Price forecasting of financial instruments

Made possible by Brainjar

19/05/2021

3 IT Factory
Artificial intelligence

College year 2020-2021

Stijn Van Peer

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

LID VAN ASSOCIATIE KU LEUVEN

brainjar

THOMAS MORE

Description of approach and own contribution

Originally, I applied to Brainjar, for another internship regarding explainable artificial intelligence. However, after discussing my (shared) personal interests, a different suggestion was made. Why not try to create a model that may guide investors and traders? From that moment on, I started going through all the possible research I could get my hands on. I noticed almost immediately that it is difficult to set a particular benchmark for such a project. As anticipated, there is also little or no information available around such projects, as this is usually not made public. Therefore that, together with my internship supervisors (Olivier Latrez & Maarten Bloemen), I mainly relied on our intuition how such problems could and should be solved.

Acknowledgment

I would like to thank Brainjar for the unique opportunity they have given me. I have never known a company where there is a better balance present between internal atmosphere and domain knowledge of staff members. More specifically, I'd also like to thank Olivier and Maarten for taking the time to help me. Finally, I would like to thank my supervisor at Thomas More (Bram Heyns) for guiding me through the process of an internship and my bachelor’s degree in general.

Table of Contents

1 Introduction

Financial markets include any marketplace (exchange) where shares are traded, such as the stock market, bond market, forex market, and derivatives market, to name a few. Financial markets are essential for capitalist economies to function properly.

1.1 The 90-90-90 rule

Apart from the fact that such markets are essential for ensuring economic stability (aside from the cryptocurrency market), it is also possible to engage in market participation as a retail investor/trader. The people who are buying and selling instruments are essentially the ones who drive the economy. Because of its accessibility, the degree of sophistication of market speculation is often underestimated. It's common knowledge in the investment community that actively managed investment funds underperform prominent market indices like the S&P 500. Investing professionals who make it their full-time task to beat the competition usually can't. 90% of traders lose 90% of their capital in 90 days.

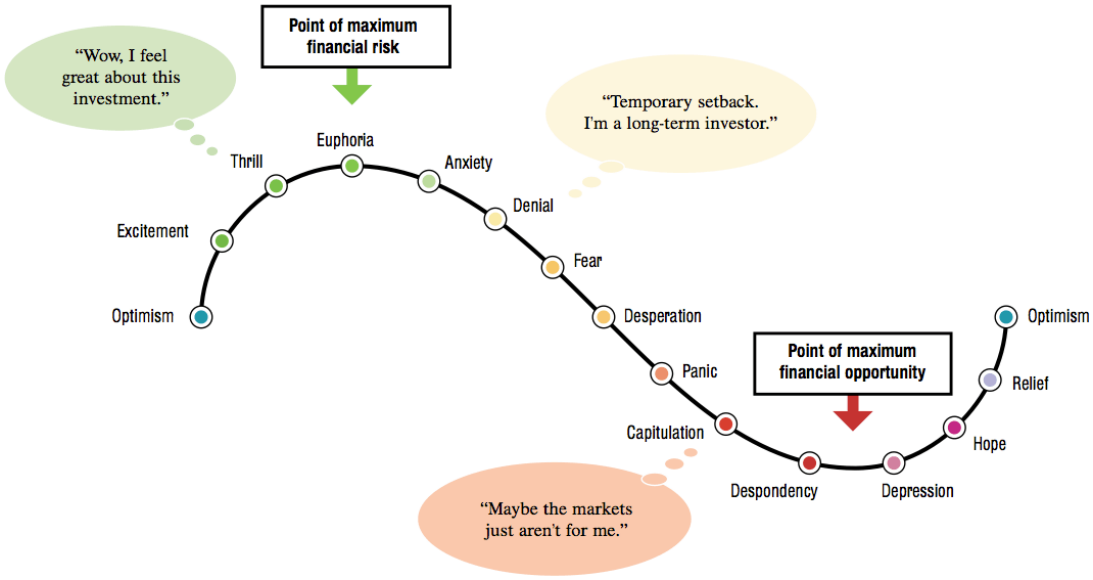


Figure 1.1

1.2 The better player

This is where artificial intelligence comes in. Computers can consider many times more influential features than the human brain can. Human psychology is also one of the most common reasons for poor market speculation. According to my understanding, an algorithm that possesses a sense of potential future value (overpriced/underpriced) could truly guide investors/traders.

