**Task Report Cos30018 Option B**

**B.6: Machine Processing 3**

**Name: Le Bao Nguyen**

**Student Id: 104169837**

1. Importing libraries (From the task B.2):



Figure 1: Importing libraries to run the code.

* The script imports previous libraries and new libraries:
* "statsmodels.tsa.arima.model.ARIMA": Provides the implementation of the ARIMA (AutoRegressive Integrated Moving Average) model for time series forecasting.
* "sklearn.metrics.mean\_squared\_error": Used to calculate the mean squared error between the predicted and actual values in order to assess the effectiveness of regression models.
* "warnings": A module to manage the warnings that are produced when the code runs.

1. Data loading and processing (From the task B.2):

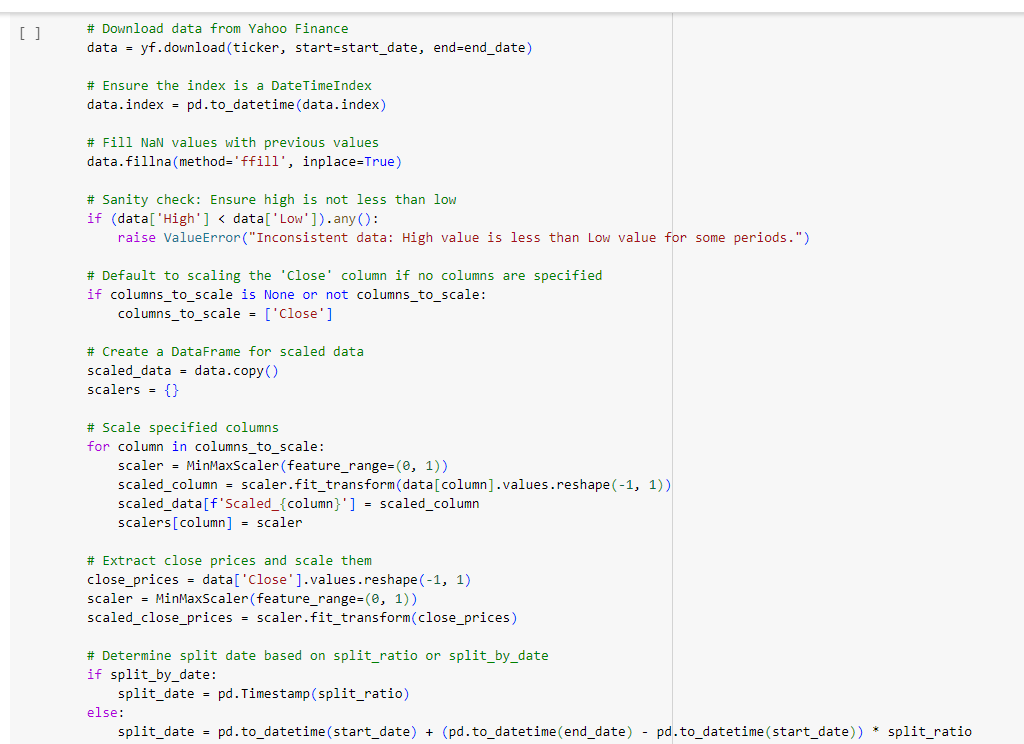


Figure 2: Loading and processing data (1).

