EECE 571T Project Update: Stock Price Forecasting

Zhaosheng Li, Menghong Huang

Proposed Work

In this project, we propose to apply three types of machine learning (ML) models to make price prediction of the *GameStop*'s stock. To elaborate on that, those ML models are support vector regression (SVR), back-propagation neural network (BPNN), and recurrent neural network (RNN). And in the domain of RNN, we implement three kinds of architectures, which include the basic RNN, long short-term memory (LSTM), and gated recurrent unit (GRU). As for the content of stock price forecasting, we only predict the close price of the *GameStop*'s stock. We conduct two short-term predictions, which are next-day forecast and 30-day forecast. The next-day forecast is based on the data of stock prices in the latest four days in a row, whereas the 30-day forecast is about using the daily stock price in the latest thirty consecutive days to predict the stock price after thirty days.

After training those aforementioned ML models, we feed the test data set into those models and compare their performance. The comparison is based on three metrics, which are root mean square error (RMSE), coefficient of determination (R^2), and the accuracy of trend prediction. To improve the data visualization, we plot the actual stock price and the predicted price for each model or architecture.

Current Progress

At the very beginning of this project, we just decide to conduct a simple daily prediction of the stock price. The simple daily prediction is using the date as feature and the close price as label. However, after implementing the SVR model to do that forecasting, we find the R^2 of the trained model is negative, which means the model fits the data even worse than a horizontal line. Reflecting on that result, we acknowledge that it is difficult for the ML models to find the relationship between the date and the close price. Plus, doing that kind of prediction is not practical in the stock market. This turns our prediction into the next-day forecast and 30-day forecast.

We apply the SVR model through `sklearn` module to do those two forecasts. We choose the radial basis function as the kernel function. For the next-day forecast, we use the close prices in the latest four successive days as input. While for the 30-day forecast, we use the close prices in the latest thirty days in a row as input data. To further visualize the performance of the SVR model, we also apply the least squares linear regression model from `sklearn` to do the same prediction. After that, we measure the performance of the SVR model and linear regression model separately. We see the performance of these two models is comparable.

Using the `keras` module, we build a BPNN model that has three fully connected layers to do the price forecasting. As for the next-day prediction, all the columns of a single entry are treated as input to the model, except the date. In order to help the BPNN capture the pattern of the sequential data, we also add the close prices of the previous three entries as input. While for the 30-day forecast, the close prices of the previous twenty-nine entries are added instead. We find the BPNN model performs as expected when doing the next-day forecast. However, as we extend the price prediction to the 30-day forecast, the model does not fit the test data as anticipated. That drives us to bring the RNN model to the table.

We build the basic RNN, LSTM, and GRU through `keras` module to do the aforesaid two forecasts. In the realm of RNN, all the columns in the data set get involved except the date. That being said, all the RNN models have the many-to-one structure. Since RNN is prone to vanishing gradient and its activation function is sigmoid function or tanh function, we normalize all the data to the range between zero and one before feeding them into the three architectures. In the structure of the basic RNN, we only build one hidden layer that contains the RNN units. In the testing stage, the pattern of price changing is decently captured, which means the temporal relationship is nicely learned. As for the LSTM and GRU architectures, we build three stacked LSTM/GRU layers for the next-day prediction, whereas we designed four stacked LSTM/GRU layers for the 30-day forecast. In order to avoid the overfitting in the training process, we also add a dropout layer with dropping rate of 0.2 after each LSTM/GRU layer.

Figures and Table

This section demonstrates the next-day forecasting results through each trained ML model or architecture.

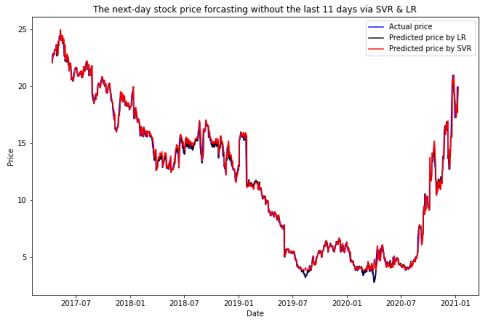

Figure 1: Next-day forecast via linear regression and SVR

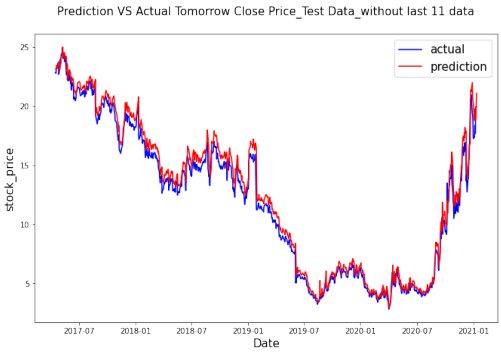

Figure 2: Next-day forecast via BPNN

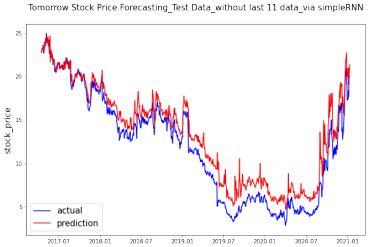

Figure 3: Next-day forecast via basic RNN

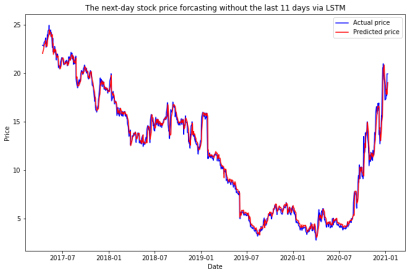

Figure 4: Next-day forecast via LSTM

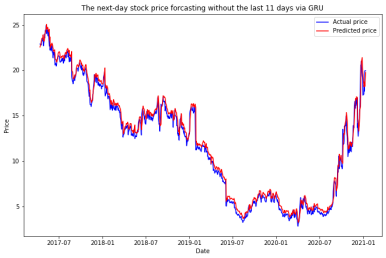

Figure 5: Next-day forecast via GRU

Model / Architecture	RMSE	R^2	Accuracy of trend prediction
Linear Regression	0.5018	0.9933	50.10%
SVR	0.5275	0.9926	48.64%
BPNN	0.8191	0.9820	49.92%
Simple RNN	1.8214	0.9110	49.63%
LSTM	0.5827	0.9909	49.58%
GRU	0.6628	0.9882	49.79%

Table 1: Next-day forecasting performance