2 Strategic approach

Technical and fundamental analysis are two forms of analysis that can be used to make predictions regarding exchange-traded instruments. The reason why I chose technical data (historical price data) is because of the restrictions on fundamental data (company data).

2.1 Data exploration

Price data is more commonly described as ohlc data, which represents the open, high, low and closing price of a candlestick (boxplot). Through such a set of time series-based representations one can describe the extremes of a given interval being a minutely, hourly, daily, weekly or even a monthly one.

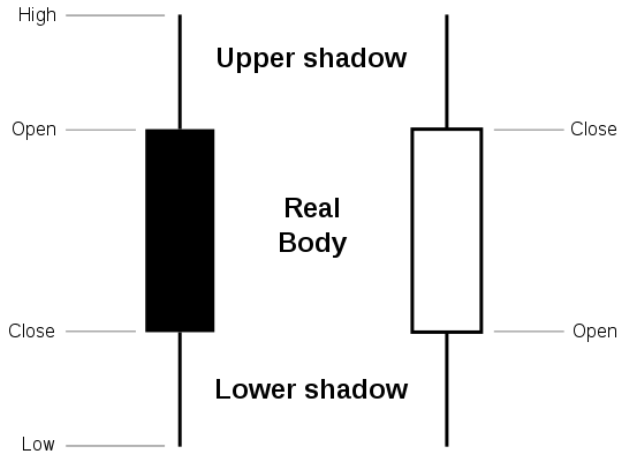


Figure 2.1

Let me clarify it some more with a practical example.

	Currency	Date	Closing Price (USD)	24h Open (USD)	24h High (USD)	24h Low (USD)
1	BTC	2013-10-02	125.455000	123.654990	125.758500	123.633830

Figure 2.2

Back in the good old days, Bitcoin was priced around $125. The interval we are dealing with is in this case a daily one, meaning that each line describes the ohlc values for a session on a particular exchange. Bitcoin started off trading at around $123.65 and finished (closed) trading at $125.46. During that particular day, the following happened; the price dropped to about 123.63 (session low), rose to $125.76 (session high) and sold off just a little (close = $125.46).

2.2 Additional measurements

Finding the right features for my model was perhaps one of the biggest challenges for me. It really was an ongoing process. It consisted mainly of making simple predictions, adding or modifying features and doing it all over again. I'll clarify it again briefly using a practical example.

I figured out that if my model could reach a consistent benchmark on the cryptocurrency market, which is the hardest market to predict due to constant changing volatility (a measure for price fluctuation), it would work out just fine on the rest of them. The hard part was getting it to work on cryptocurrency.

When I tried to forecast the closing price of tomorrow's session (using an 80/20 split), I was in for a set of quite mixed results;

