Journal of Economics and International Business Management
Vol. 6(1), pp. 10-19, June 2018
ISSN: 2384-7328
Research Paper

Statistical arbitrage on the KOSPI 200: An exploratory analysis of classification and prediction machine learning algorithms for day trading

Ian Sutherland • Yesuk Jung* • Gunhee Lee

Department of Business Analytics, Sogang Business School, Sogang University, Seoul, South Korea.

*****Corresponding author. E-mail: karayesuk@gmail.com.

Accepted 14th May, 2018

Abstract. In this study, several machine learning methods on the representative stock market index of South Korea and the Korean Composite Stock Price Index (KOSPI) 200 were tested as machine learning has become ubiquitous in the financial field for asset selection. Compared to other major global stock markets, KOSPI has remained relatively flat over time. Despite the extremely low overall market growth, all of the tested models experienced annualized returns between 2.4 and 7.5 times the KOSPI 200 index over the same period. Even after applying an overestimated 0.5% transaction fee per daily trade the models beat out the market by a notable margin. While all tested models outperformed the market significantly, some models outcompeted the other tested models. A highlight of the present research is determining whether predicting class labels or predicting values is preferable in machine learning-driven daily trading algorithms. Four classification models - logistic regression, random forest, deep neural network, gradient-boosted trees - are compared to four prediction models - multiple regression, random forest, deep neural network, gradient-boosted regression trees. Additionally, an equally-weighted ensemble of the classification models is compared to an equally - weighted ensemble of the prediction models. Of the total of ten models, classification techniques which tend to outcompete prediction techniques by a slight margin was shown, and all models outperform the market.

Keywords: Statistical arbitrage, machine learning, random forests, gradient boosted trees, deep neural networks.

INTRODUCTION

Investors want to consistently beat the market. Whether the market is efficient or not, investors seek to capture any opportunities that they can to increase their return on investment. Recently, developments in artificial intelligence and machine learning technologies are providing effective tools for identifying signals for beating markets. These technologies have become ubiquitous in the finance industry because they are powerful, automated or semi-automated and cost efficient. There have been many proposed academic models for beating markets and many more proprietary models in use by investment firms and fund managers. While proprietary models are closely guarded secrets for the private firms utilizing them, several models in this study are adapted from the academic literature to test model performance on the KOSPI 200 constituents over a twelve-year period. More specifically, regression models, random forests, deep neural networks, gradient-boosted trees and ensemble models were tested for the purposes of comparison. Models are compared by training them to predict the following market day's out-sample adjusted closing price for each stock in the KOSPI 200 constituent list every day over the twelve-year testing period. Analogously, we train corresponding models on the same data, but employ discrete classification predictions based on the probability of whether each stock will outperform the median of the adjusted close price of all stocks during the following market day. The performance between the different types of machine learning models, as well as the performance between training each model on discrete or continuous output variables were compared.

Historically, there have been two general approaches toward investment strategies: fundamental analysis and technical analysis. Fundamental analysis arguably began with Benjamin Graham in the first half of the 19th century when he argued for the notion of buying stocks below their intrinsic value to sell after the market corrected itself (Graham and Dodd, 1934; Graham and McGowan, 2005). Since then, the body of research in fundamental analysis has become extensive, broad and effective. Most commonly, fundamental analysis strategies include some type of quantitative analysis, that is, delving into the financial statements of the company to identify growth and/or value stocks, which has been shown to be an effective strategy in practice (Gray and Carlisle, 2012).

While fundamental analysis has been shown to work, e.g. a notable proponent is Warren Buffett, the shortcomings of analysis solely based on fundamentals is clear. The accessibility to reliable, timely and full data in terms of financial ratios and other market and accounting indicators can be difficult and expensive at the very least. Furthermore, many fundamental indicators, which include book-to-market ratio and price-to-earnings ratio, are often only provided annually for each company, making them insufficient on their own for short-term trading.

Alternatively, technical analysis is based on the premise that prices lead the market and drive fundamentals. Therefore, past prices drive current and future prices. This is the crux of momentum-based stock trading strategies, and its effectiveness can be attributed to predictable behavioral tendencies of investors (Hong and Stein, 1999; Gray and Vogel, 2016). Generic momentum is calculated by taking the product of gross returns for twelve months and subtracting one, while a more common approach is to calculate generic momentum whilst omitting the most recent month (Asness et al., 2013; Gray and Vogel, 2016). In practice, such simple momentum strategies have been replaced by more complicated and cutting-edge techniques.

