LSTM Stock Predictor

Due to the volatility of cryptocurrency speculation, investors will often try to incorporate sentiment from social media and news articles to help guide their trading strategies. One such indicator is the Crypto Fear and Greed Index (FNG) (https://alternative.me/crypto/fear-and-greed-index/) which attempts to use a variety of data sources to produce a daily FNG value for cryptocurrency. You have been asked to help build and evaluate deep learning models using both the FNG values and simple closing prices to determine if the FNG indicator provides a better signal for cryptocurrencies than the normal closing price data.

In this assignment, you will use deep learning recurrent neural networks to model bitcoin closing prices. One model will use the FNG indicators to predict the closing price while the second model will use a window of closing prices to predict the nth closing price.

You will need to:

1. Prepare the data for training and testing
2. Build and train custom LSTM RNNs
3. Evaluate the performance of each model

Files

Closing Prices Starter Notebook (Starter_Code/lstm_stock_predictor_closing.ipynb)

FNG Starter Notebook (Starter_Code/lstm_stock_predictor_fng.ipynb)

Instructions

Prepare the data for training and testing

Use the starter code as a guide to create a Jupyter Notebook for each RNN. The starter code contains a function to create the window of time for the data in each dataset.

For the Fear and Greed model, you will use the FNG values to try and predict the closing price. A function is provided in the notebook to help with this.

For the closing price model, you will use previous closing prices to try and predict the next closing price. A function is provided in the notebook to help with this.

Each model will need to use 70% of the data for training and 30% of the data for testing.

Apply a MinMaxScaler to the X and y values to scale the data for the model.

Finally, reshape the X_train and X_test values to fit the model's requirement of samples, time steps, and features. (*example:* X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1)))

Build and train custom LSTM RNNs

In each Jupyter Notebook, create the same custom LSTM RNN architecture. In one notebook, you will fit the data using the FNG values. In the second notebook, you will fit the data using only closing prices.

Use the same parameters and training steps for each model. This is necessary to compare each model accurately.

Evaluate the performance of each model

Finally, use the testing data to evaluate each model and compare the performance.

Use the above to answer the following:

> Which model has a lower loss?
>
> Which model tracks the actual values better over time?
>
> Which window size works best for the model?

Resources

[Keras Sequential Model Guide (https://keras.io/getting-started/sequential-model-guide/)](https://keras.io/getting-started/sequential-model-guide/)

[Illustrated Guide to LSTMs (https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21)](https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21)

[Stanford's RNN Cheatsheet (https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks)](https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks)

Hints and Considerations

Experiment with the model architecture and parameters to see which provides the best results, but be sure to use the same architecture and parameters when comparing each model.

For training, use at least 10 estimators for both models.

Submission

- Create Jupyter Notebooks for the homework and host the notebooks on GitHub.

- Include a Markdown that summarizes your homework and include this report in your GitHub repository.

- Submit the link to your GitHub project to Bootcamp Spot.

Requirements

Data Prep for Training and Testing (26 points)

To receive all points, your code must:

- Use the FNG values to predict future closing prices. (7 points)
- Use the past closing prices to predict future closing prices. (7 points)
- Apply the MinMaxScaler to the X and Y values to scale the data for the model. (6 points)
- Reshape X_train and X_test to fit the model requirements (samples, time steps, features). (6 points)

Build and train custom LSTM RNNS (20 points)

To receive all points, your code must:

- Create a notebook to fit the data using FNG Values. (10 points)
- Create a notebook to fit the data using closing prices. (10 points)

Evaluate Model Performance (24 points)

To receive all points, your code must:

- Determine which model had the lowest loss. (8 points)
- Determine which model tracks the actual values best over time. (8 points)
- Determine the appropriate Window Size for the model. (8 points)

Coding Conventions and Formatting (10 points)

To receive all points, your code must:

- Place imports at the beginning of the file, just after any module comments and docstrings and before module globals and constants. (3 points)
- Name functions and variables with lowercase characters and with words separated by underscores. (2 points)
- Follow Don't Repeat Yourself (DRY) principles by creating maintainable and reusable code. (3 points)
- Use concise logic and creative engineering where possible. (2 points)

Deployment and Submission (10 points)

To receive all points, you must:

- Submit a link to a GitHub repository that’s cloned to your local machine and contains your files. (5 points)
- Include appropriate commit messages in your files. (5 points)

Code Comments (10 points)

To receive all points, your code must:

- Be well commented with concise, relevant notes that other developers can understand. (10 points)