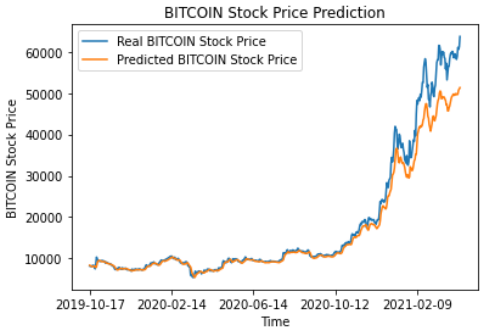


Figure 2.3

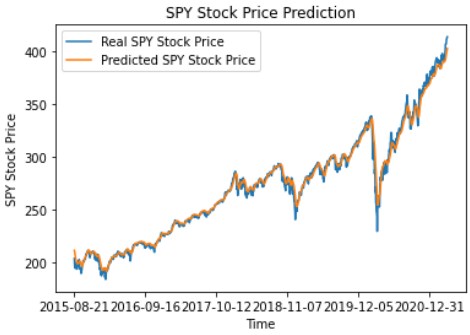
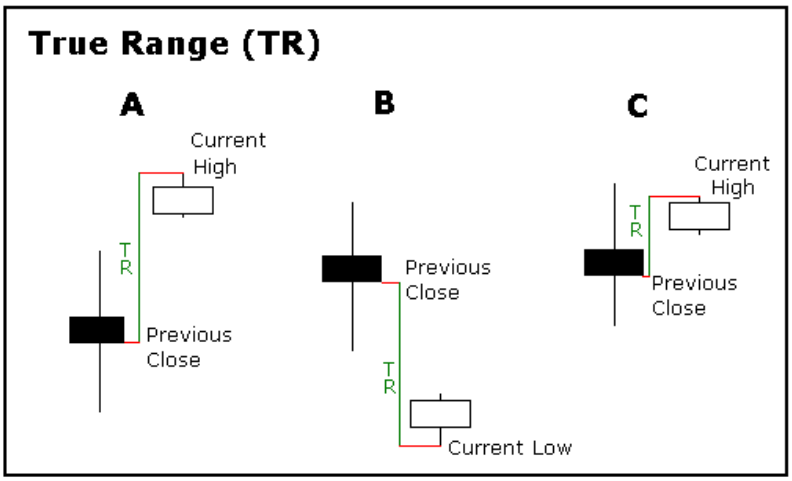


Figure 2.4

How could it be that the SPY (SPDR S&P 500 ETF Trust) performs way better all the way through? It all made sense fairly quickly. I was trying to predict the most volatile market in the world, without even considering the continuous changing volatility as a feature.

A rather smart way to parameterize volatility, is a concept better known as the average true range (ATR);



Volatility = Average (TR1, TR2 & TR3)

Figure 2.5

The process I described above was repeated several times. By doing so, I added several measurements, each of which was calculated on ohlc data.

```
Index(['Currency', 'Date', 'Change closing price', 'Change open price',
       'Change high price', 'Change low price', 'Signal',
       'Algorithmic return o-h', 'Algorithmic return o-l',
       'Algorithmic return o-c', 'Volatility (ATR)', 'Expo ma h', 'Expo ma l',
       'Expo ma c'],
      dtype='object')
```

Figure 2.6

2.3 Data preprocessing

Data preprocessing is the most important step in building any model. Real-world data is often incomplete, unreliable, misleading (due to mistakes or outliers). This is where data preprocessing comes into play, as it facilitates in the cleaning, formatting, and organization of raw data.

2.3.1 Methods of transformation

Brainjar expects me to develop a universally applicable model, meaning that it should be able to make meaningful predictions on every financial market. This actually means that the training process of our algorithm can't be affected by the share price of an instrument throughout time, which may fluctuate.

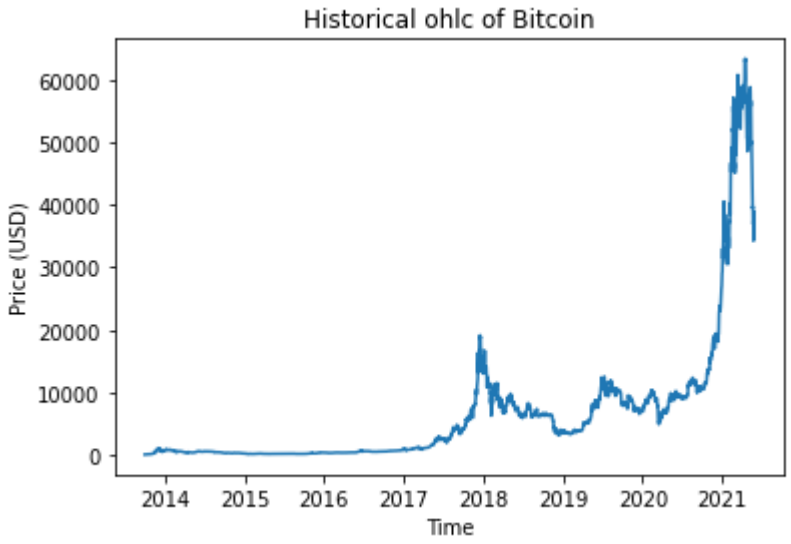


Figure 2.7

Needless to say, the patterns acquired in the data between 2014-2018 are completely different from those of recent data (2020-2021). I considered the subsequent transformation methodologies;