Statistical arbitrage models are characterized by systematic trading signals, market neutral trading book used for asset selection, and statistical mechanisms driving the generation of excess returns (Thomaidis et al., 2006; Avellaneda and Lee, 2008). One common type of statistical arbitrage model is pairs trading, which came about around the same time in the mid-1980s and identifies a pair of stocks that have correlated price movements over time in order to enter into spread bets when those prices diverge by a certain amount. Due to the high correlation in price movements of the past, the prices are likely to re-converge at some point in the future at time which the investor exits the bet. Therefore, the driving force of the success of pairs trading techniques is based on mean-reversion. There is a large breadth of academic work on pairs trading to determine how to identify trading pairs, and when to enter and exit spread bets (Bogomolov, 2013; Do and Faff, 2010; Huck, 2009; Jacobs and Weber, 2015; Lei and Xu, 2015; Song and Zhang, 2013). Cointegration is an approach developed by the Nobel laureates (Engle and Granger, 1987) and has been applied to pairs trading wherein the non-stationary time series of two assets can become stationary through a linear combination of the two. This approach has been applied in order to identify pairs and set rules for when to enter and exit trades as well (Chiu and Wong, 2015; Clegg and Krauss, 2018; Krauss et al., 2017).

As the paradigm of the financial field has shifted towards computer driven algorithmic trading, the underlying foundations of the models are based on technical analysis, fundamental analysis or a hybrid of both. Atsalakis and Valavanis (2009) analyzed 100 academic papers on computing techniques that forecast financial markets or assets and found that neural networks, time series models, genetic algorithms and other machine learning techniques were among the most popular, in which over one third of the models utilized input variables based on prices or some derivative of asset prices. Since then, there have been many new developments.

The same mean-reversion principle that makes pairs trading techniques viable has been utilized by statistical arbitrage models using machine learning and artificial intelligence for asset selection. Where pairs trading usually bets that the pair of assets will revert to a mean, statistical arbitrage models often rely that the trading book of assets will revert to a mean, and those assets moving toward or away from that mean can be identified through statistical signals. Neural networks and other machine learning algorithms are common and effective techniques that identify the statistical signals that generate statistical arbitrage, and these machine learning techniques have been expanded on in several notable academic papers (Burgess, 2000; Dixon et al., 2015; Enke and Thawornwong, 2005; Huang et al. 2005; Sermpinis et al., 2013). Tsai et al. (2011) combined classifier machine learning algorithms into an ensemble that outperformed any single classifier to predict asset prices. Takeuchi and Lee (2013) developed another statistical arbitrage model using stacked restricted Boltzmann machines that made particularly clever use of preprocessing asset prices into time lags spanning over every market day of the month and every month thereafter for one year. The input variable method therefore provides price movement information very concisely and was used by Krauss et al. (2017) to develop another classifier ensemble which used a diverse set of single-classifiers including a random forest, gradient-boosted tree, and deep neural network model. Krauss et al. (2017) applied their classifier ensembles on the S&P 500 to achieve annualized returns several times higher than that of the market. This classifier ensemble, along with the single-classifier models was adapted and applied to the KOSPI 200 in this study. Since the S&P 500 is a relatively high-growth index with about 9-10% annualized return, the preset study tends to corroborate the high model performance on an extremely low-growth index, the KOSPI 200, which only achieves about 2-3% annualized return.

While many models make use of machine learning classifiers, there is little research into the difference in performance between classification models and level estimation models (which are referred to as prediction models henceforth). One such study that analyzed the difference between four classification and four prediction time-series-based machine learning models found that the classification models generally outperformed the prediction models on the S&P 500, FTSE 100, and Nikkei 225 (Leung et al., 2000). Since that time, there have been many developments in machine learning techniques and very little additional research into prediction versus classification model comparisons. The aim of this study is to apply adapted models onto the KOSPI 200 and comparing corresponding classification and prediction models for four machine learning algorithms, in addition to equally weighted ensembles.

