Journal of Modeling and Optimization 2020;12(2):84-99
https://doi.org/10.32732/jmo.2020.12.2.84

Using Machine Learning Algorithms on Prediction of Stock Price

Li-Pang Chen*

Department of Statistical and Actuarial Sciences, University of Western Ontario, 1151 Richmond St., London, Ontario, N6A 3K7, Canada
Email: lchen723@uwo.ca (Corresponding author)

Received: 24 April 2020; Accepted: 18 May 2020; Available online:30 June 2020

Abstract: In this paper, we investigate analysis and prediction of the time-dependent data. We focus our attention on four different stocks are selected from Yahoo Finance historical database. To build up models and predict the future stock price, we consider three different machine learning techniques including Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN) and Support Vector Regression (SVR). By treating close price, open price, daily low, daily high, adjusted close price, and volume of trades as predictors in machine learning methods, it can be shown that the prediction accuracy is improved.
Keywords: Data analysis; Long short-term memory; Neural networks; Prediction; Support vector regression; Time-dependent.

1. Introduction

Investors in the stock markets have been attempting for a long time to develop methods for predicting stock prices to make more profit. However, making precise and consistently correct forecasts is complicated because of various factors, such as military war or politics, which affect investors' sentiments and thus, causing stock market movements. As a result, developing a consistent and accurate method is not easy. Even so, it is crucial to construct proper models and obtain reliable results.

An important feature of stock price is *time-dependent*. To characterize such a time-dependent data, time series analysis has been a widely used approach. The most popular method in modeling time series data is Auto Regressive Integrated Moving Average (ARIMA) models proposed by [1], and the ARIMA models are frequently used in practical applications (e.g., [2]). However, this is a linear model and does not consider the volatility or the movements in the variance of the underlying stock prices. That makes ARIMA unsuitable for forecasting stock price data, especially when there are fluctuations in the volatilities. However, with the recent advancements in technology and data science, machine learning has been popular and many methods have been fully discussed. For example, [3] used financial time series analysis based on the wavelet kernel support vector to forecast the Nasdaq composite index. The wavelet kernel SVMs were shown to increase prediction accuracy compared with the polynomial kernel SVM and Gaussian kernel SVM. [4] studied several individual and hybrid computational intelligence methods based on statistical learning algorithm. [5] proposed to integrate wavelet transform, Multivariate Adaptive Regression Splines (MARS), and Support Vector Regression (SVR) and improve prediction accuracy. More discussions regarding general machine learning methods and related applications are reviewed in [6], [7], and [8].

While most of the previous works focused on developing and improving a particular machine learning algorithm, the purpose of this paper is to develop different machine learning methods and compare their performances based on their accuracies. In particular, we consider the Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN) and Support Vector Regression (SVR) methods. Regarding data analysis, we are interested in predicting stock prices of Apple, Mastercard, Ford, and ExxonMobil that are collected from Yahoo Finance. The daily variables of each stock are considered as models' features with a lookback window of 30 to 180 days. This approach creates a high-dimensional dataset that needs to be pre-processed before applying machine learning models.

The remainder is organized as follows. In Section 2, we introduce the real dataset that we are going to analyze in this paper. In Section 3, we introduce several machine learning methods. In Section 4, we introduce notation and algorithms that will be implemented by the computational programming. In Section 5, we present real data analysis and its results. Finally, we conclude the article with discussions in Section 6.

2. Data description

The datasets used in this project include Apple, Mastercard, Ford, and ExxonMobil historical data obtained from Yahoo Finance. The data contain open price, close price, daily high price, daily low price, adjusted close price and the volume of trades for these stocks from January 1st, 2002 until March 11th, 2020 (except for Mastercard where data are available from 2006). The adjusted close price of these stocks are shown in the Figure 1. (Daily log-returns presented in Appendix A).

Figure 1. Stock prices for Apple, Mastercard, Ford, and ExxonMobil

Traditionally financial time series are assumed to be stationary, and if not, number of techniques have been proposed to transform data into a stationary process. Other than the stationarity, ergodicity is also required to ensure that the ensemble average for a quantity converges to the expected value of that quantity's time average. Regarding these issues, investigating statistical properties of data is a common approach when it comes to stock price forecasting. A comprehensive analysis was conducted on the properties of datasets and the results are presented in Appendix A. Some of these properties which were investigated for our data are:

1) Sharp peak and heavy tailed (non-normal) distribution of returns (contrary to the Black-Scholes Normal distribution assumption)

This can be noticed by referencing to Table 1, and some visualize evidence is also displayed in Figures A2, A3, A4, and A5, where log-returns distributions are better fitted to a Student-t distribution rather than the normal distribution.

Table 1. Kurtosis and Skewness metrics of the financial data

	Kurtosis	**Skewness**
AAPL	5.642884	-0.199756
MAST	8.719550	0.362088
FORD	17.013923	-0.016349
EXON	12.596108	-0.194201

2) The high degree of variability in returns which is observed in the form of many bursts in returns plot (as is shown in Figure A1).

3) Volatility clustering.

There are periods of high volatility observable in both Figure A1 and Figure A7.

4) Weak (linear) correlation between the returns and their lagged values (as is shown in Figure A6).

One of the critical characteristics of stock price and return data is the presence or absence of dependence between price increments. The presence of linear dependence can be readily checked by the autocorrelation function. Moreover, it is well known that in efficient markets, the price follows a random walk, and the linear dependence, will decay rapidly to zero. This is reason that traditional time series approaches cannot discriminate returns and white noise (containing all frequencies).

5) Long range dependence detectable by nonlinear autocorrelations of returns

As mentioned above, the linear independence fades when dependence is investigated on the squared return data. As shown in Figure A7, there is considerable dependence especially on the most recent returns. However, this dependence also dissipates as time moves forward.

3. Model and methodology

3.1 Support Vector Machines (SVM)

Support vector machines (SVM) were initially proposed as a classification method (e.g., [9]), which uses a hyperplane (linear surface) as a boundary between the two classes of observations maximizing the distance or the margin of the nearest point from each class to that separating hyperplane. However, this simple definition of SVM is valid only when the classes are separable by a linear boundary. If the classes are not separable, the margins are relaxed by soft margins, allowing for some observations to be within the boundary or even to the other side of the hyperplane. When the boundary is truly non-linear, then a linear hyperplane based SVM cannot satisfactorily classify observations into classes. To overcome this problem, the observations are projected into higher dimensional space using kernel-based basis functions so that the boundary becomes linear in that higher dimensional space. The choice of tuning parameters in the kernel function provides the flexibility of the SVM models to model the right nonlinear boundary.