- MinMaxScaler, StandardScaler
- Price in terms of (algorithmic) returns (%)

Any scaler (MinMaxScaler & StandardScaler) would significantly alter my algorithm's scalability, due to the fact that future prices may be higher than the ones scaled in my training. I ended up preprocessing my data by calculating the change between my features.

Algorithmic return o-h	Algorithmic return o-l	Algorithmic return o-c	Volatility (ATR)	Expo ma h	Expo ma l	Expo ma c
0.031485	-0.001850	0.015297	-0.070541	0.009067	0.013824	0.008980
0.003768	-0.014774	-0.005425	-0.099798	0.003570	0.010845	0.005286
0.006023	-0.017311	0.005158	-0.082998	0.001869	0.006000	0.005257
0.007665	-0.005183	0.003613	-0.106478	0.003114	0.008910	0.004841

Figure 2.8

2.3.2 Describing relationships

- Returns in terms of feature change itself
  - » Closing price = (closing price – previous closing price) / previous closing price
- Returns in terms of change among features
  - » Session high = (session high – session open) / session open

The first row is thus inevitably removed from the dataframe (2 rows required for estimation). In this manner, the trend per feature and the interrelations between features are both described.

3 Supervised learning

Supervised Learning is a machine learning methodology that involves working with labeled data in order to make meaningful predictions. Labeled data ensures that the dataset you'll use to model includes both the properties and the result of what you're trying to forecast.

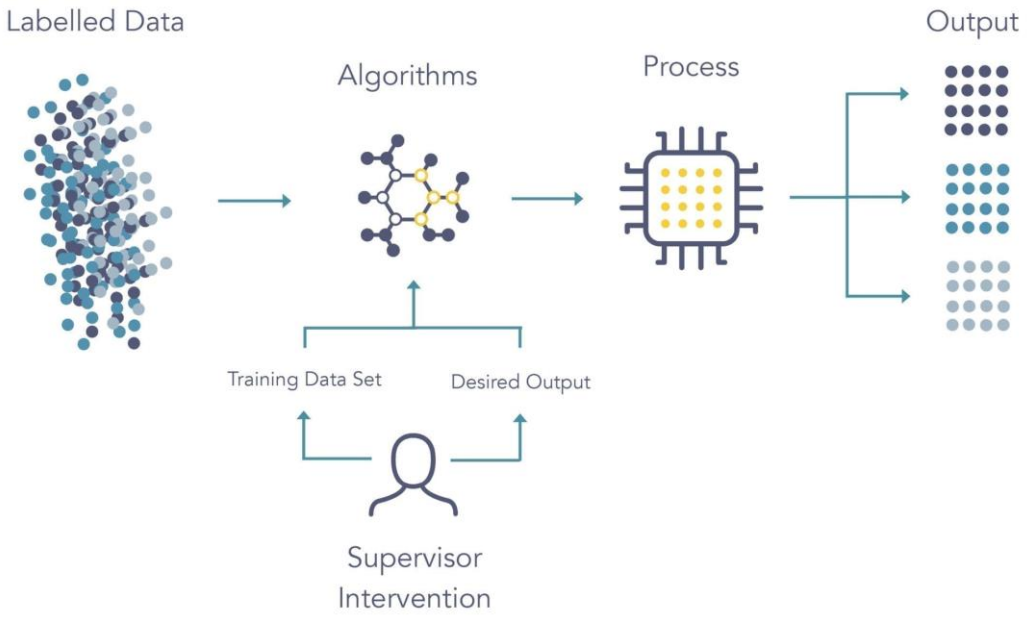


Figure 3.1

3.1 Algorithm architecture

I started off with a set of fairly simple models from scikit-learn, as I believe that such less complex models often work well too. However, due to the fact that acquired patterns in ohlc data are constantly changing, a lstm model proved to be more appropriate.