MATERIALS AND METHODS

Data

For empirical testing of selected models, the Korean Composite Stock Price Index (KOSPI) 200 were selected. The KOSPI 200 was selected for several reasons. (1) There are plentiful amounts of research on other global stock indices such as the S&P 500 and FTSE 100. (2) KOSPI 200 is a low growth market remaining relatively flat over the analysis period, with roughly only 2% annualized returns over the testing period, allowing us to test whether daily trading on a non-growth scenario can still payoff. (3) KOSPI 200 constituents have a feasibly small and computationally efficient number of daily assets for back-testing nearly a dozen separate machine learning (ML) models.

Assets' adjusted close prices were obtained for all daily constituents of the KOSPI 200 from January 4, 2000 until April 21, 2017 via FnGuide Inc. The data was split into non-overlapping training and trading sets over the period for a total of 13 training-trading set pairs. Training sets consisting of the prior three years of data were used to train each of the ML models at the beginning of the consecutive year to predict the corresponding trading set. Although infrequent, stocks with missing data during a training period were filtered out of the training-trading set pair. There is little to no survivorship bias in the methodology, since only data that would have been available at the time of trading was used. Predictions and classifications were made on a total of 3,212 market days starting from the first market day of 2004 over the roughly 200 KOSPI constituents for each day, resulting in about 642,400 predictions for each of 10 models.

Feature generation

Training sets span three years of stock prices prior to the year of the corresponding trading set. Features are generated at time *t* over one year of time lags, where lag $m \in \{\{1,2,\cdots,20\} \cup \{40,60,\cdots,240\}\}$. The time lags represent daily market lags within the month, and a time lag for each month thereafter up to one year, as per Krauss et al. (2017) and Takeuchi and Lee (2013). Thus, 31 features are generated as follows:

$$R^s_{t,m} = \frac{P^s_t}{P^s_{t-m}} - 1 \tag{1}$$

where *s* is a stock in the KOSPI 200 constituent list at time *t*.

Two types of output variables were used, one for classification models and one for prediction models. Prediction models use a simple dependent variable of the following day's expected return calculated as such:

$$Y^s_{t+1,1} = \frac{P^s_{t+1}}{P^s_t} - 1 \tag{2}$$

Classification models use the following day's expected return and classify it depending on whether the return for each stock is above or below the median of the expected return over all stocks for time *t*+1. In other words, the output is 1 for stocks expected to outperform the median return of the following day, and the output is 0 for stocks expected to underperform the median. It should be noted that the output of the classification models is a probability that the stock outperforms the median at *t*+1. The training data uses the realized *t*+1 returns to train the models, since this data would be available at the time of training the model prior to the trading set. However, the trading set predicts the *t*+1 values as if they were unknown, since this would in fact be the case in the simulated scenario at every time of *t*.

Stock rankings and trades

For prediction models, a ranking of stocks at each time *t* is extracted for each model sorted by those stocks with the highest predicted return for time *t*+1 to the lowest predicted return. In the classification models, stocks are sorted for each model by their probability to outperform the median, wherein stocks with the highest probability are ranked higher and the stocks with a lower probability to outperform the median are ranked lower. For each time t, the top ranked k stocks are longed for one day, meaning that they are bought at the adjusted closing price of time t and sold at the adjusted closing price at time t+1. This assumes full liquidity of all stocks, and thus friction costs are assumed to be zero. The concentration of k is tested for $k \in \{5,10,15,\cdots, 50\}$. It is expected that higher concentrations of stocks, that is, lower k, will result in relatively higher returns, while more diluted concentrations of stocks, that is, higher k, will result in more stable returns. Furthermore, an excessive transaction fee of 0.5% is incurred for each trade for every stock. While 0.5% is an over-estimation since 0.35% is more realistic, the over-estimated 0.5% is used to show a conservative trading scenario.

Software

The statistical programming language R was used for data preprocessing, data handling and modeling. The package *xts* (Ryan and Ulrich, 2017a) was used for time series partitioning, *quantmod* (Ryan and Ulrich, 2017b) was used to get KOSPI 200 index data and *PortfolioAnalytics* (Peterson and Carl, 2015) for financial statistical analysis. For modeling, the open source H2O software was called from the *h2o* package in R to develop the deep neural networks (Arora et al., 2015) and gradient-boosted trees (Click et al., 2015). Specifically, gradient-boosted trees use the XGBoost algorithm (Chen et al., 2018) via the *h2o* package in R (The H2O.ai Team, 2017). Random forests were trained using the *randomForest* R package (Liaw and Wiener, 2002).

Model specification

Models were trained annually on the first market day of January for the trading sets beginning in 2004 and ending in 2017. Model predictions for each trading set were saved and aggregated for out-sample analysis.