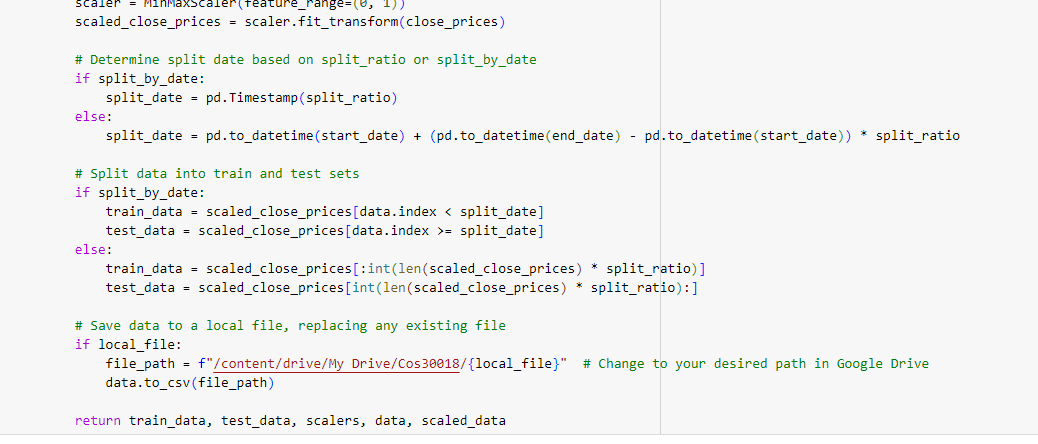


Figure 3: Loading and processing data (2).

* We still use the same data loading and processing just like B.2.

1. Displaying data in a custom table (From the task B.2):

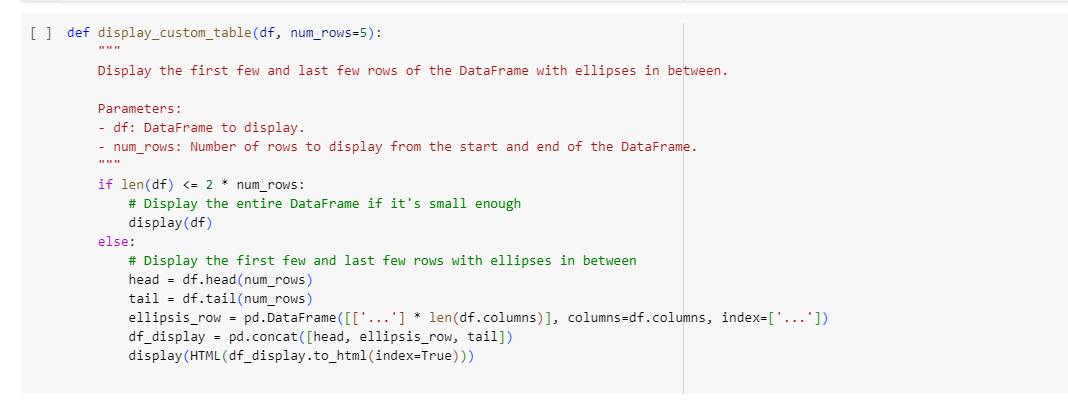


Figure 4: Displaying the data from csv file.

* We still use the same displaying data function just like B.2.

1. Model Creation (From the task B.4):



Figure 5: Code to create the model.

* We still use the same displaying data function just like B.4.

1. Experimentation with Different Configurations (From the task B.4):



Figure 6: Code to experiment with model (1).



Figure 7: Code to experiment with model (2).

* We still use the same displaying data function just like B.4.

1. Arima Predictions:

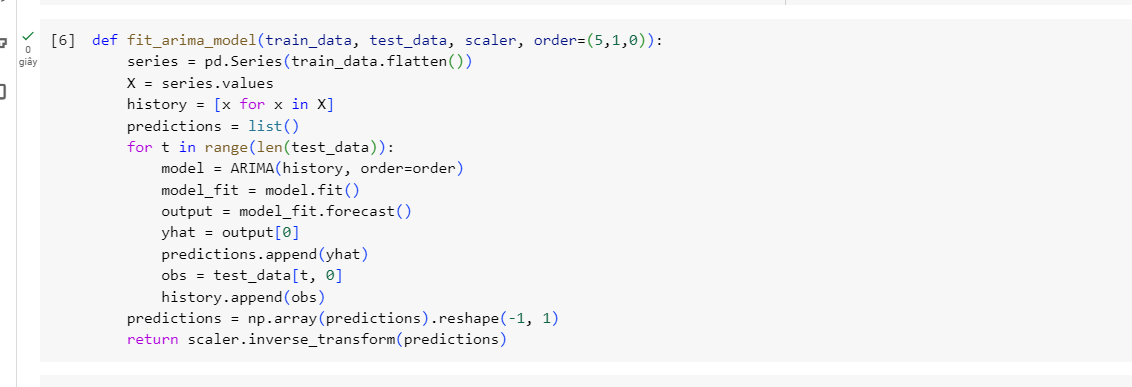


Figure 8: Code to fit arima model.