To describe the SVM classifier explicitly, let us consider the training data $(y_i,\ x_i)$ for $i = 1, \dots, T$, where $y_i = +1$ for upward movement of the stock prices, $y_i = -1$ for downward movement, and x_i is the p-dimensional vector of covariate and past stock prices. Then a classifier $f(x) \geq 0$ if $y_i = 1$ and $f(x) < 0$ for $y_i = -1$ will optimally classify the observations, that is, we have $y_i f(x_i) > 0$. A linear classifier is given by

$$f(x) = \beta_0 + \beta^{\top} x, \tag{1}$$

where β is the p-dimensional vector of parameters. We maximize the margin of this hyperplane to the nearest points from the two classes, that is, $y_i f(x_i)$. The margin is given by

$$\frac{\beta^{\top} x_+ - \beta^{\top} x_-}{\|\beta\|} = \frac{2}{\|\beta\|}. \tag{2}$$

Therefore, the optimization problem can be written as

$$\text{maximize} \frac{2}{\| \beta \|}$$

subject to

$$y_i(\beta_0 + \beta^{\top} x) \geq 1 \ \text{ for } i = 1, \dots, T,$$

or equivalently,

$$\text{minimize} \frac{1}{2} \| \beta \|^2$$

subject to

$$y_i(\beta_0 + \beta^{\top} x) \geq 1 \ \text{ for } i = 1, \dots, T.$$

Moreover, by the Lagrange multiplier method, the optimization problem can be written as

$$\text{minimize } \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{T} \epsilon_i \tag{3}$$

subject to

$$y_i(\beta_0 + \beta^\top x) \geq 1 - \epsilon_i, \qquad \epsilon_i > 0, \text{ for } i = 1, \dots, T,$$

where C is a regularization parameter with a large value of C. We can write the optimization problem as

$$\text{minimize} \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{T} \max\big(0, 1 - y_i(\beta_0 + \beta^T x_i)\big), \tag{4}$$

and (4) is indeed a convex optimization problem. The dual version of the optimization problem can be written as

$$\max_{\alpha_i \geq 0} \sum_{i=1}^{T} \alpha_i - \frac{1}{2} \sum_{j=1}^{T} \sum_{k=1}^{T} \alpha_j \alpha_k y_j y_k x_j^\top x_k \text{ subject to } 0 \leq \alpha_i \leq C, \sum_{i=1}^{T} \alpha_i y_i = 0. \tag{5}$$

If we consider a dual version of the classifier $f(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i y_i (x_i^T x) + b$ then the objective function of the dual becomes

$$\sum_{i=1}^{T} \alpha_i - \frac{1}{2} \sum_{i=1}^{T} \alpha_i y_i f(x_i). \tag{6}$$

In order to accommodate for the nonlinear boundary, the feature vectors x_i is projected on to a higher dimensional space through a nonlinear function ϕ: $\mathbb{R}^p \to \mathbb{R}^d$, where $d \geq p$ and then apply SVM classifier so that the classifier becomes $f(\boldsymbol{x}) = \sum_{i=1}^{T} \alpha_i y_i \big(\phi(x_i)^\top \phi(x)\big)$. In general, the inner product between $\phi(x_i)$ and $\phi(x)$, $\phi(x_i)^\top \phi(x)$ can be replaced by the kernel function $k(x_i, x)$ to have the classifier

$$f(x) = \sum_{i=1}^{T} \alpha_i y_i k(x_i, x). \tag{7}$$

For the regression problem, we are interested in forecasting the stock prices instead of just predicting a downward or upward trend, our response y_i represent the real value. The optimization problem can be changed to

$$\text{maximize } \frac{1}{2} \sum_{i=1}^{T} (\alpha_i - \alpha_i^*)\big(y_i - f(x_i)\big) - \epsilon \sum_{i=1}^{T} (\alpha_i + \alpha_i^*) \tag{8}$$

subject to

$$\sum_{i=1}^{T} (\alpha_i - \alpha_i^*) = 0 \text{ and } 0 \leq \alpha_i \leq C, \tag{9}$$

Where

$$f(x) = \sum_{i=1}^{T} (\alpha_i - \alpha_i^*) k(x_i, x) + b. \tag{10}$$

The absolute errors smaller than ϵ are disregarded. $C > 0$ is a regularization constant determining a trade-off between non-linearity of f and the amount up to which deviations larger than ϵ can be considered. The most popular choices of the kernel functions are the radial basis functions and the polynomial kernel (e.g., [9]).

3.2 Long Short-Term Memory (LSTM)

LSTM is a variation of the recurrent neural network, which avoids long term dependence problem and makes it suitable for predicting financial time series, including stock prices. As pointed by [10], the key feature is the state of these memory cells in the hidden layer, and these states are updated with the input through the gate structure, as shown in Figure 2. Each of the memory cell contains three layers of sigmoid transformation and one layer of tanh transformation.

At period t, the memory cell takes the output of the period $t-1$, h_{t-1}, and the external input or covariates x_t at period t as inputs and uses a sigmoid transformation

$$f_t = \sigma\big(W_f . [h_{t-1}, x_t] + b_f\big), \tag{11}$$

where W_f and b_f are the matrices of weights and the bias vector, respectively, of the forgotten gate. This will result in a value between 0 and 1, where 0 indicates discarding all information and 1 presents keeping all information from the previous period.

The input gate then determines the state of the cell which needs to be updated by the sigmoid function

$$I_t = \sigma\,(W_t.[h_{t-1}, x_t] + b_t). \tag{12}$$

It also updates the information that needs to be updated to the cell at period t, using a tanh function

$$\hat{C}_t = \tanh(W_c.[h_{t-1}, x_t] + b_c). \tag{13}$$

Finally, the state of the memory cell is updated using

$$C_t = f_t * C_{t-1} + I_t * \hat{C}_t. \tag{14}$$

The output information is determined by using a sigmoid layer and then processed as a tanh function as

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{15}$$

and the final output from the cell

$$h_t = O_t * \tanh(C_t). \tag{16}$$

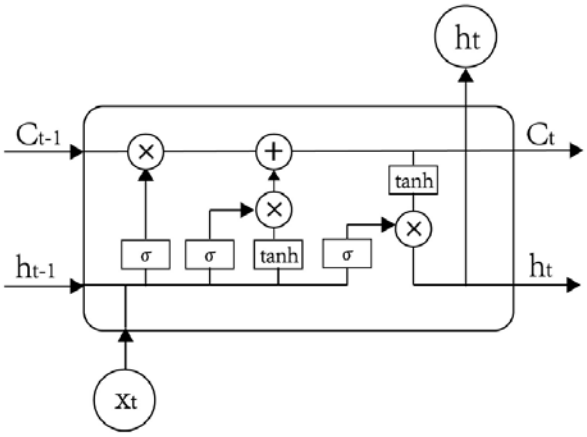

Figure 2. The memory cell structure of an LSTM model [11]

3.3 Convolutional Neural Networks (CNN)

Neural Network is one of the most popular machine learning algorithms. There are different types of Neural Networks methods such as CNN (Convolutional Neural Networks), Feedforward Neural Network (FNN), and RNN (Recurrent Neural Networks). Traditionally, the most application of CNN is in the image processing. We can utilize the CNN method for prediction of the stock price. Several early literature, such as [12], [13], and [14], has applied the CNN method for stock market prediction, whose approaches were to take a one-dimensional input for making predictions only based on the history of closing prices while ignoring other possible sources of information like technical indicators.