Logistic and multiple regression

Regression models are a broad set of models that often use statistical methods to fit to a feature space. Both the logistic regression model (LGT) and the multiple regression model (REG) were trained using all variables. While the base assumptions of regression are breached due to autocorrelation of the features and high multi-collinearity, we assess the model performances on the ability to identify stocks that will increase in value the most, rather than assessing based on model fit. As such, there are no other parameters set for the regression models and basic statistical approach to compare to the more complex machine learning models used. Our expectation for the models is low based on their limitations of dealing with non-linear relationships between the features and the dependent variable. The logistic regression model gives the natural log of the odds that $Y^{s}_{t+1,1}$ for stock s at time t will be above the median adjusted close price for all stocks in the constituent list the following market day, time $t+1$. It is formulated as follows:

$$\ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{31} x_{31} \quad (3)$$

where β_0 is the intercept and β_m is the coefficient to the m^{th} input variable and m is 1 to 31

The multiple regression model is used for the corresponding prediction model to the classification model - logistic regression. The multiple regression model predicts the expected adjusted close price for stock s at time t+1. The multiple regression model is formulated as follows:

$$Y^{s}_{t+1,1} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{31} x_{31} \quad (4)$$

where β_0 is the intercept and β_m is the coefficient to the m^{th} input variable and m is 1 to 31

Random forests

Random forests are a bootstrap aggregating (bagging) tree-based machine learning method. Bootstrap aggregating works by randomly sampling and resampling observations in the training set and aggregating the results. For the number of trees in the random forest, a separate randomly sampled subset with replacement of the training data is created for each tree in the forest. By randomly subsampling observations of the training dataset for each tree in the forest, overfitting to the training set is averted while maintaining the underlying characteristics of the data. Furthermore, at each node in the tree, only a random subset of variables is considered to make the splitting decision, which de-correlates the trees. Since random forests are a non-parametric method, there are no underlying assumptions about the distribution of data.

The classification random forest (RAF_{class}) and prediction random forest (RAF_{pred}) were trained after 2 tuning criteria were specified - (1) the number of trees, B; and (2) the subsampling parameter, m. The random forest algorithm creates B trees to their maximum depth and aggregates them for a majority vote. In order to mitigate the overfitting problem that are inherent in decision trees, random forests subsample m features randomly at each split for every tree in the forest. We set

B to 500 as to minimize computation time and not overfit the training data by oversampling. The subsampling parameter *m* differs between the RAF$_{class}$ model and RAF$_{pred}$ model, as per Breiman (2001). In the RAF$_{class}$ model, *m* is set as the floor of the square root of the *p* features, where *p* is 31, so *m* is 5. In the RAF$_{pred}$ model, *m* is set as the floor of *p* divided by 3, where *p* is 31, so *m* is 10.

Deep neural networks

Neural networks are a broad set of algorithms with the goal of identifying and interpreting signals via multiple interconnected nodes, roughly modeled on the way neurons identify and interpret signals in the human brain. Deep neural networks are a class of 'stacked neural networks', meaning that multiple layers of nodes are stacked on top of one another. Deep neural networks are effective at identifying complex relationships between the input features and dependent variable. The classification deep neural network (DNN$_{class}$) and prediction deep neural network (DNN$_{pred}$) are feed-forward multilayer artificial neural networks trained using stochastic gradient descent via back-propogation. As a multilayer perceptron model, the node structure is set with three hidden layers with 31 input nodes, one for each feature, the first hidden layer with 31 nodes, the second hidden layer with 31 nodes, the third hidden layer with 31 nodes and an output layer of two nodes for the classification model and single output node for the prediction model (that is, I31-H31-H31-H31-O2 for classification; I31-H31-H31-H31-O1 for prediction). Thus, the number of nodes remains constant and relatively large to detect sensitive signals in the feature space as the inputs are fed forward to either a classifier or prediction output.

All inputs are normalized with a minimum of 0 and maximum of 1. Stopping parameters for training were set to cease model training on the deep neural networks for whichever came first, either 1000 epochs or the stopping threshold of an improvement on the training set MSE of 10^{-6} for DNN$_{pred}$ or a misclassification error rate improvement equal to or lower than zero on the training data for DNN$_{class}$. All models stopped within 5 to 20 epochs of training, which is sufficient for convergence of the model and yet efficient enough for relatively short computation times. In order to minimize overfitting to the training sets, the input dropout rate was set to 0.1, dropout rate for hidden layers set at 0.5 and L1 regularization was set at 10^{-5}. The activation function used was 'maxout with dropout'.

