Stock-Trend Prediction

About

Using XGDBoost And A custom neural Network to analyse the gradients of stocks.

I highly recommend creating a new environment for every project.This will create the env and activate it

```
python3 -m .venv venv
source venv/bin/activate
Copy
```

This will install all the needed deps

```
pip install -r requirements.txt
Copy
```

Usage

- `notebooks/model.ipynb` : Consists a diverse set of technical indicators that it evaluates off of, input can be any notebook ending with `usable.ipynb`.
- `notebooks/model2.ipynb` : Consists a less diverse set of techincal indicators however, gives a higher accuracy. Both these notebooks are based on xgboosts regression model
- `notebooks/nueral_net.ipynb` : A custom neural network written in pytorch, takes the longest to run (˜1min), however does provide the the most accuracy. Input can be any data directly from the NSE webitse or directly from yahoo finance.
- `docs/neural_network/` : consists of all the notebooks run on the neural network along with their accuracy and some test predictions
- `docs/regressor/`: consists of all the notebooks run on the xgboost regressor model along with their accuracy and predictions

Methodology

A total of 4 predictions were made with the neural network and 2 with the gradient boosting algorithm. The data was fed in the following manner.

- **Case 1:** Entire weekly data of 2020-2022, predicting the dataof 2023
- **Case 2:** Entire monthly data of 2020-2022, predicting the dataof 2023
- **Case 3:** Entire weekly data of 2020-2022, predicting the data of the first week of 2023. Then feeding in the data of the first week as well and trying to predict the next week and so on till August 26th
- **Case 4:** Entire monthly data of 2020-2022, predicting the data of January 2023. Then feeding in the data of the January as well and trying to predict the trend of February and so on till August 26th

Results

From the models created, we can come to the following conclusions

- The neural network performs much better than the gradient boosing algorithm. The gradient boosing algorithm was created using XGDBoost, the neural network was created from scratch using PyTorch.
- The runtime on the neural network for each excel sheet was ˜2min, the runtime on the gradient boosting algorithm was ˜1min.
- There are three cases mentioned above the in the *Methodoloy* section, the average accuracy using measures of central tendency will be given below.
- **1) XGDBoost**:

Case	Mean	Median	Mode	Mean Deviation	Standard Deviation
1	61%	59%	58.34%	1.2	1.3
2	60%	59.85%	60%	0.7	0.4
3	59%	59.43%	58.31%	1.0	0.8
4	58%	57.89%	58.12%	1.3	1.2

Variance: 0.1

- **2) PyTorch Neural Network**:

Case	Mean	Median	Mode	Mean Deviation	Standard Deviation
1	67%	66.34%	66%	1.1	0.5
2	65%	66%	65.78%	0.5	0.6
3	61%	60.15%	60%	0.7	0.6
3	61%	60.5%	60%	0.9	0.8

Variance: 0.23