As its name suggests (long short-term memory), a lstm has the ability to simply ignore patterns that are no longer valid. They were originally developed in order to deal with the vanishing gradient problem that may occur when training traditional recurrent neural networks.

3.2 Mapping my input(s) to an output

Throughout my internship, I changed my output several times so I could make my model as useful as possible. I have considered the following possibilities;

- Sentiment prediction (daily & weekly)
- Value prediction (daily & weekly)
- Signal prediction (buy & hold)

3.2.1 Sentiment prediction

Sentiment prediction is better known as predicting market phases. By doing so, people try to get a notion of the direction that price will pursue.

I prepared batches consisting of 2 months' worth of previous closing prices. The interval (60 days) came from a series of previous attempts and proved to be the best fit. My output was repeatedly built of 7 closing prices, that came directly after the batches of 60 I fed my algorithm.

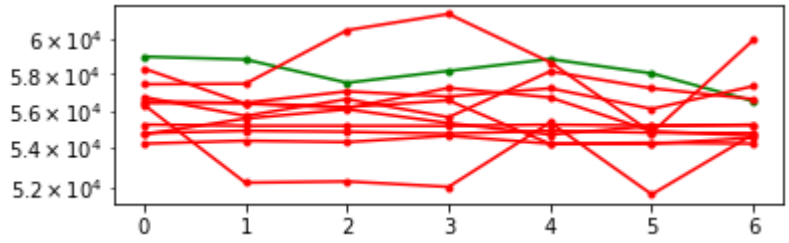


Figure 3.2

I trained my model several times so I would have multiple forecasts. The reason for this being that this way I could make an average of each day's predicted closing prices in order to obtain better overall prediction quality.

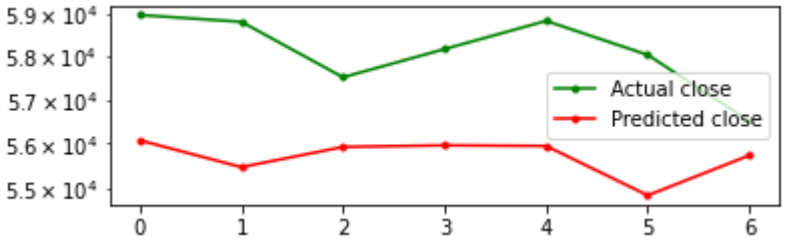
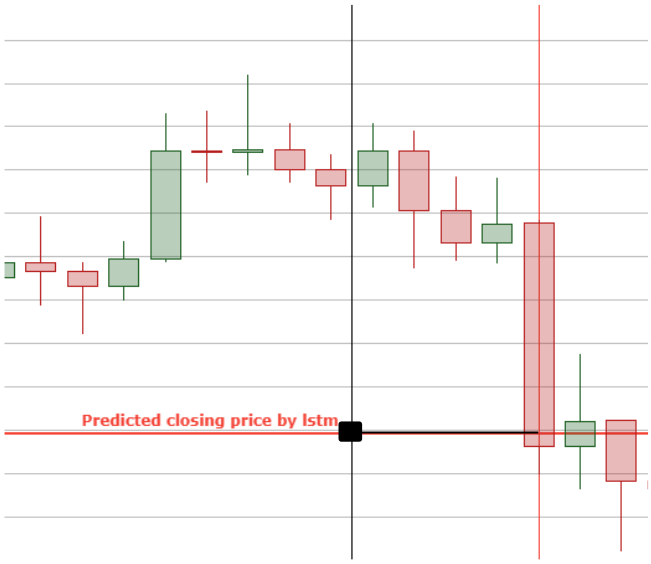


Figure 3.3

I abandoned this method fairly quickly due to limited 'accuracy'. The reason being that you actually have to correctly estimate 2 variables, price and time. It was of course merely an experiment to get a better idea of what was possible.