Gradient-boosted trees

Gradient-boosting is a boosting technique that converts weak learners, in our case shallow decision trees, into strong learners. *B* learner trees are generated from the training data by iteratively generating tree upon the error of the previously iterated tree. The sequential iterations of trees are trained on the residuals of the previous iteration, and hence the learner tree's classification or prediction rules are updated to minimize a loss function. The Extreme Gradient Boosting (XGBoost) algorithm was utilized due to its power, flexibility and efficiency.

The classification gradient-boosted trees model (GBT$_{class}$) and prediction gradient-boosted trees model (GBT$_{pred}$) are successively fit shallow decision trees, where each successive tree builds on correcting the respective classification error or regression error from the previously iterated tree to build a strong ensemble of weak learning trees. This study subsamples the gradient boosted trees by variables, using half of the available feature space for each tree. Input features were standardized with a minimum of 0 and maximum of 1. To maintain the shallow depths of trees, maximum tree depth was set to 3 to avoid over fitting. Shrinkage on the magnitude of each gradient step is used to slowly converge the model towards the observed values in the training data in the form of the learning rate, which is set at 0.1. The number of 100 trees per model allows for convergence of the models within a reasonably short computation time.

Ensembles

Two simple ensemble models were built using the aggregated output of other models. Ensemble models use the outputs of other models as inputs and aggregate the results to build a more robust classification or prediction. A simple soft voting ensemble (ENS$_{ssv}$) was constructed using the output probabilities from the classification models. The term *soft* in simple soft voting pertains to the use of output probability values instead of the crisp classification values. Therefore, the ENS$_{ssv}$ model uses the probability outputs for the LGT, RAF$_{class}$, DNN$_{class}$, and GBT$_{class}$ models. For each time *t*, the ENS$_{ssv}$ model averages the probabilities of the four classification models over all constituents at time *t* and outputs a new probability for each stock *s* at time *t* that it will surpass the median for all stocks the following market day. The ENS$_{ssv}$ takes each classification model's output $h_i(x)$ and averages the probability outputs as follows:

$$H^{j,s,t}(x) = \frac{1}{K}\sum_{i=1}^{K} h_i^{j,s,t}(x) \qquad (5)$$

where $H^{j,s,t}$ is the ENS$_{ssv}$ output based on the *K* classification models, and *j* is the class each stock *s* over all the constituent stocks at time *t*

The simple average ensemble model (ENS$_{avg}$) takes a simple average of all prediction models' outputs $h_i(x)$. The

ENS_{avg} uses the prediction outputs of the REG, RAF_{pred}, DNN_{pred}, and GBT_{pred} models as inputs in order to combine a simple average of the model predictions for each stock *s* at time *t* formulated as follows:

$$H^{s,t}(x) = \frac{1}{K}\sum_{i=1}^{K} h_i^{s,t}(x) \tag{6}$$

where $H^{s,t}$ is the ENS_{avg} output based on the *K* prediction models, and *s* is each stock over all the constituent stocks at time *t.*

RESULTS

Classification models outperform their prediction model counterparts in terms of annualized returns at most levels of asset concentration, with some notable exceptions in the ensemble models and gradient-boosted trees. Annualized downside deviations are nearly identical between all models at the same asset concentrations, wherein the deviations become more stable as the number of assets increases in the portfolio. Figure 1 displays the annualized returns for each model, as well as the annualized downside deviations of returns over ten levels of asset portfolio concentrations. The cell background with the minimum performance is blackened while the maximum performing model-portfolio concentration combination is colored with a white background. All models in between are a different shade of gray corresponding to their performance relative to the minimum and maximum, that is, lower performances have a darker background and vice versa.