The CNN contains at least three layers. They include convolution layers, pooling layers (e.g., max pooling), and fully connected (FC) layers. The convolutional layer performs a series of convolutional operations on its inputs to facilitate pattern recognition, and each input is convoluted with kernel or filter. In convolution layers, there are three features that we should determine. They are the length and number of kernels, pooling stride, and padding, respectively.

The pooling layers have the responsibility to divide the dimension of the input, which is the most relevant information preserved. To utilize its down sampling function, a pooling window and a pooling stride should be configured. The fully connected layers (FC) combine each neuron to accurately and efficiently classify each input.

4. Notation and algorithm

This section describes the notation and algorithm utilized in this study. A computer program has been developed in Python for data analysis, pre-processing of the dataset, and implementation the machine learning algorithms. Following steps provide an overview of how the methods are implemented.

Step 1: Four stocks of Apple, Mastercard, Ford, and ExxonMobil are selected. Stock prices for January 1st, 2002 until March 11th, 2020, are downloaded (except for Mastercard data which is available from 2006).

Step 2: Data processing is performed on the dataset to study the distribution of prices and daily log-returns, the correlations between different stocks, and the autocorrelations in historical prices.

Step 3: LSTM is applied to the Apple dataset considering a lookback window of 180 days on adjusted close prices as the response variable. Figure 3 displays the architecture developed as a sequential model using LSTM and Dense layers. This would create a 180-feature dataset. The model is then trained on 75% of each dataset with validation split and batch size being 0.25 and 256, respectively. ADAM, a method for stochastic optimization, is used as the optimizer with a learning rate of 0.001. Training loss and validation loss are monitored, and it is observed that convergence is achieved after 100 epochs.

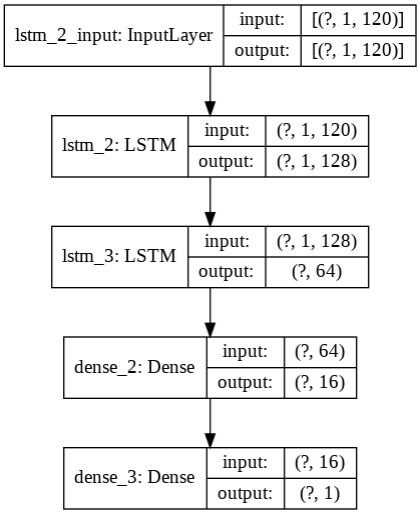

Figure 3. Developed architecture as a sequential model using LSTM and Dense layers

Step 4: Steps 2 and 3 are carried out for the other three stocks.

Step 5: Another model on the entire Apple dataset is trained using LSTM and is then used to predict the different three stock prices. This is to investigate whether it is possible to use a model trained on one stock to make forecasts on another one.

Step 6: The same approach in Steps 2, 3, 4, and 5 is repeated for machine learning models using combinations of CNN and LSTM. Figure 4 displays that architecture developed as a sequential model using a convolutional layer, dense layers, CNN, and LSTM. The inputs of the model are a 120-day lookback window of adjusted close prices. This would create a 120-feature dataset. The model is then trained on 75% of each dataset with kernel size of 8, strides equal to 1, and the activation function being ReLU. ADAM is used as the optimizer with the learning rate of 0.001. Training and validation losses are monitored, and it is observed that convergence is achieved after 100 epochs.

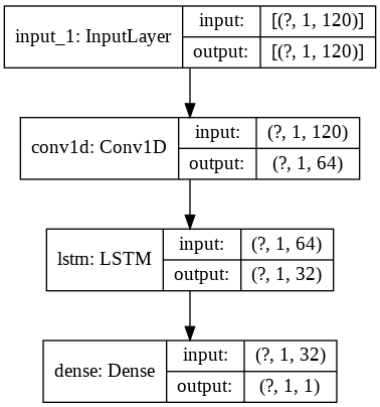

Figure 4. Developed architecture as a sequential model using a convolutional layer, dense layers, CNN, and LSTM

Step 7: An SVR method is applied to the Apple dataset with a lookback window of 30 days. Here, all daily stock variables are considered as inputs of the model (i.e., open price, close price, daily high, daily low, adjusted close price, and volume of trades) and therefore create 180 total features.

Step 8: Model is trained on 75% of the dataset, and 1-day prediction is performed.

Step 9: Grid search method is used to find the optimal parameters of the model.
Step 10: Steps 7, 8, and 9 are carried out for the other three stocks.
Step 11: Another model on the entire Apple dataset is trained using SVR and is then used to predict the other three stock prices. This is to investigate whether it is possible to use an SVR model trained on one stock to make forecasts on another one.

5. Data analysis

5.1 Method 1: LSTM

5.1.1 Stock price prediction using LSTM

Convergence rate and one-day forecasts of the adjusted close price as the response variable, displayed in Figure 5 and Figure 6, is presented for AAPL. Similarly, the LSTM model is trained on MAST, FORD, and EXON datasets, and predictions are displayed in Figures B1-B6 of Appendix B. Mean Absolute Percentage Error (MAPE) for each case is calculated to evaluate the performance.

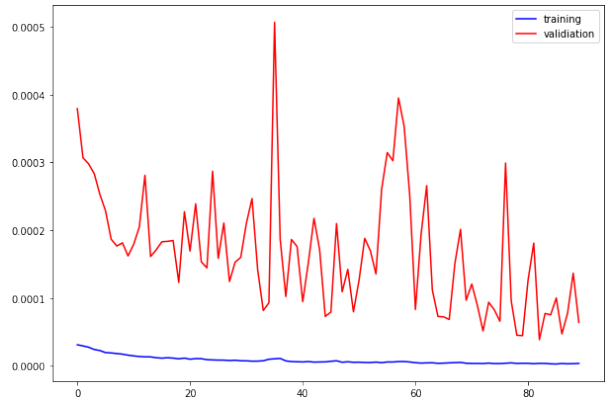

Figure 5. Convergence rate of LSTM for AAPL dataset

Figure 6. Stock price predictions using LSTM for AAPL

5.1.2 One single LSTM model for all datasets

As explained earlier, we would like to assess the accuracy of a model trained on one stock dataset when it is used to forecast other stock prices. This is to investigate whether it is possible to use a model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using LSTM and is then used to predict the other three stock prices. As it is observed in Figures 7, 8, and 9 displayed below, a model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

