ALGORITHMIC TRADING

DSO 530 Final Group Project

Group Member:
Hakan Imrohoroglu (8999975990)
Nanchun Shi (1219637819)
Jiayue Chen (3367772528)
Bingru Xue (7861503871)
Yuyao Shen (5123444853)

Contact Person
himrohor@usc.edu

USC Marshall
School of Business

CONTENTS

PROBLEM STATEMENT

The trading problems tackled in this project are questions that every major investment firm faces on a regular basis. How can we predict future stock behavior? What kind of models work best? Can our models outperform market averages? How do we design the decision-making process that connects our prediction model to investment decisions? The financial implications of a well-designed prediction and decision model are quite obvious. Retail and institutional investment firms manage massive amounts of money and even small percentage improvements over market averages can mean billions of dollars in profits. The fundamental issue is uncertainty. Hundreds of variables (many of which are difficult if not impossible to identify or quantify) have an impact on a stock's performance. Not to mention major unexpected economic shocks (i.e. Coronavirus) can render even the most carefully designed trading algorithms useless in just a matter of days.

Any prediction and decision model regarding financial markets will be limited by this significant uncertainty. Algorithmic trading is an essential part of financial market movements and the models developed in this project will be intended to function within an algorithmic trading framework. The specific scenario considered in this project is a simplified version of the larger investment strategy problem. We have $1,000,000, 50 stocks, and 251 days to maximize our ROI. Various market realities will be considered throughout the model building and evaluation process. Transactions costs will not be ignored. Aspects of prospect theory in behavioral economics will inform the decision-making process. The objective function of our model must find some balance between profit maximization and risk minimization. This is a very difficult objective to work towards. However, by using the appropriate statistical approaches combined with a deep understanding of market realities we hope to design a trading algorithm that yields successful financial results.

DATA MANIPULATION

Data Description

The raw data are the daily stock data of 50 companies, which are constituents of the SSE 50 index. The period of our data is from Day 1 to Day 756 (in the years 2017-2019). The data of one day includes open, high, low, volume, and adjusted close. The adjusted closing price, in particular, is the closing price that takes the dividends, stock splits, and other factors into account and hence is a better reflection of the current value of a stock. Thus, the adjusted closing price is used when calculating percentage change of the price of a stock and the return of a trade. Together with the raw data, we also received 11 engineered features calculated using raw data.

Field name	Field type	Min	Max	% populated
Open	Numerical	4	8555	100%
Close	Numerical	4	8574	100%
High	Numerical	4	8629	100%
Low	Numerical	4	8535	100%
Volume	Numerical	247900	1872299776	100%

Variable Creation

We have created 5 categories of variables, 50 variables in total. Most of them were calculated using a package called ta-lib and we also defined some indicators by ourselves. Variable examples are as follows:

- Overlap functions:
  - EMA: Exponential Moving Average is a type of moving average that places a greater weight and significance on the most recent data points.

    $$EMA = EMA_{-1} + \frac{2}{n+1} * (EMA_t - EMA_{-1})$$

  - DEMA: Double Exponential Moving Average Technical Indicator which is used in a similar way to traditional move averages. The average helps confirm uptrends when the price is above the average and when the price crosses the average that may signal a trend change.

    $$DEMA_t = 2 * EMA_t - EMA(EMA_t)$$

- Momentum functions:
  - +DI: Positive directional indicator is used to measure the presence of an uptrend. When the +DI is sloping upward, it is a signal that the uptrend is getting stronger

    $$+DI = (\frac{Smoothed+DM}{ATR} * 100)$$

  - STOCH: a stochastic oscillator is a momentum indicator comparing a particular closing price of a security to a range of its prices over a certain period of time. It is used to generate overbought and oversold trading signals

    $$\%K = 100 * \frac{Close - LowestLow_{\text{last n periods}}}{HighestHigh_{\text{last n periods}} - LowestLow_{\text{last n periods}}}$$