For simplicity and clarity, we set the number of assets at 50 for subsequent model comparisons. It is important to note that the best single models in terms of both annualized returns and the highest Sharpe Ratios are the random forest models using 10 assets, and the best average annualized return at any tested asset concentration is 15 assets. Despite the higher performance at higher concentrations of stocks, the 50-asset concentration was chosen for several reasons. First, choosing the best concentration based on post hoc model selection often has a tendency to be overfit to the data that it was tested on. Randomness in the data may result in any specific asset concentration to either over or underperform. Therefore, comparison based on the single best models or concentration with the best average performance of models is misguided, since it may be selecting on this bias. Instead, there was a focus on the rankings of models and it was found that the 50-asset concentration generally follows a similar pattern to other portfolio concentrations in terms of which models outrank other models. Most importantly, the deviation in annualized returns is most reliable at lower asset concentrations, meaning that including more assets gives a more robust estimate of the models performance. While higher portfolio concentrations with fewer stocks have better performance, for the purposes of model comparisons, a more robust estimate is preferable. Thus, the portfolio concentration at 50 assets was set for subsequent analysis and model comparison since 50 assets gives the most reliable results for generalization.

Table 1a and b compare the machine learning techniques over several metrics both prior to and after applying transaction fees, respectively. The metrics include annualized return, annualized deviation, average drawdown, downside deviation, skewness of returns, kurtosis of returns, Sharpe Ratios and Sortino Ratios. The annualized return is the percentage change in principal averaged over the number of trading sets. The annualized deviation is the deviation of the returns of trading sets averaged over the number of trading sets, since each trading set covers a one-year period. Average drawdowns (ADD) is ostensibly a measurement of the magnitude of drawdowns, and is formulated as a simple average of the magnitude of drawdowns in the period, meaning that a higher ADD corresponds to more investor dissatisfaction. Downside deviation is a better measure for risk than annualized deviation because downside deviation considers only those deviations that drop below a previously specified level of Minimum Acceptable Return (MAR). The MAR can be specified by an investor to be any investment goal that meets their expectations, so that downside deviation uses MAR as a measure from which to determine if returns are positive (that is, above MAR) or negative (that is, below MAR). This study sets MAR as 0 for simplicity and comparability. Skewness of returns is the asymmetry measure of the portfolio return distribution. A skewness of 0 describes a symmetric distribution, where a higher skewness is indicative that returns are more skewed to the right, leading to higher investor satisfaction. Kurtosis measures the degree of flatness of the distribution of returns, so a high kurtosis is indicative of more extreme values, where the normal distribution is characterized by a kurtosis of 3. Therefore, higher kurtosis is more platykurtic and thus more undesirable to investors.

Table 1a and b display the results from both before and after applying 0.5% transaction fees on each asset trade. Since trades are daily, there becomes a significant drop in performance from Table 1a to b, especially since a low portfolio concentration of 50 stocks was used. However, despite over-estimating the transaction fee at a slightly over-realized 0.5% for comparison's sake, even the lowest performing model results in annualized returns over twice that of the market. Annualized return deviations in the machine learning models exceed that of the market, which may be misleading. By analyzing the downside deviation statistics, it becomes clearer that the risk is more comparable to that of the market, as some models are slightly higher while others are slightly lower. Since the ML models generate significantly higher

Table 1a. Annualized performance of models before transaction fees.

Model		Annualized returns (%)	Annualized deviation (%)	Average drawdown (%)	Downside deviation (%)	Skewness of returns	Kurtosis of returns	Sharpe ratio	Sortino ratio
KOSPI 200 Index	BM	2.28	21.52	4.056	1.002	-0.609	7.171	1.341	1.815
Classification Models (50 assets)	LGT	12.85	21.79	3.665	0.997	-0.584	9.286	4.187	5.765
	RAF$_{class}$	12.98	21.69	3.692	0.999	-0.663	8.716	4.231	5.786
	DNN$_{class}$	14.21	21.64	3.644	0.987	-0.587	7.981	4.555	6.287
	GBT$_{class}$	10.91	21.64	4.108	0.989	-0.541	8.159	3.699	5.097
	ENS$_{ssv}$	12.28	21.65	4.203	0.988	-0.538	8.140	4.058	5.603
Prediction Models (50 assets)	REG	12.77	21.73	3.390	0.993	-0.530	8.937	4.172	5.750
	RAF$_{pred}$	12.97	21.69	4.046	0.992	-0.574	8.713	4.229	5.825
	DNN$_{pred}$	11.64	21.81	4.025	0.997	-0.529	8.863	3.871	5.337
	GBT$_{pred}$	8.95	21.68	4.788	0.996	-0.530	7.990	3.178	4.357
	ENS$_{avg}$	12.04	21.63	4.116	0.985	-0.500	8.302	3.996	5.526

*LGT, Logistic Regression; RAF, Random Forest; DNN, Deep Neural Network; GBT, Gradient-Boosted Trees; REG, Multiple Regression.