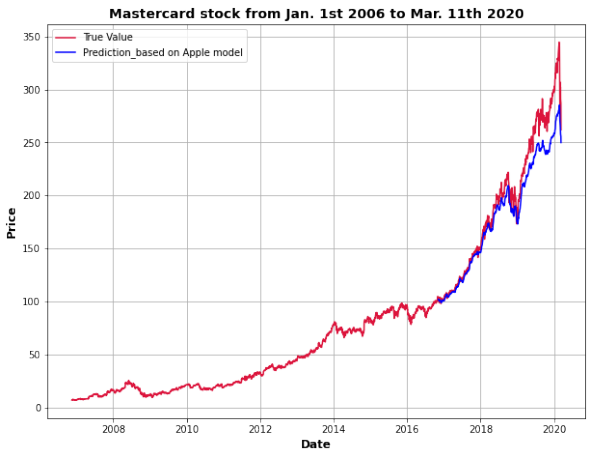

Figure 7. MAST stock price predictions using LSTM trained on AAPL

Figure 8. FORD stock price predictions using LSTM trained on AAPL

Figure 9. EXON stock price predictions using LSTM trained on AAPL

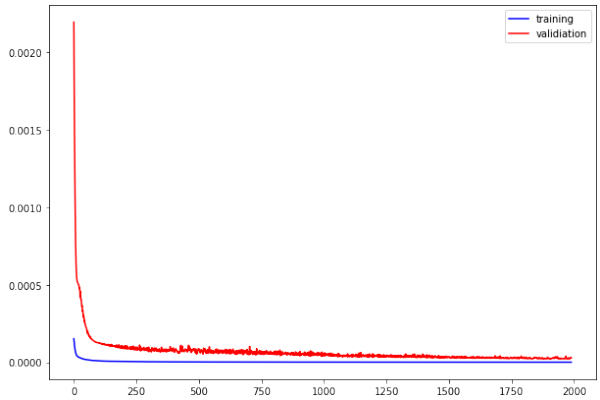

Figure 10. Convergence rate of CNN-LSTM for AAPL dataset

5.2 Method 2: CNN-LSTM

5.2.1 Stock price prediction using CNN-LSTM

Convergence rate and one-day forecast of the adjusted close price as the response variable, displayed in Figure 10 and Figure 11, are presented for AAPL. Similarly, the CNN-LSTM model is trained on MAST, FORD, and EXON datasets, and predictions are presented in Figures B7-B12 of Appendix B. MAPE for each case was calculated to evaluate the performance.

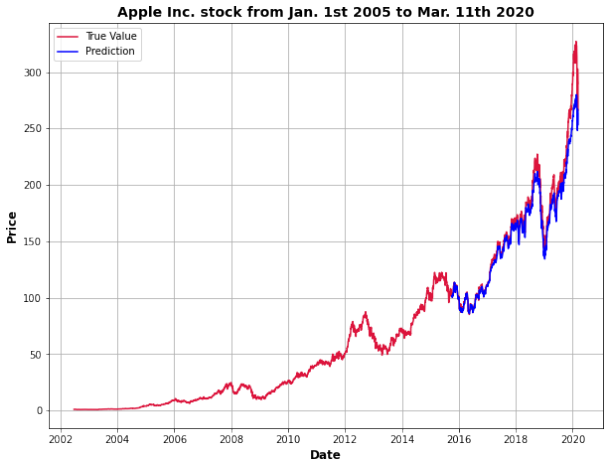

Figure 11. Stock price predictions using CNN-LSTM for AAPL

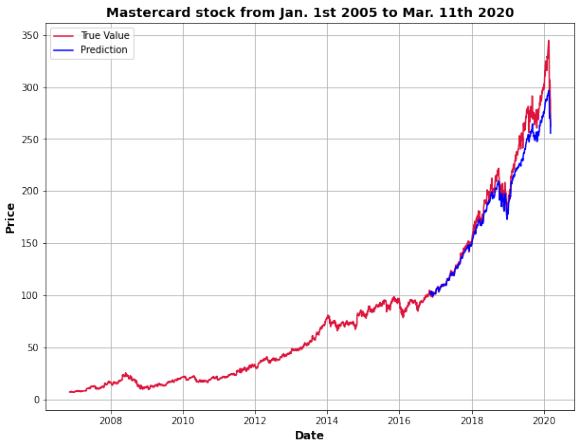

Figure 12. MAST stock price predictions using CNN-LSTM trained on AAPL

5.2.2 One single CNN-LSTM model for all datasets

Similar to the previous subsection, we would like to assess the accuracy of a model trained on one stock dataset when it is used to forecast other stock prices. This is to investigate whether it is possible to use a CNN model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using CNN and is then used to predict the other three stock prices. As it is observed in Figures 12, 13, and 14 presented below, a model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

5.3 Method 3: SVR

An SVR method is applied to each stock dataset with a lookback window of 30 days. All daily stock variables are considered as inputs of the model (i.e., open price, close price, daily high, daily low, adjusted close price and volume of trades) and therefore create 180 total features. Analysis results are displayed in Figures 15, 16, 17, and 18. The model is trained on 75% of each dataset, with the kernel function being set to RBF. Grid Search is then conducted to determine the best combination of the following hyperparameters: C: 500, 1000, 2000, Epsilon: 0.001, 0.1, 0.5, Gamma: 0.001, 0.1, 0.5.

It turns out that for all datasets, C = 2000, Epsilon = 0.001, and Gamma = 0.001 is the best combination for this SVR model. The following figures present forecasts using the SVR model. MAPE for each case was calculated to evaluate the performance.

Figure 13. FORD stock price predictions using CNN-LSTM trained on AAPL

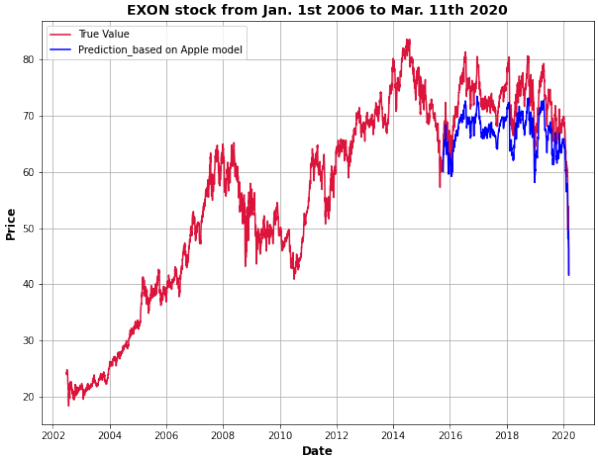

Figure 14. EXON stock price predictions using CNN-LSTM trained on AAPL

Figure 15. Stock price predictions using SVR for AAPL

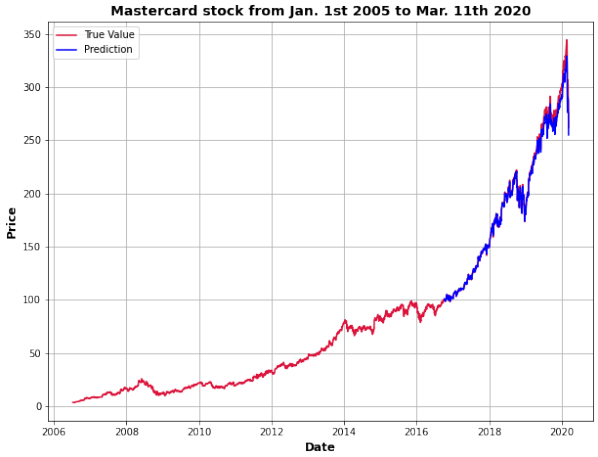

Figure16. Stock price predictions using SVR for MAST

Figure 17. Stock price predictions using SVR for FORD

Figure 18. Stock price predictions using SVR for EXON

Similar to the previous section, we would like to assess the accuracy of the SVR model trained on one stock dataset when it is used to forecast other stock prices. Analysis results are displayed in Figures 19, 20, and 21. This is to investigate whether it is possible to use an SVR model trained on one stock to make forecasts on another stock. To do so, a model on the entire Apple dataset is trained using SVR and is then used to predict the other three stock prices. As it is observed in the figures presented below, an SVR model trained on the Apple dataset is capable of making accurate forecasts about other stock prices. MAPE for each case was calculated to evaluate the performance.