The prediction is +- 4 hours off.

Figure 3.4

3.2.2 Value prediction

The idea behind this way of working is mainly to create opportunities and to mitigate risk. When you rely purely on sentiment (closing price), you have no clue about intermediary fluctuations or extrema (session high & session low).

This time my input consisted of just about every feature at my disposal. My output consisted of two variables being the positive extrema (high) and negative extrema (low) of the time interval being considered (daily or weekly).

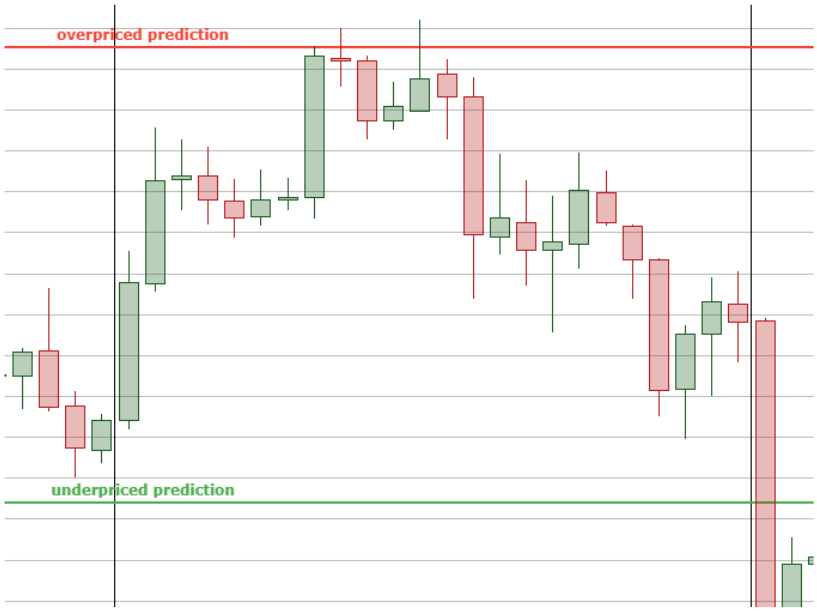


Figure 3.5

Predicting value already yielded better performance, being that we only need to predict 1 single variable (price) within a given interval and not at a given time. The problem with this way was mainly the micro unpredictability of price itself;

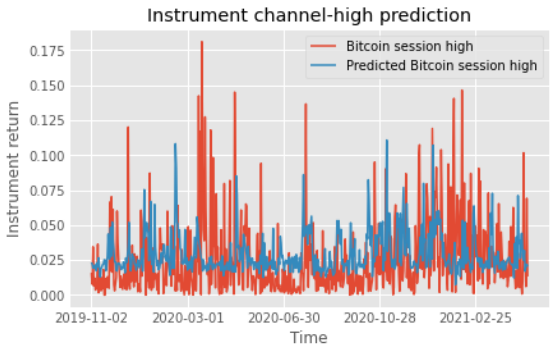


Figure 3.6

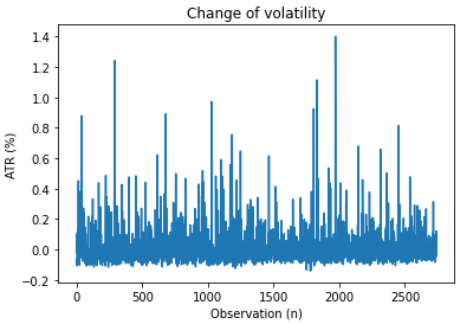


Figure 3.7

My model's predictions were either perfect or bad due to sudden change in volatility. Predicting the price itself on a consistent basis is virtually impossible due to the enormous number of factors influencing it. But what if we could make it a classification problem?

