Capstone Final Report

Predicting stock price trends using AIML

This capstone project was aimed at the exploration of the usage of classical Machine Learning algorithms and certain types of neural networks in predicting stock price trends two hours into the future.

Rustom Fracis

Sreekanth Sama

Siddhant Bhura

(GROUP-4)

Approach

At the outset, our objective was to try and find a method that would help predict trends in stock prices two hours into the future. We hoped to use classical machine learning algorithms and various types of neural networks to try and build a model that had the potential to aid traders in their predictions and thereby help with the quick decision making required in trading.

To this end, we procured tick by tick data from an NSE authorised vendor for various symbols for a period of one year. The data contained the Open Price, High Price, Low Price, Close Price and Volume.

We approached the project as a classification problem containing three classes. We initially picked just one stock – TATASTEEL for our experiments. We used Technical Analysis to add features to the dataset, which we felt might help with predictions. The technical indicators used included Exponential Moving Averages of various lengths and periods, Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Average Directional Index (ADX), and Linear Regression Angle. We also extracted several categorical features from these technical indicators hoping they may add extra information for the algorithms to process.

We tried pre-processing and using this dataset in several different ways. We experimented with resampling the data into different timeframes (5 minute, 15 minute, hourly), different window lengths were tried in order to look back into the sequence, we tried adding features from higher timeframes and using them in combination with each other, and we even tried using a combination of 3-4 stocks to predict trends.

Algorithms and Neural Networks used for predictions included:

- Logistic Regression
- Gaussian Naïve Bayes
- Random Forests
- SVC
- Dense Neural Network
- Convolutional Neural Network
- LSTM

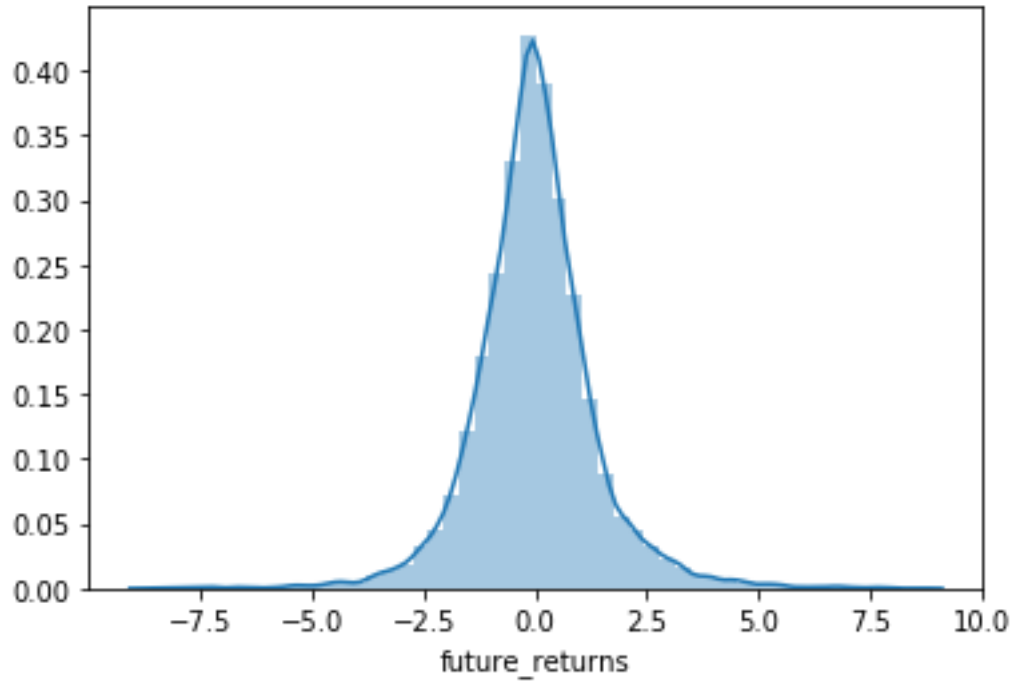
Classes:

We created 3 different class labels. They are as follows:

- Class 0 – The closing price 2 hours into the future has neither gone up nor come down significantly. This class comprises of samples where the returns after two hours have not been above +0.49% or below -0.49%
- Class 1 – The closing price 2 hours into the future has gone up significantly. This class consists of samples where the returns after two hours have been greater than or equal to +0.5%
- Class 2 – The closing price 2 hours into the future has gone down significantly. This class consists of samples where the returns after two hours have been lesser than or equal to -0.5%

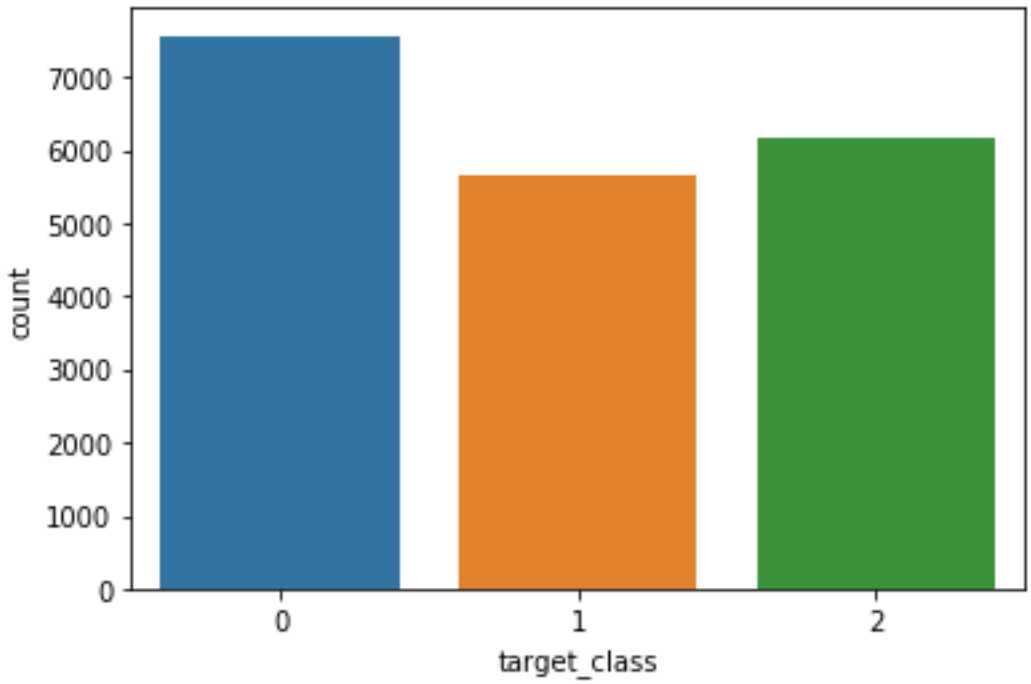
Range of Returns (2 hours into the future):

EDA showed that there was a high fluctuation in the 2 hour returns. There were a lot of outliers and the returns range all the way from -8% to +8.5%.



Class Distribution:

Once the labels were created, the distribution of classes in the dataset was as follows:



As you can see, Class 0 was slightly more dominant than classes 1 and 2. To counter this, we tried using class weights while training our neural networks.

Findings:

With all the various combinations of algorithms, timeframes, window lengths and selection of stocks used in predictions, at best we were able to achieve 51% accuracy in our predictions.

Below is a table that displays the results from all the various combinations which were tried.

Windowing	Window Length	Multiple Timeframe	Multiple Stocks	PCA	LR	NB	RF	DNN	CNN	LSTM
No	0	No	No	No	46.70%	44.14%	35.72%	47.74%	46.83%	47.01%
Yes	24	No	No	No	42.53%	40.67%	44.09%	44.70%	43.44%	44.89%
Yes	24	5min, Hourly	No	No	40.73%	39.49%	51.19%	46.35%	45.09%	46.28%
Yes	24	5min, Hourly	No	Yes	39.08%	39.87%	43.23%	41.11%	41.19%	42.56%
Yes	72	5min, Hourly	No	Yes	39.33%	39.17%	43.94%	41.18%	42.08%	42.12%
Yes	144	5min, Hourly	No	Yes	39.33%	39.17%	43.94%	41.18%	42.08%	42.12%
Yes	24	5min, Hourly	Yes	No	39.08%	39.87%	43.23%	41.11%	41.19%	41.80%
Yes	72	5min, Hourly	Yes	No	39.33%	39.17%	43.94%	41.18%	41.34%	41.53%
Yes	144	5min, Hourly	Yes	No	39.33%	39.17%	43.94%	41.18%	41.05%	41.11%