* The ARIMA (AutoRegressive Integrated Moving Average) model is a popular time series forecasting technique that makes predictions about future points in a series using historical data.
* Order parameters:
* P: The number of lag observations included in the model (autoregressive part).
* D: The number of difference analyses (integrated part) performed on the raw observations.
* Q: The size of the moving average window (moving average part).
* Implementation:
* Convert the training data into a pandas series.
* Flatten the series values to make it a 1D array.
* Initialize the values from the training data into a list called "history".
* Make an empty list called "predictions" that contains the expected values.
* Fit the "history" data to an ARIMA model.
* Applying the fitted model, predict the next value.
* Add the predicted value (yhat) to the list of predictions.
* For the following iteration, add the real observation (obs) from the test data to "history".
* Restructure and convert the prediction list to a numpy array.
* To inversely transform the predictions back to the original scale, use the scaler.

1. Ensemble Predictions:

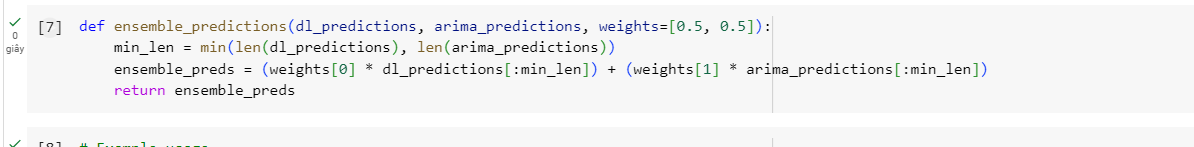


Figure 9: Code to create ensemble predictions.

* With ensemble methods, the final prediction is generated by combining the predictions of several models. The goal is to minimize each model's problems while improving its strengths. We integrated the ARIMA model with predictions from deep learning algorithms for our work.
* Why we use ensemble predictions:
* Deep learning models: These models are effective at identifying complex patterns and trends in data, but at specific volatile times, they may overfit or perform worse.
* ARIMA model: This model may not be as good at modeling non-linear patterns as it is at capturing linear dependencies and trends, which makes it difficult to overfitting.
* We hope to get more accurate and dependable forecasts by combining the two models in a way that balances each of their strengths and limitations.
* Implementation:
* Define the function using weights for averaging, ARIMA predictions, and deep learning prediction parameters.
* Determine the minimum length of predictions to ensure both arrays align correctly.
* Take a weighted average of the ARIMA and deep learning predictions to get the ensemble predictions. The weights parameter gives you more freedom in deciding how much weight to give each model's predictions.
* Return the combined predictions.

1. Main script run:



Figure 10: The script to run the code and the prediction (1).



Figure 11: The script to run the code and the prediction (2).