3.2.3 Signal prediction

Predicting prices of any kind proves to be extremely difficult. But is it really necessary to predict the price of tomorrow's extrema with fairly high precision? In my latest version, it has emerged that it is completely unnecessary.

I used a combination, so to speak, of the two previous methodologies, which were not viable individually. I started looking for how much return on average a particular instrument was yielding for 7 days. Whenever a particular time period returned double or more (x >= 2 * avg), I classified it's first day as a buy opportunity (1). All other weeks are classified as a hold signal (0).

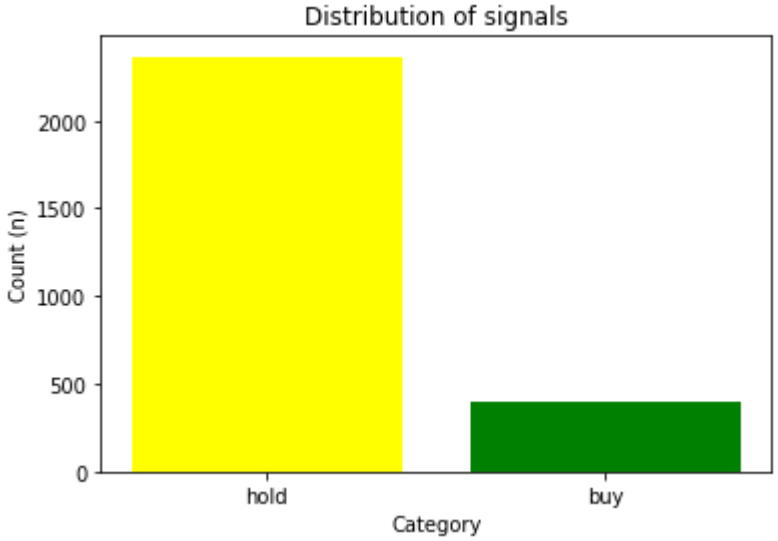


Figure 3.8

The reason I chose a factor of 2 (x >= 2 * avg) is because of the amount of noise (false signals). At first glance this may not seem so, but the hold class is the most important of the two. If our model can recognize normal conditions with fairly high accuracy, it may also recognize abnormal ones.

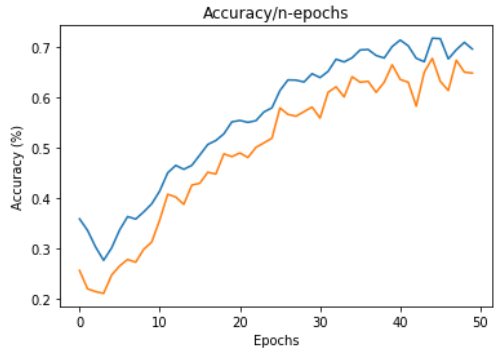


Figure 3.9

	precision	recall	f1-score	support
0	0.94	0.65	0.77	475
1	0.24	0.71	0.36	73
accuracy			0.66	548
macro avg	0.59	0.68	0.56	548
weighted avg	0.84	0.66	0.71	548

Figure 3.10

4 Backtesting

Throughout my internship, I found it very challenging to conduct a quality assessment on my model. Since there are no existing standards, it is difficult to evaluate the progress you have made so far. To address the foregoing, I decided to create a backtest that should reflect the current quality of the model.