- As you may notice, that majority of our experiments yielded accuracy scores in the range of 39% to 42%

- What we find most surprising is that in our first experiment where we did not window our data nor use any combination of data from multiple timeframes or multiple stocks, we got accuracy scores of 46.5% to 47.75% using Linear Regression, DNN, and CNN. These scores are better than instances where windowed data was used with DNNs, CNNs, and LSTMs. We did try an ensemble of these models as well but there was no significant improvement in the score.

- Our intuitive expectation was that if more information (longer window, prices of other stocks) is given to the various ML algorithms and neural networks along with class weights, these scores would improve. But that has not happened. All models we built were overfit, and at best we ended up getting scores of around 41%-42% when these approaches were taken.

- The best accuracy score we got came from using a Random Forest, taking 5 minute data, adding on hourly data to it, and using a window length of 24. This gave us an accuracy score of 51.9%. We used RandomizedSearch and GridSearch to try and find the best hyperparameters, but ended up finding that the default settings worked best. Below is the classification report for the same.

Walkthrough of the solution:

- Step 1 : Import 5 minute data of the stock containing Open, High , Low, Close, Volume

- Step 2 : Create additional features using technical analysis (EMA, MACD, RSI,ADX, Linear Regression Angle, etc..), create categorical features based on the common notions of how these technical indicators could be used

- Step 3 : Calculate percentage returns two hours into the future

- Step 4 : Use the calculated returns to create class labels

- Step 5 : Import hourly data of the same stock

- Step 6 : Create additional features using technical analysis (EMA, MACD, RSI,ADX, Linear Regression Angle, etc..), create categorical features based on the common notions of how these technical indicators could be used

- Step 7 : Merge both the dataframes

- Step 8 : Create a sliding window that iteratively captures 24 rows of data at a time as new individual samples

- Step 9 : Scale the data using zscore

- Step 10 : Split the data into train and test sets with a ratio of 80:20 without disturbing the sequence of the data (this is done in order to ensure that the model does not look into the future while learning)

- Train the Random Forest model and make predictions on test data

Model Evaluation:

The classification report of the model is as follows:

```
Train set accuracy = 0.9999111111111111
Test set accuracy  = 0.5101315321720583
_____________________________________________________
Confusion matrix:
[[893 153 270]
 [341 183 151]
 [309 154 359]]
_____________________________________________________
Classification report:
              precision    recall  f1-score   support

           0       0.58      0.68      0.62      1316
           1       0.37      0.27      0.31       675
           2       0.46      0.44      0.45       822

    accuracy                           0.51      2813
   macro avg       0.47      0.46      0.46      2813
weighted avg       0.49      0.51      0.50      2813
```

As you can see, this model too was overfit and despite efforts could not be regularized satisfactorily. In the end we decided that we may as well go with this model since it had the best K-Fold Cross Validation Score in comparison to any others that were built.

The precision and recall on classes 1 and 2 are quite low. The benchmark we had hoped to achieve was an accuracy score of 60%.

Implications and Limitations:

Given that an accuracy score of 51% cannot really be considered as better than random, it is likely that this model cannot be used in trading.

It is quite possible that the models that we have been trying to build are probably not being given the correct data and hence are not able to get to the benchmark that we had set out to achieve.

Closing Reflections:

Predicting stock prices is a complex task and there isn't really a perfect equation that can be distilled to do this. However, we would still like to believe that the benchmark that we set out is achievable provided that the models are given the right inputs for training.

Perhaps, in future we will try combining stock prices with data of different indices and commodity futures to see if this helps in any way. Instead of trying to predict 2 hours into the future, we may also try reducing that to 1 hour or less.

We thoroughly enjoyed conducting all these numerous experiments during the course of this project. Despite not achieving what we had set out to, we are glad that our learnings from this project will add value to the research and experiments which we will continue to conduct in future.

----Thank You----