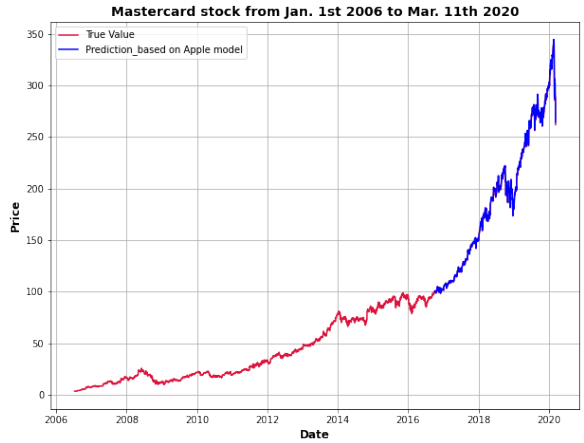

Figure 19. MAST stock price predictions using SVR trained on AAPL

Figure 20. FORD stock price predictions using SVR trained on AAPL

Figure 21. EXON stock price predictions using SVR trained on AAPL

6. Discussion

From the figures displayed in the Section 5, we can see that predictions are fairly accurate when the market is in a stable state (2015 – 2017 period), but as the volatility increases as in January 2020 forward, although the models still capture the general pattern, misalignments are observed in the forecast results and actual prices.

Table 2 summarizes the accuracies of different models in terms of the mean absolute percentage error (MAPE). This table proves that combining CNN and LSTM increases the accuracy compared to LSTM. Also, according to these results, the SVR model is able to predict stock prices with the highest accuracy compared to the other two methods. The reason is thought to be due to a more comprehensive set of features that were used in the SVR model.

It is also shown in this paper that we can train machine learning models on one stock dataset and use it to forecast different stock prices with a mild sacrifice in terms of accuracy. In the case of CNN model for predicting FORD stock price, for example, Table 2 shows that a model trained on Ford dataset will perform with MAPE = 2.66 while a model trained on AAPL used to forecast FORD prices has MAPE of 8.68. In Section 2, we discuss that all these four stocks have positive correlations with each other, and this could be a reason why a model trained

on one of them can predict the others. In the future research, we can further extend those machine learning methods with nonparametric estimations (e.g., [9] and [15]).

Table 2. Summary of the performance of different methods

Model	Trained on Data from	Mean Absolute Percentage Error MAPE of each Machine Learning's prediction on			
		AAPL	MAST	FORD	EXON
LSTM	AAPL	13.75	6.67	8.68	9.71
LSTM	MAST	-	19.64	-	-
LSTM	FORD	-	-	2.66	-
LSTM	EXON	-	-	-	1.34
CNN	AAPL	2.18	4.78	6.77	7.39
CNN	MAST	-	4.17	-	-
CNN	FORD	-	-	0.55	-
CNN	EXON	-	-	-	0.77
SVR	AAPL	0.67	0.11	0.51	0.21
SVR	MAST	-	0.86	-	-
SVR	FORD	-	-	0.097	-
SVR	EXON	-	-	-	0.085

7. References

[1] Box GEP, Jenkins GM, Reinsel GC, Ljung GM. Time series analysis: forecasting and control. New York: Wiley; 2015.

[2] Chen LP, Tian WT, Yeh HC. A study of the birth rate in Taiwan. Journal of Data Analysis. 2015; 10(6): 141-190. DOI: 10.6338/JDA.201512_10(6).0006

[3] Huang C, Huang L, Han T. Financial time series forecasting based on wavelet kernel support vector machine. In: 8th International Conference on Natural Computation. IEEE. 2012. http://doi.org/10.1109/ICNC.2012.6234569

[4] Wu J, Lu C. Computational intelligence approaches for stock price forecasting. In: International Symposium on Computer, Consumer and Control (IS3C). IEEE. 2012; 52 – 55.

[5] Kao L, Chiu C, Lu C, Chang C. A hybrid approach by integrating wavelet-based feature extraction with MARS and SVR for stock index forecasting. Decision Support Systems. 2013; 54(3): 1228 – 1244.

[6] Chen LP. Multiclassification to gene expression data with some complex features. Biostatistics and Biometrics Open Access Journal. 2018; 9(1): 555751

[7] Chen LP. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar: Foundations of machine learning, second edition. Statistical Papers. 2019; 60(5): 1793–1795.

[8] Chen LP. Model-based clustering and classification for data science: with application in R by Harles Bouveyron, Gilles Celeus, T. Bredan Murphy and Adrian E. Raftery. Biometrical Journal. 2020. (In press). DOI: 10.1002/bimj.201900390

[9] Hastie T, Tibshirani R, Friedman J. The elements of statistical learning: data mining, inference, and prediction. New York: Springer; 2008.

[10] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997; 9(8): 1735–1780.

[11] https://ai.stackexchange.com/questions/6961/structure-of-lstm-rnns

[12] Gunduz H, Yaslan Y, Cataltepe Z. Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations. Knowledge-Based Systems. 2017; 137(1): 138–148.

[13] Chong E, Han C, Park FC. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications. 2017; 83(15): 187–205.

[14] Chen K, Zhou Y, Dai F. A LSTM-based method for stock returns prediction: A case study of China stock market. In: International Conference on Big Data (Big Data). IEEE. 2015; 2823–2824.

[15] Chen LP. Investigating effects of bandwidth selection in local polynomial regression model with applications. Model Assisted Statistics and Application. 2019; 14(1): 31-45.