    $$\%D = MovingAverage(\%k)$$

- Price transformation:
  - WCLPRICE: the weighted close price

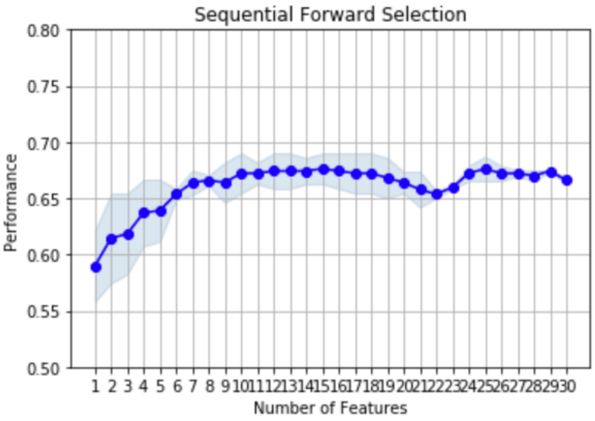
$$WCLPRICE = \left(\frac{C_P * 2 + P_H + P_L}{4}\right)$$

- Pattern Recognition Indicators
  - CDLBELTHOLD: Bearish Belt Hold is a bearish reversal pattern, appearing in an uptrend. In this pattern, the day opens at its high level, but then price falls and closes near its low, not necessarily at the lows of the day
  - CDLENGULFING: A bullish engulfing pattern occurs in the candlestick chart when a large white candlestick fully engulfs the smaller black candlestick from the period before. This pattern usually occurs during a downtrend and is thought to signal the beginning of a bullish trend in the security
  - CDLHIGHWAVE: The high wave candlestick has a very small real body, and it typifies a stock or index plagued by uncertainty. The spinning top has small upper and lower shadows, whereas in the high wave the shadows are longer, revealing more volatility

Feature Selection

After the feature creation process, we have 50 variables in total, because of the limited size of our dataset, we moved forward with feature selection. We applied sequential forward selection. After trying different numbers of features, we decided to choose the best 15 features for each model.

The following plot is the number of features used versus the accuracy for a certain estimator:



MODELING/STATISTICAL LEARNING APPROACH

At the beginning, we tried both classification and regression models and to be flexible with trading strategy, we have 3 response variables for both models. Our classification models have 1-day up or down, 3-day up or down and 5-day up or down and for regressions, we have 1-day, 3-day and 5-day average return. For example, if we are on day 18, then 1-day up or down for today is obtained by comparing returns of day 17 and today.

When training the models, we applied different sliding windows (30, 60, 120). That is, we only use the most recent t days data for training and predict for the next day. After experiment, we found that for our models, the training accuracies are better when using all available data in the past. Therefore, for all the analysis below, we will use all available data for prediction.

However, our regression models' performances were only mediocre, and we decided to move along with classification models only.

We considered some appropriate models as options for this very difficult prediction problem. 6 different types of models were employed:

- K-nearest Neighbors
- Logistic Regression
- Random Forest
- Boosted Trees
- Support Vector Machine
- Neural Networks

Since we have 50 different stocks and each stock has 3 different response variables, we decided to try all 6 models for all stocks, which brought us 900 models with different features selected. From those models, with the use of cross validation, we picked the relatively stable model based on its performance on training data.

NOTE: We wanted to take a look at some of the metrics/distributions between the training set and testing set to get an idea as to whether or not these two sets of data (time) behaved similarly. Over a large enough amount of time stock market returns are quite predictable (as a whole). However, short term projections of specific stocks are essentially a random walk. We therefore realized that the success of any predictive model would require that there be some similarity in stock behavior between the training and test set. We found a significant difference between the two sets, which informed us that we would be facing major difficulties in predicting stock movements.