Table 1b. Annualized performance of models after transaction fees.

Model		Annualized Returns (%)	Annualized deviation (%)	Average drawdown (%)	Downside deviation (%)	Skewness of returns	Kurtosis of returns	Sharpe ratio	Sortino ratio
KIOSPI200 Index	BM	2.28	21.52	4.056	1.002	-0.609	7.171	1.341	1.815
Classification Models (50 assets)	LGT	10.15	21.80	4.110	1.003	-0.609	9.307	3.486	4.772
	RAF$_{class}$	10.31	21.70	4.077	1.006	-0.687	8.742	3.536	4.806
	DNN$_{class}$	11.49	21.65	3.952	0.994	-0.610	8.006	3.852	5.285
	GBT$_{class}$	8.27	21.65	4.713	0.996	-0.565	8.179	2.997	4.105
	ENS$_{ssv}$	9.61	21.65	4.345	0.994	-0.562	8.160	3.355	4.604
Prediction Models (50 assets)	REG	10.07	21.74	4.097	1.000	-0.555	8.951	3.470	4.754
	RAF$_{pred}$	10.30	21.70	3.659	0.999	-0.598	8.733	3.534	4.838
	DNN$_{pred}$	8.97	21.82	4.874	1.003	-0.553	8.878	3.172	4.346
	GBT$_{pred}$	6.36	21.69	5.457	1.002	-0.553	8.008	2.477	3.376
	ENS$_{avg}$	9.38	21.64	4.628	0.992	-0.523	8.321	3.295	4.529

*LGT, Logistic Regression; RAF, Random Forest; DNN, Deep Neural Network; GBT, Gradient-Boosted Trees; REG, Multiple Regression.

returns, higher risk should be expected.

It may be argued that the higher deviation compared to the market in machine learning models is justified by the high Sharpe and Sortino ratios in relation to the market. The Sharpe Ratio is reported, as it has become a relative standard to compare risk-adjusted return between portfolios, despite its crude use of standard deviations of returns. An arguably better metric for comparing portfolios is the Sortino Ratio, which measures the excess return similarly to the Sharpe Ratio but deciphers a difference between deviations of returns and those deviations of returns that fall below a specific investment goal (that is, MAR). By that measure, the classification models outperform their prediction model counterparts in every case except the random forests. Random forests are known to be resilient against scaling, so it may make little difference whether the random forest is a classification or prediction model for the purposes of this study.

In all counts, our models beat the market by a generous margin. Specifically, the deep neural networks, random forests and regressions outperformed all other models, while the ensembles and gradient-boosted trees underperformed relatively to other models. However, it is important to note that the skewness and kurtosis of returns for the ensembles and gradient-boosted tree models are remarkable compared to other models. A lower kurtosis indicating fewer outliers, and a higher skewness indicating a distribution that lends itself to higher returns is good evidence for a safer investment strategy. The nature of the gradient-boosting algorithm focuses on error reduction which makes the distinction

Table 2. Annual cumulative return statistics.

Model		Cumulative return as of period end													
		2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
KIOSPI200 Index	BM	0.00	0.02	0.51	0.52	0.89	-0.02	0.38	0.63	0.33	0.43	0.40	0.29	0.25	0.33
Classification Models (50 assets)	LGT	0.00	0.15	0.96	1.18	2.41	1.06	2.13	3.08	2.85	3.04	3.21	3.00	3.62	3.67
	RAF_{class}	0.00	0.33	1.43	1.69	2.69	1.35	2.77	3.44	2.63	3.07	3.21	3.24	3.80	3.74
	DNN_{class}	0.00	0.16	0.99	1.38	2.93	1.22	2.46	3.56	3.26	3.41	3.73	3.56	4.24	4.44
	GBT_{class}	0.00	0.07	0.74	1.03	2.11	0.82	1.84	2.60	2.17	2.29	2.33	2.19	2.62	2.74
	ENS_{ssv}	0.00	0.06	0.83	1.08	2.35	1.10	2.37	3.15	2.63	2.76	2.85	2.78	3.34	3.38
Prediction Models (50 assets)	REG	0.00	0.17	1.10	1.22	2.28	0.98	1.94	2.57	2.06	2.16	2.46	2.84	3.53	3.63
	RAF_{pred}	0.00	0.26	1.34	1.70	2.73	1.29	2.48	3.09	2.67	2.91	2.78	2.73	3.49	3.73
	DNN_{pred}	0.00	0.15	0.87	1.06	2.30	0.90	1.92	2.76	2.24	2.38	2.65	2.49	2.95	3.07
	GBT_{pred}	0.00	-0.01	0.66	0.82	1.85	0.71	1.62	2.27	1.69	1.78	1.70	1.59	1.89	1.98
	ENS_{avg}	0.00	0.08	0.84	1.08	2.25	1.01	1.88	2.51	2.13	2.21	2.55	2.47	3.07	3.26