Appendix A: Figures referring to Section 2

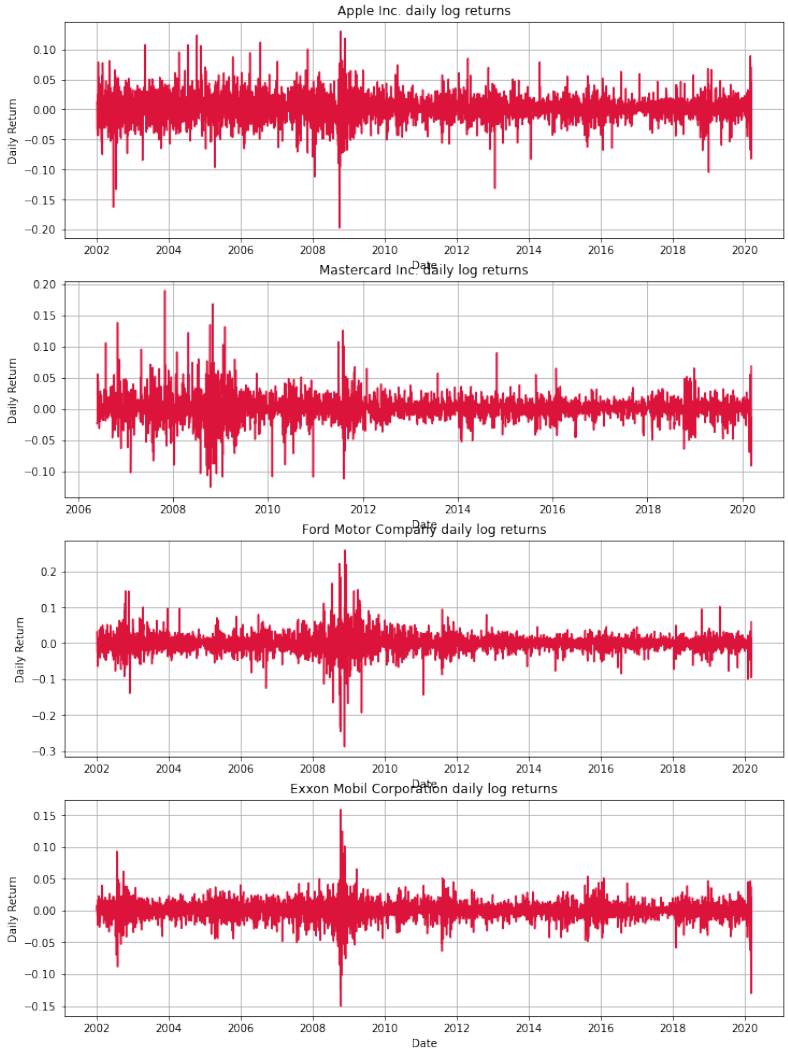

Figure A1. Daily log-returns for Apple, MasterCard, Ford, and ExxonMobil

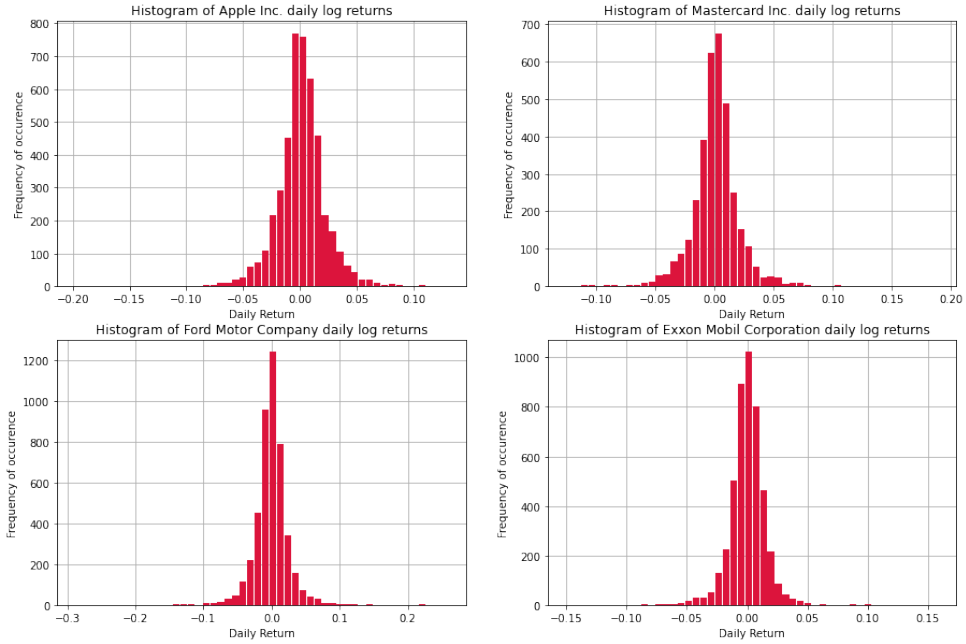

Figure A2. Histograms of daily log-returns for Apple, MasterCard, Ford, and ExxonMobil

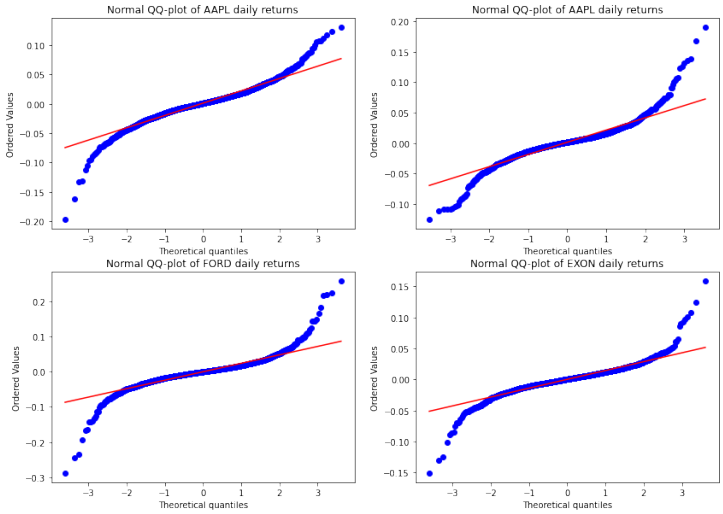

Figure A3. Comparison of daily log returns distribution to Normal distribution

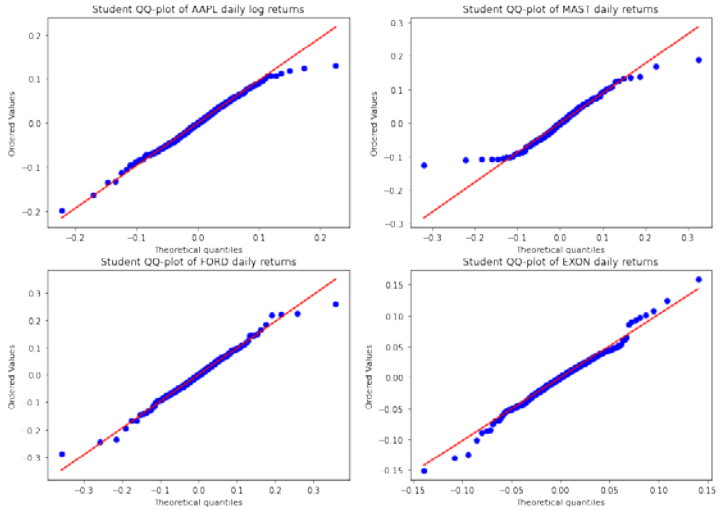

Figure A4. Comparison of daily log returns distribution to Student's t-distribution