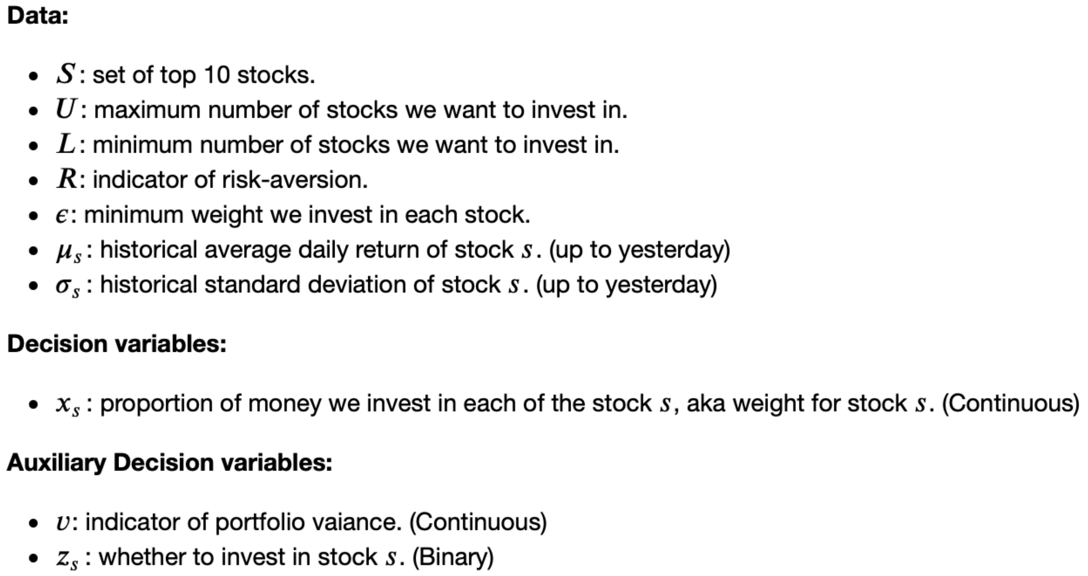
TRADING STRATEGY

We have several assumptions for our trading strategy:

- Sell stocks and make purchases in the last three minutes of trading hours and sell first
- No option trading
- T + 1 trading rule
- Fraction of a share is allowed
- Unlimited Liquidity
- Transaction cost of 0.03% for a roundtrip

The trading strategy we employ was developed via an optimization model through python. We developed a total of 150 models (3 for each stock). Based on the three response variables, i.e. 1-day, 3-day and 5-day, we can twist the weights assigned to each variable according to our trading strategy.

On each investing day, we rank the average predicted movement for each stock and invest in a certain number of stocks that have the highest predicted performance and sell stocks we owned with low-predicted-performance, bottom 40. To automate optimization process, we utilized Gourbi:



Data:

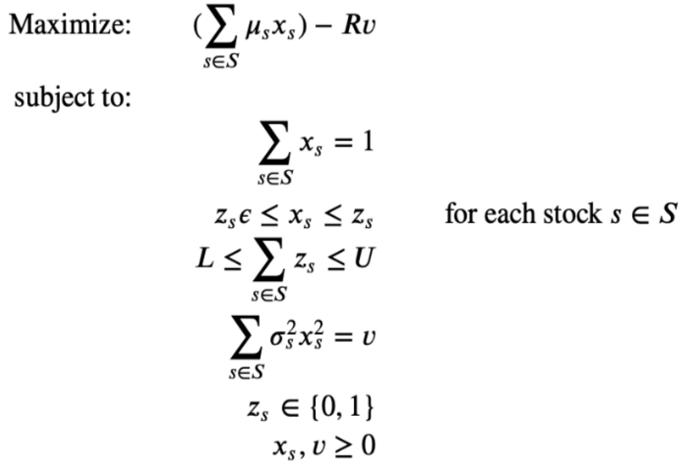
- S: set of top 10 stocks.
- U: maximum number of stocks we want to invest in.
- L: minimum number of stocks we want to invest in.
- R: indicator of risk-aversion.
- ϵ: minimum weight we invest in each stock.
- μ_s : historical average daily return of stock s. (up to yesterday)
- σ_s : historical standard deviation of stock s. (up to yesterday)

Decision variables:

- x_s : proportion of money we invest in each of the stock s, aka weight for stock s. (Continuous)

Auxiliary Decision variables:

- v: indicator of portfolio vaiance. (Continuous)
- z_s : whether to invest in stock s. (Binary)



$$\text{Maximize:} \quad \left(\sum_{s\in S} \mu_s x_s\right) - Rv$$

subject to:

$$\sum_{s\in S} x_s = 1$$

$$z_s \epsilon \leq x_s \leq z_s \qquad \text{for each stock } s \in S$$

$$L \leq \sum_{s\in S} z_s \leq U$$

$$\sum_{s\in S} \sigma_s^2 x_s^2 = v$$

$$z_s \in \{0, 1\}$$

$$x_s, v \geq 0$$

This weighting strategy gives each stock a maximum weighted average of 1 and minimum of 0. This means that for a given stock if all 3 time-models predict UP, its weighted average is 1. Stocks with this weighted average would fall under the "invest" bin for our model. As part of a general risk minimization and diversification strategy we set the following rules:

- Minimum of 5 stocks will be purchased/held each day
- Maximum of 10
- Maximum of 80% of total funds ($800,000) can be invested/held at any moment.

If we own any of the bottom 40 performing stocks based on their predicted weighted averages, we sell.

The objective of this linear programming is to maximize the estimated return of the portfolio and minimize the portfolio "risk". Here risk is in quotes because given that the data for this project is a simplified version and days of each stock may not match with each other, it would be not reasonable to calculate the covariance between different stocks. Therefore, when calculating the risk of the portfolio, we removed the covariances in the original formula and only kept the variances of each stock. This gives us an indicator of portfolio risk. Mathematically, it would be smaller than the real risk measure. We also set a risk parameter (R) for various trading strategy options within our linear programming model (this parameter can take values between 0 and 1). This measures the risk-aversion of investors. When R increases, the optimization process will weigh more on the portfolio "risk".

Our decision strategy is fundamentally guided by a risk reduction position. Given that specific stock movements in the short term are a random walk we acknowledge the need to diversify and mitigate risk. Additionally, reducing transaction costs means reducing the number and value of transactions where possible. Algorithmic trading in the real world is an incredibly high speed and computerized action. For this project we need to balance the assumption that we are trading frequently with the associated transactions costs and unpredictability.

When evaluating various parameter combinations for our trading strategy we decided to consider a balanced set of metrics as our objective function. This means that we do not consider total profit alone to be the deciding indicator of which trading strategy we employ. Instead, we consider total profit, days positive, and the Sharpe ratio. We tested different parameter combinations on the training set. Based on the result, we picked the combinations that gave us the most balanced outcome for all of the three metrics. Then we implemented our strategy on the test set. The optimal decision strategy we identified gives us the following results against the test data:

Summary of results

Baseline Model:

In order to test the effectiveness of our trading strategy, we made a naive investment plan on the test set. We equally distributed capital on the first day to each of the stocks and sold them on the last day. The profit we got is: $134,010.

Our Model:

1. Aggressive Investing:

An aggressive investing means we consider less about risks and reserve less money on hand each day. Specifically, we only reserve 10% cash each day. The best parameters we got from the training set is:

- Bottom 40 stocks considered for sale
- Weight allocation: 1-day model (0.3), 3-day model (0.3), 5-day model (0.4)
- Risk parameter = 0.1

The results on test set:

- Profit: $263628.12, Sharpe Ratio: 1.04, Number of positive days: 136.

2. Conservative Investing:

We also implemented a conservative investing plan. This means we were more risk-averse and reserve more money on hand (20%). Specifically:

- Bottom 40 stocks considered for sale
- Weight allocation: 1-day model (0.3), 3-day model (0.3), 5-day model (0.4)
- Risk parameter = 1

The results on test set:

- Profit: $223122.89, Sharpe Ratio: 0.97, Number of positive days: 138.

Together, we have:

		Profit	**Return**
Baseline Model		$134,010.00	13%
	Conservative	$223,122.89	22%
Our Model	**Aggressive**	$263,628.12	26%

Future Work

Given the statistically significant differences between the training and test periods shown by t-tests, we were pleased to find that our model and decision strategy was able to yield very positive results. Moving forward, we considered several options to further improve our model. Qualitative market data about specific companies, industries, and other market realities could be quantified and used as an input in our model. Different feature selection estimators could be tested. Additionally, further parameter tuning is an option to improve model performance.