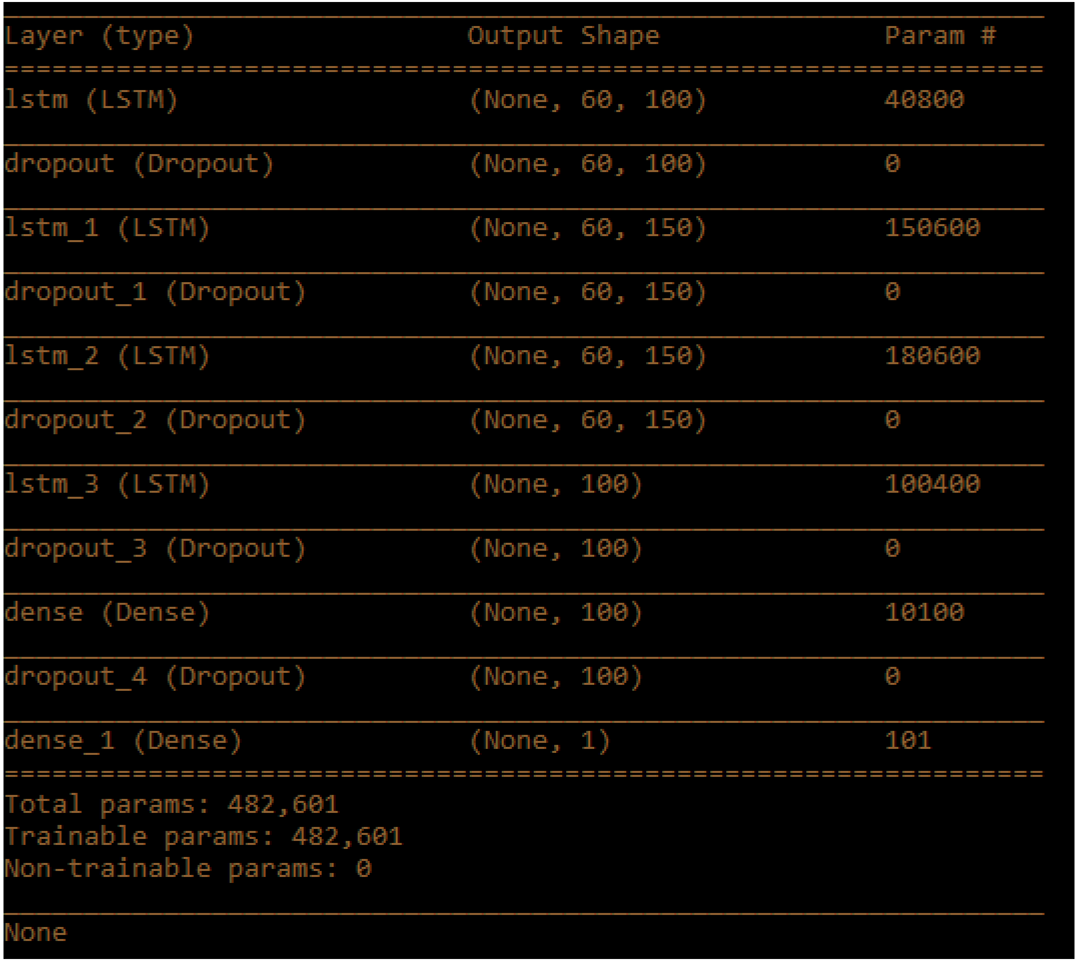
Time series are commonly decomposed into three characteristics:

- Trend
- Seasonality
- Residuals

With Vawelet Transforms we can remove the noise from the model. Deconstruct the series and the reconstruct it. Some series propose using ARIMA to model the linear part of the series, using the CNN to model the hierarchical part and LSTM the long-term dependencies.

4.1.3 Models

Currently implemented stacked LSTM



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 60, 100)           40800
_________________________________________________________________
dropout (Dropout)            (None, 60, 100)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 60, 150)           150600
_________________________________________________________________
dropout_1 (Dropout)          (None, 60, 150)           0
_________________________________________________________________
lstm_2 (LSTM)                (None, 60, 150)           180600
_________________________________________________________________
dropout_2 (Dropout)          (None, 60, 150)           0
_________________________________________________________________
lstm_3 (LSTM)                (None, 100)               100400
_________________________________________________________________
dropout_3 (Dropout)          (None, 100)               0
_________________________________________________________________
dense (Dense)                (None, 100)               10100
_________________________________________________________________
dropout_4 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 101
=================================================================
Total params: 482,601
Trainable params: 482,601
Non-trainable params: 0
_________________________________________________________________
None
```

For weight initialization we used GlorotUniform and we used very small R2 regularization of 0.00001 for all layers. The model was training for 200 epochs with early stopping (but it did not stop).

The other two models have yet to be implemented.

4.1.4 Fitting

For the fitting process 3 to 10 fold cross validation will be used.

Our Model is already using Model Checkpointing and Early stopping with patiance of 20 epochs with respect to RMSE.

5 Evaluation Metrics

We will evaluate root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) and absolute percentage standard deviation (SDAPE).

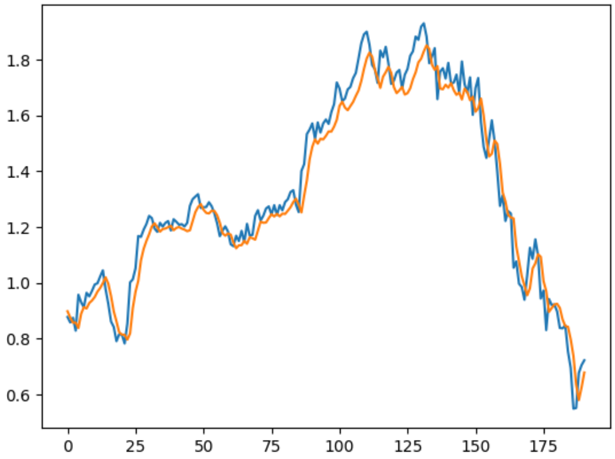
6 Used Libraries

Our project will be developed in Python.

For our neural networks we will use Keras Sequential model. Our ARIMA model will be implemented with Python's statsmodel. And we have yet to decide which Python library we will use for vawelet transform (PyVawelet seem promisng).

7 Achived Results (So far)

We have trained our stacked LSTM on Apple opening price for the time period from the 31. dec. 2013 to 29. dec 2017 with a memory window of 60 days. We tested our model on Apple 2. jan 2018 to 31. dec 2018. and got the following graph:



The scale is normalized with the MinMax scale transform the training set. Our RMSE was 0.05 on our transformed data.

8 Reproducibility

All data and trained models will be saved and availible on my Github.