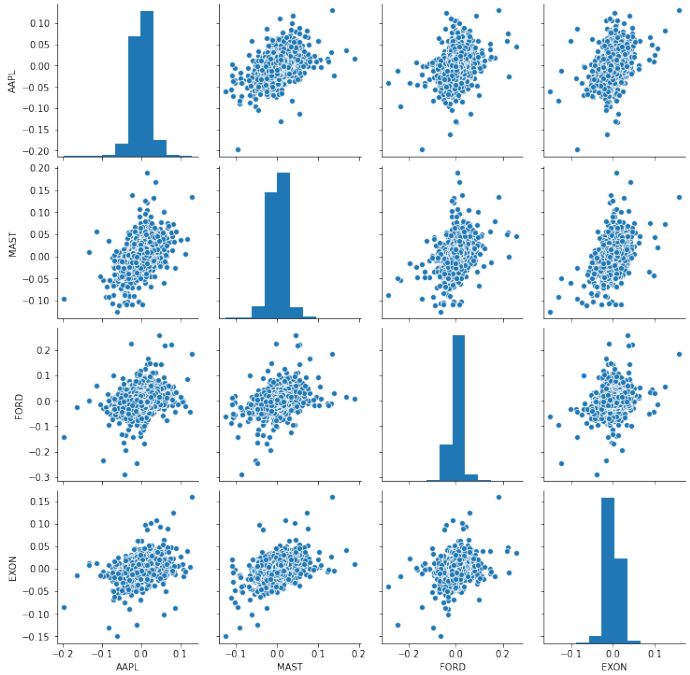

Figure A5. Pair-plots of log-returns for Apple, MasterCard, Ford, and ExxonMobil

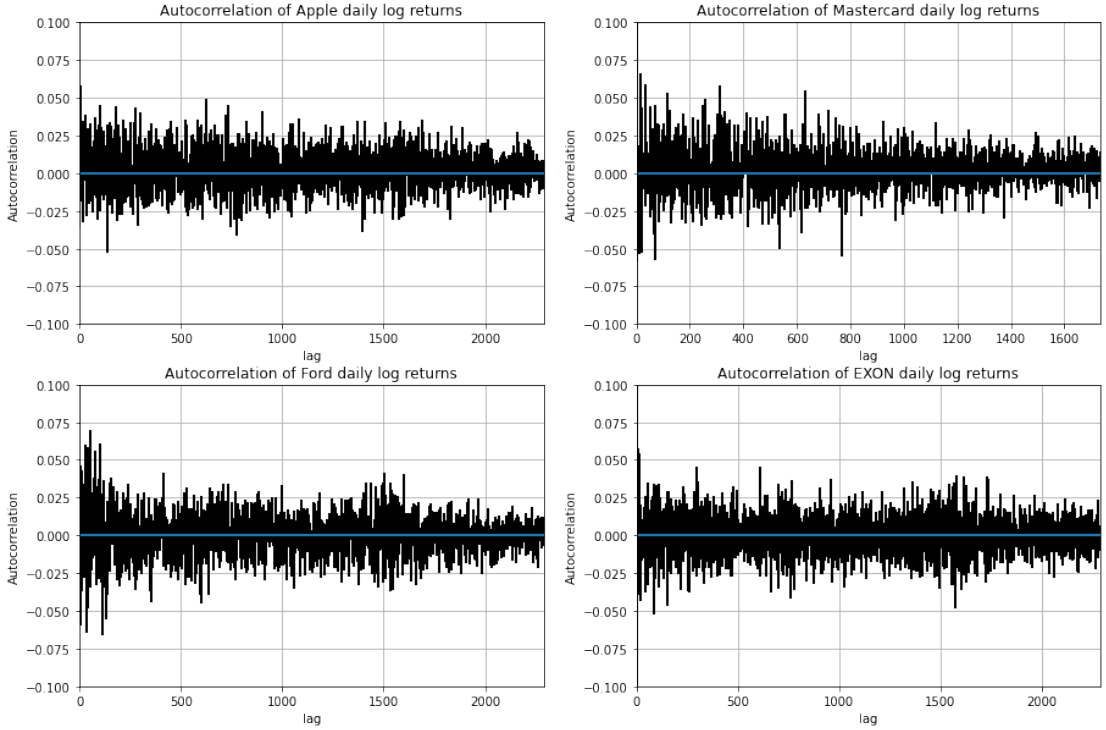

Figure A6. Autocorrelation of log-returns for Apple, MasterCard, Ford, and ExxonMobil

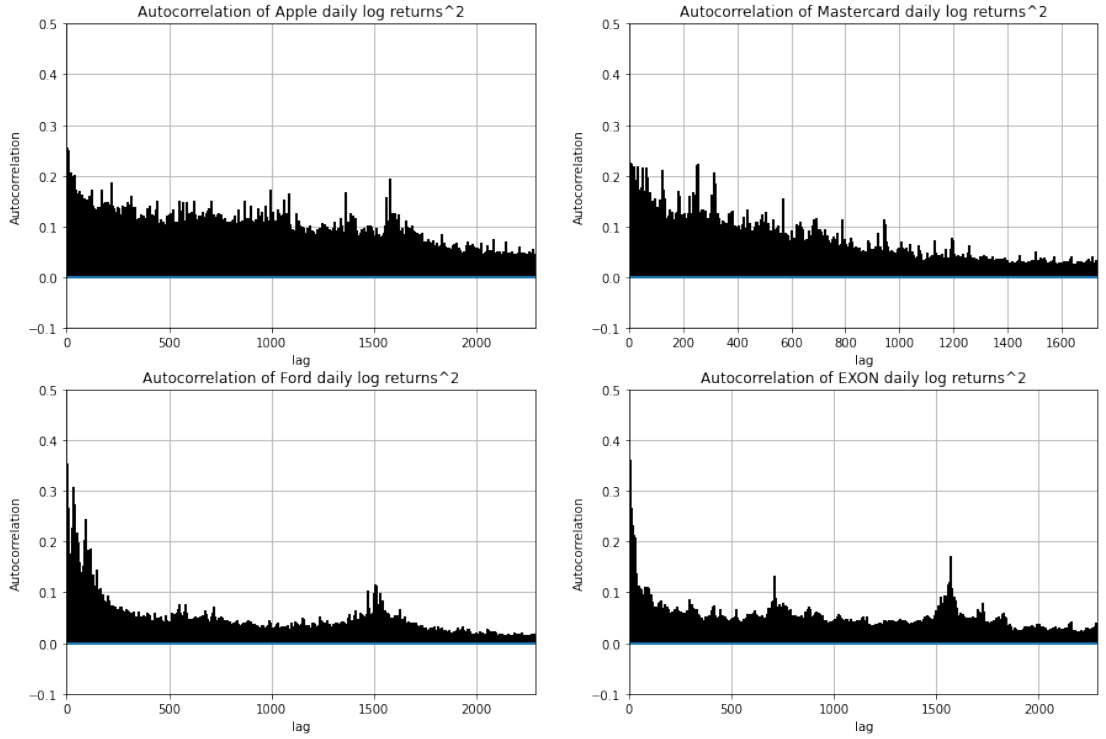

Figure A7. Autocorrelation of squared of log-returns for Apple, MasterCard, Ford, and ExxonMobil

Appendix B. Figures referring to Section 5

Figures below show the convergence rates and prediction results of different models trained to forecast MAST, FORD, and EXON prices.