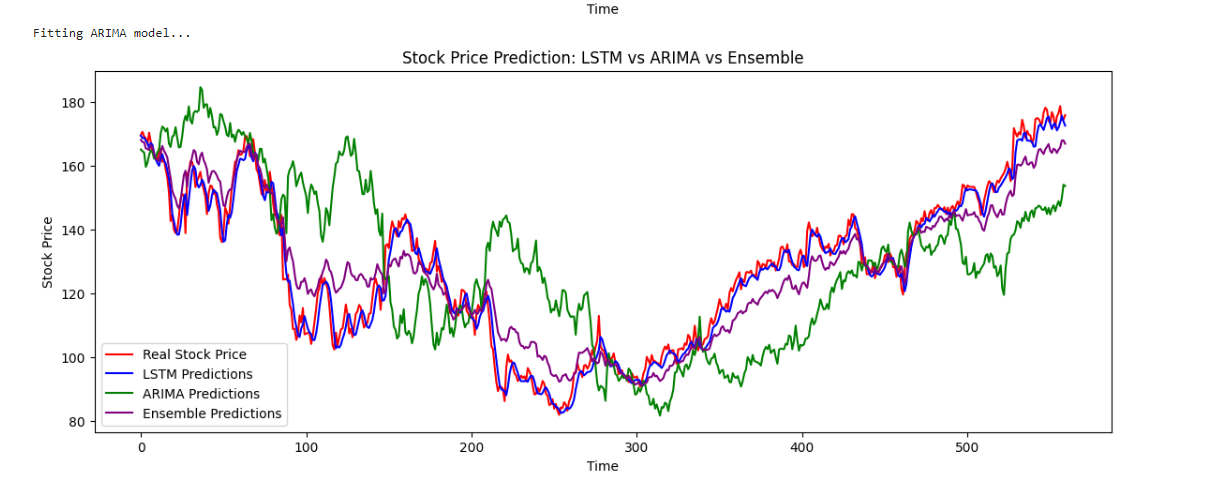


Figure 12: The arima and ensemble predictions (with LSTM).

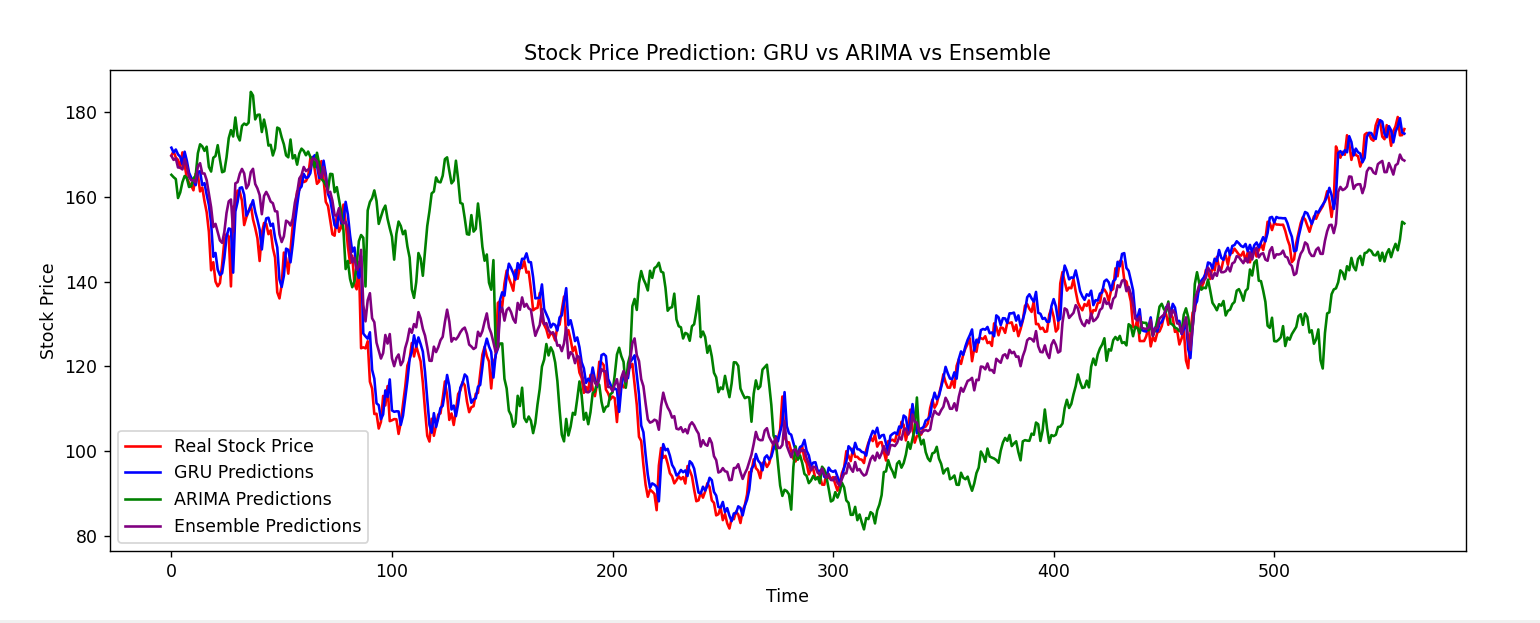


Figure 13: The arima and ensemble predictions (with GRU).