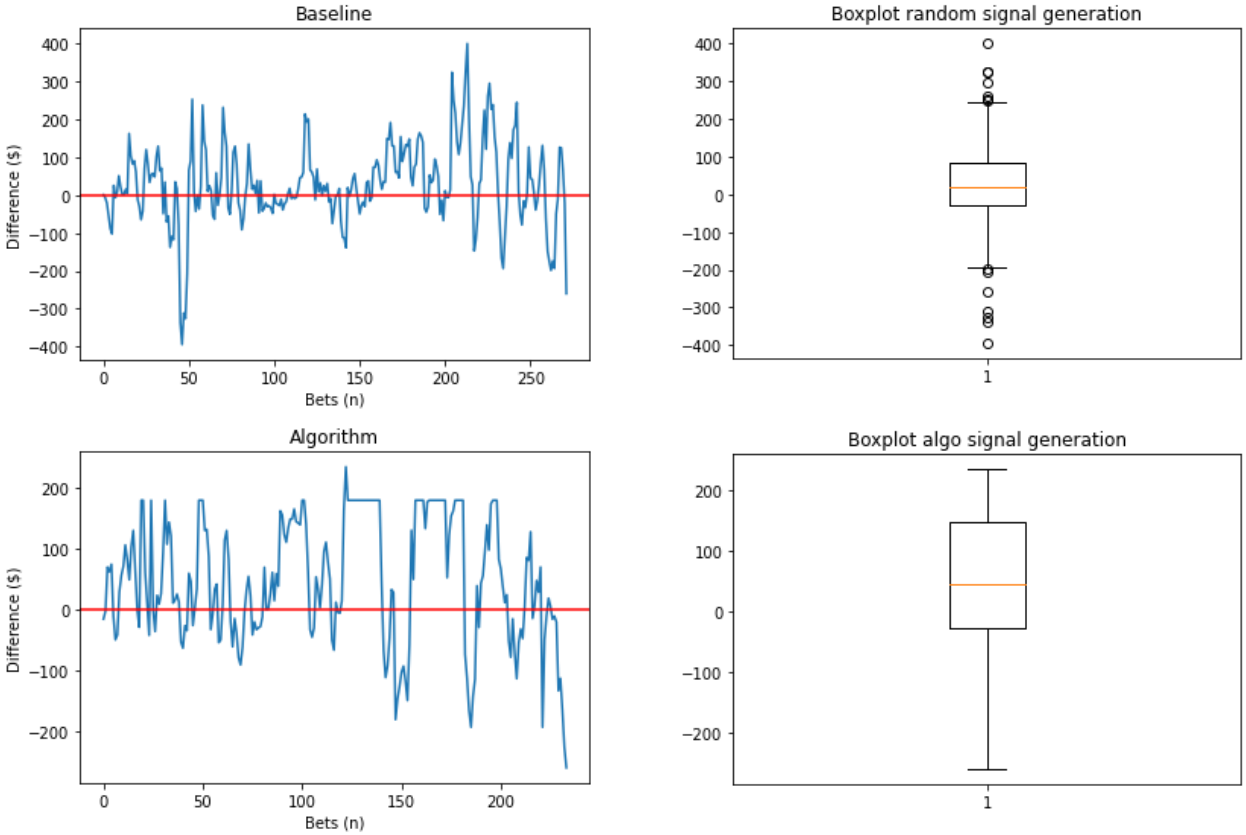
4.1 Deceptive market conditions

During backtesting, it is important to look at the profitability of our program compared to a certain baseline. As a baseline, we chose a random signal generator, which returns either 'hold' or 'buy' regardless of market conditions.

At first glance, this sounds like a bad baseline, but it actually isn't. If by chance one had bought a few times during the last few months, due to the exceptional market conditions one had made a profit without having any idea of what they were actually doing.

4.2 Performance

Let's compare the performance (profit-and-loss distribution) of both our algorithm and the baseline we provided;



Random performance ($);

```
DescribeResult(nobs=272, minmax=(-394.50004255927087,
400.32185806270036), mean=28.655248325087015,
variance=11779.614636146534, skewness=-0.1740736508436697,
kurtosis=1.8989559638319777)
```

Algorithmic performance ($);

```
DescribeResult(nobs=234, minmax=(-260.06318249311323,
233.7887289889713), mean=48.102112810985254,
variance=10537.474237758943, skewness=-0.32657082305730106,
kurtosis=-0.5710412302467658)
```

When you compare the statistics, you immediately notice that our algorithm covers it's position much more efficiently; mean_algo = +- 1.70 * mean_rdm. Despite the market conditions, our model even managed to beat the baseline by a fair amount (+- 3500$).