B-1. MAST stock price prediction using LSTM

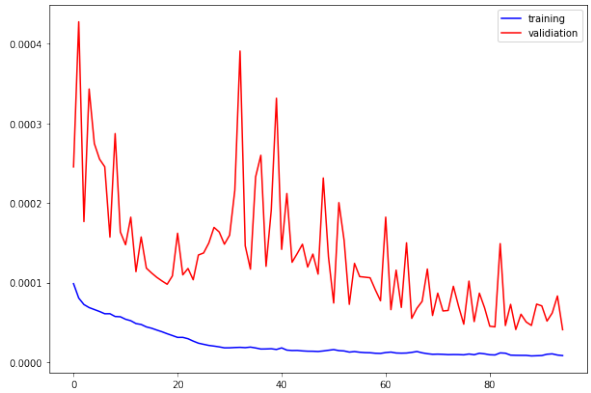

Figure B1. Convergence rate of LSTM for MAST dataset

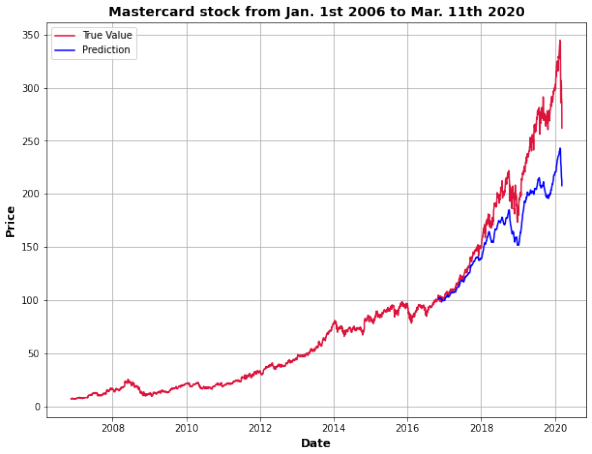

Figure B2. Stock price predictions using LSTM for MAST

B-2. FORD stock price prediction using LSTM

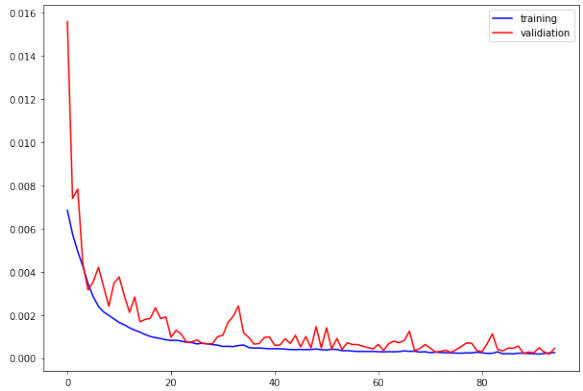

Figure B3. Convergence rate of LSTM for FORD dataset

Figure B4. Stock price predictions using LSTM for FORD

B-3. EXON stock price prediction using LSTM

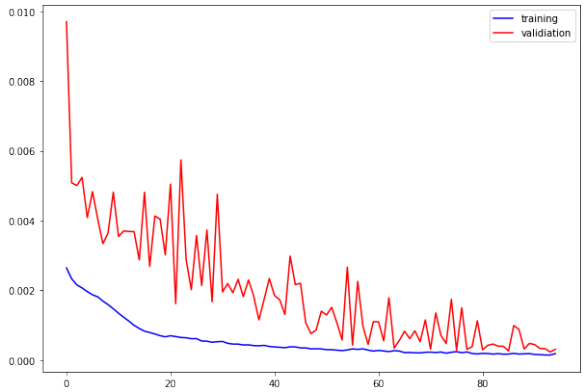

Figure B5. Convergence rate of LSTM for EXON dataset

Figure B6. Stock price predictions using LSTM for EXON

B-4. MAST stock price prediction using CNN-LSTM

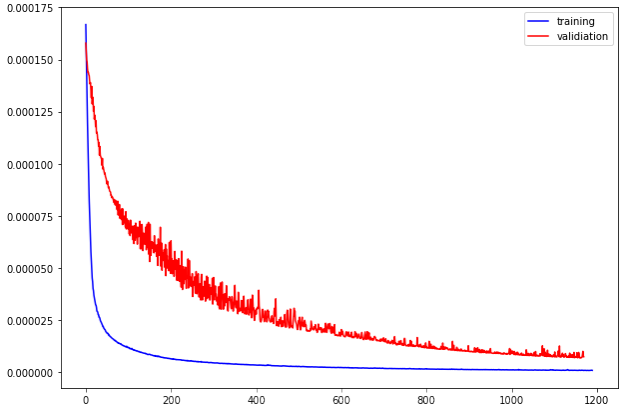

Figure B7. Convergence rate of CNN-LSTM for MAST dataset

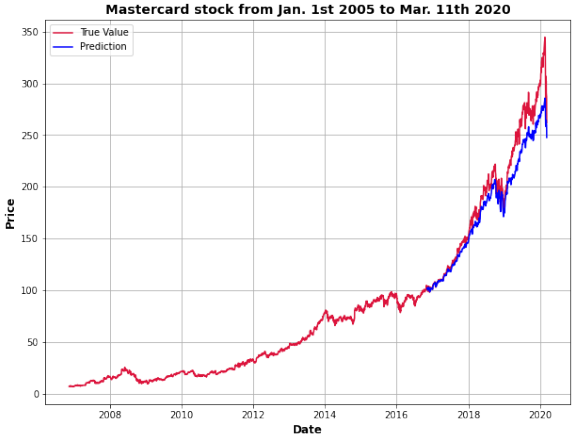

Figure B8. Stock price predictions using CNN-LSTM for MAST

B-5. FORD stock price prediction using CNN-LSTM

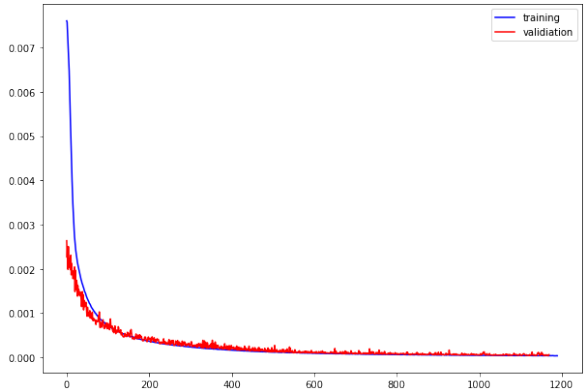

Figure B9. Convergence rate of CNN-LSTM for FORD dataset

Figure B10. Stock price predictions using CNN-LSTM for FORD

B-6. EXON stock price prediction using CNN-LSTM

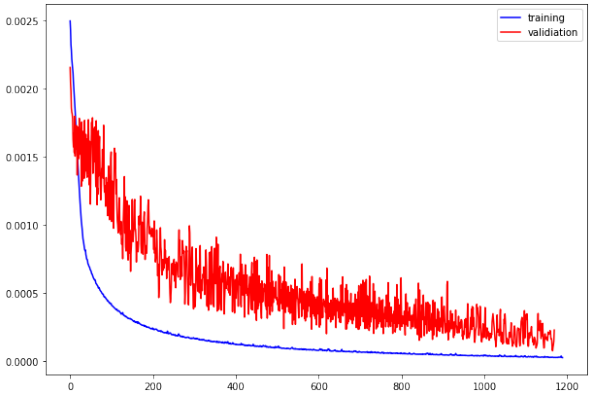

Figure B11. Convergence rate of CNN-LSTM for EXON dataset

Figure B12. Stock price predictions using CNN-LSTM