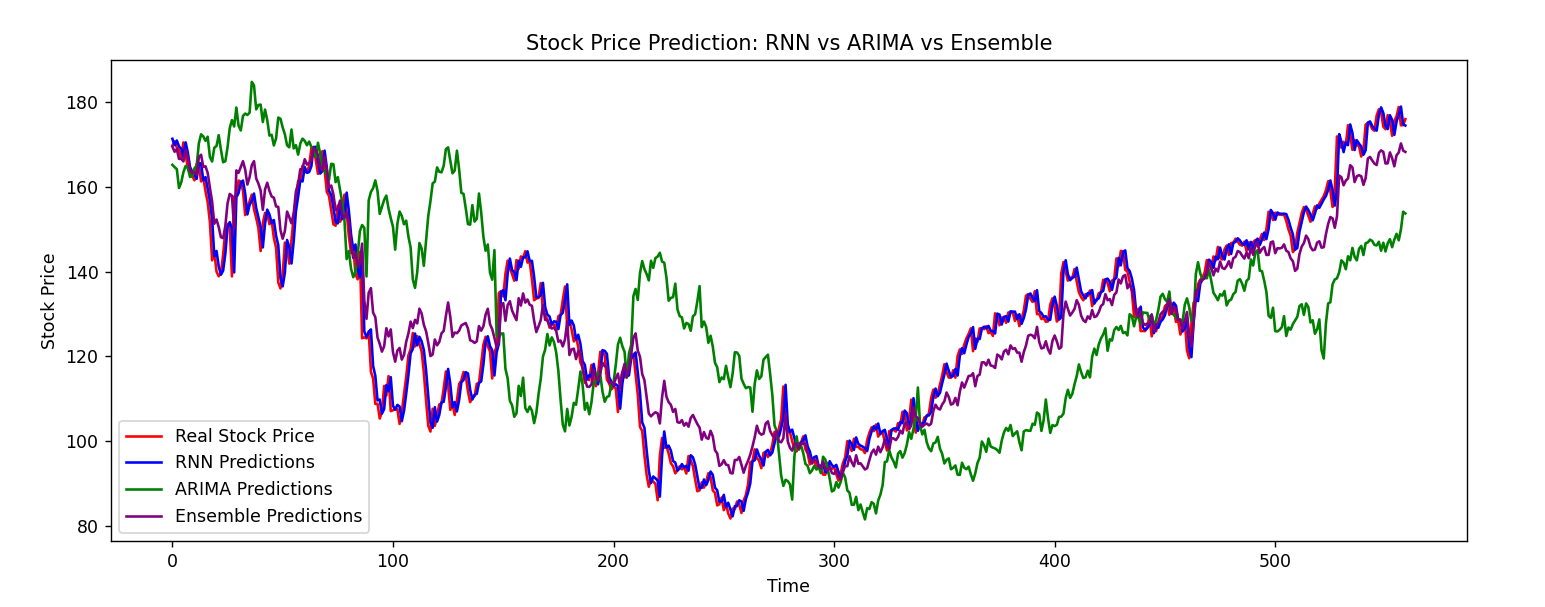


Figure 14: The arima and ensemble predictions (with RNN).

1. References:

RAZA, M. (2020). 1CAC 40 Stock Price Forecast with ARIMA & LSTM. <https://www.kaggle.com/code/razamh/1cac-40-stock-price-forecast-with-arima-lstm?fbclid=IwY2xjawEQoChleHRuA2FlbQIxMAABHURYRNO29-P9jXI1fUxeQb70JPGjKfPS9VpmFBDA3Rbw8tJqAUDe1sTiog_aem_deYZIQC-B71qaA0j4gVFyQ>