**Proof-it**

Proof-it is a machine learning model and a backtest algorithm that has been coded with Python/VBA. It aims to build a unique trading tree which identifies the general daily track of an instrument. The algorithm uses the said tree to find the optimum take-profit/stop-loss (TP/SL) ratio and shows the best results for each strategy in it.

The logic behind the algorithm while building trading tree is very similar to those of Decision Tree and Random Forest ML models. However, it is not exactly the same due to the unique features of the financial data. Time range of the data and limited features (buy and sell) are two unique features that is worth mentioning.

Firstly, in order for the algorithm to run correctly, reasonably detailed data must be used to identify the order of the increase/decrease of the price (because in daily data it can not be undestood whether take-profit or stop loss comes first.) Therefore the most suitable approach can be to use 1 minute data. However, this way fairly increases the amount of data and algorithm running-time. Also, it brings about the drawbacks of overfitting. Secondly, in our case the features that are taken into account while splitting the leaves may only be “buy” or “sell”. It means that our tree can not have huge ramification and it actually enables us to generalize the daily track of an instrument.

While splitting the leaves of the tree, the algorithm uses maximum expected return rather than using gini index and information gain. Besides, it does not assign a label (like 0 or 1) to the test data like the mentioned models do, but shows the possible trading track and the possible return according to the tree built by using training data.

To find the accuracy score, the algorithm compares the possible trading return based on the tree with the daily return of the instrument based on the strategy of buying at the beginning and selling at the end of the day. However, the algorithm takes into account the net profit or sharpe ratio while building the tree.

Because the algorithm imitates the logic of two abovementioned models, it also inherits their limitations. Firstly, the tree is not always the most optimal tree, because the optimal tree can only be found by scanning all nodes at the same time and it is almost impossible while the number of the trading step increases (for 1-step there are 40^2 combinations, for 2-step 40^14; for 3-step 40^65). So the algorithm uses an approach that substitutes the optimal tree within reasonable running-time.

Secondly, tree models have a potential of being overfitting due to the reason that the structure of the tree is too dependent on the training data. However, the algorithm uses some ways to manage this problem like pruning the leaves in order to shrink the size of the tree, cross-validation and Random Forest.

To use the algorithm like Random Forest, some arrangements should be made (e.g. changing the strategy as closing the position at the end of the day in order to make the daily data independent) so that new training data can be derived from the original data.

The main idea behind this algorithm is to assume that the daily movement of financial instrument can be generalized. Although it seems that movements are unique and unpredictable, it can be possible to gain an understanding about the probability of the result (profit or loss) and amount of return by scrutizing the tree.

Currently, it includes 8 sub-packages;

* X-ray: it shows the basic info of an instrument (# of positive close, annual % change, volatility, beta, correlation with market etc.) and its best strategy (return, sharpe ratio, # of trade, # of day in position etc.)
* Data: it consists of 15-minute OHLC prices of the stocks since 01.09.2014.
* Fix data: it changes the data according to split ratio for the strategy which does not close the position at the end of the day.
* Strategies: it holds 3 main strategies and its branches.
* Porfolio: it enables to see the results of the portfolio strategies.
* Stock-track: it shows the TP/SL ratio, TP/SL prices, TP/SL frequency and expected return of each position for all stocks. It helps you to understand if the current performance of your stock differs from the past average performance.
* Results: it keeps all kinds of result (strategy return, ratio, info etc.) for each instrument for comparing with each other.
* Documentation: it shows in which order you should follow the steps to get the best results.

It has two user interfaces; first one is finding the optimum TP/SL ratio according the criteria (max net profit or the best sharpe ratio) selected. On the other hand, second one allows you to find out the track and the detailed results based on the ratios found in the previous step.

After building the main strategies, many scenarios were added into the code in order to see their conribution. If a scenario made positive contribution for more than half of the instrument in the database, it was named as an improvement and it was added to the main code or took part as sub-package like “Portfolio”. Proof-it has 5 improvements already.

* Differentiate trading ratios: Because the movement of an instrument mostly changes according to whether the price is above or below of the X-day MA, the algorithm tries to find different trading ratios in order to make maximum contribution to profit/sharpe. As a result of some tests, Proof-it is using 19-day MA (Moving Average) to find optimum trading ratios. A different methodology rather than MA can be used for differentiation.
* Time Analysis: it uses average duration of position for increasing strategy return
* Sharpe ratio: you can buid the trading tree according to best sharpe ratio beside max net profit.
* Long+Short position together: you can use only long, only short strategies or use them both for different periods of trade.
* Portfolio: You can make a portfolio.

Following steps show how the system works;

1. First determine the stock, time of period, strategy, scanning criteria (max net profit or sharpe ratio) and extend of scan/pace of scan. Use first interface for scanning and finding the optimum ratios. Then use second interface to find out the trading tree and detailed results.
2. Divide the trading period based on a criteria (Proof-it uses 19-day-MA for it) and scan again.
3. Change the positioning criteria (only buy, only sell or both) and scan again.
4. Use time-analysis and scan again.
5. Make potfolio if the strategy is suitable for it and and scan again.
6. Compare all the results and pick the best.
7. Send the best strategy to “X-ray” and “Results” sub-packages.

Types of strategies in Proof-it:

* Buying in the closed session / Buying within day
* Closing the position at the end of the day / Closing the position when the price reach the calculated price.
* Strategy with buy/sell/or both positions
* 1/2/3 steps strategies

System it contains:

* ML Algo: Python/VBA
* Database: SQLite3
* Visualization: Matplotlib, Seaborn