between stock rankings less pronounced, thus, less likely to highly rank a riskier stock that might jump in price by a large margin. Similarly, the ensembles are amalgams of the other models probabilities, and thus while individual models are likely to hone in on a specific signal that might cause a risky stock to be selected for, the averaging over other models makes for a robust, yet less risky approach to stock selection.

Cumulative return in Figure 2a and b depict the relative performance of classification and prediction models, respectively, in relation to the KOSPI 200 benchmark. The growth periods, such as that leading out of the 2008 financial crisis, are the highest rallying periods for the ML algorithms with respect to the benchmark. The outperformance of the classification models over time becomes clearer when comparing the two figures. Within the classification models in Figure 2a, the deep neural network clearly outperforms the other models as time processes, while conversely, the underperformance in returns of the gradient boosted tree model over time becomes clearer. Analogously, the gradient boosted tree in Figure 2b also clearly underperforms the other prediction models.

Table 2 depicts the cumulative return of all models using the last market day in 2003 as the initial investment period. The cumulative return is calculated as the simulated returns accumulated since the initial investment period. While the prediction models contend with the classification models for the first few years, a distinct outperformance of the classification models begins, starting from 2008. In nearly every case after 2008, the classification models outperform their prediction counterpart. We surmise that there are two main reasons for this. First, over short time horizons, subtleties have a strong effect on the performance of the models, while longer time horizons give more reliability in the performance outcomes. Secondly, the classification models outperform particularly during periods of growth, and therefore, the bull market following the 2008 financial crisis leads the classification models to a rise in performance that the prediction models cannot overtake.

DISCUSSION

The outperformance of classification models over prediction models corroborates the findings of Leung et al. (2000). Leung et al. compared level predictions and classification models on the S&P 500, FTSE 100, and Nikkei 225. Their research concluded that the classification models are superior to level estimation techniques by the measures of both predicting stock price movement direction and by simulated returns.

Using machine learning models trained on price lags was also utilized by Krauss et al. (2017) to simulate trades on selected stocks. Testing similar machine learning models on the Korean KOSPI 200, this technique yielded high returns for this study. In line with these results, Krauss et al. found that their model beat the S&P 500 index by large margins as well. Their research found that using asset concentrations of 15 stocks yielded the highest return, and diluting the portfolio concentration resulted in lower returns, consistent with our results.

Whereas Krauss et al. (2017) found that the equally-weighted ensemble performed the highest returns on the S&P 500, research on the KOSPI 200 revealed otherwise. While ensemble results were by no means bad on the KOSPI 200, the ensembles did not outperform all other models. It is expected that this result is due to the averaging of probabilities on a smaller set of assets, that is, 200 rather than 500. The discrimination between the assets in the KOSPI 200 is not very large, so the averaging results in very similar results in the ensemble.

It is expected that the model used by Krauss et al. (2017) had more substantial differences between probabilities of the classifier models, which would result in a more distinctive advantage of using an ensemble. Therefore, using a larger book of assets to train on appears to have a higher benefit for the ensemble results.

CONCLUSION

Features generated using time lags of adjusted closing prices are effective at selecting the daily winners of the KOSPI 200 constituents. Using several machine learning algorithms trained only on daily adjusted closing price data, the machine learning models achieved annual returns several times higher than the KOSPI 200 Index. Despite the extreme low-growth nature of the KOSPI 200, the machine learning models still obtained returns rivaling those of the high growth S&P500 over the same period. The methodology therefore is shown to be an effective StatArb methodology even in an extremely low-growth market using only limited data - daily adjusted close prices only.

By comparing the performance of the classification models with their prediction model counterparts, it becomes apparent that the classification models edge out the prediction models by a small, but consistent margin. Since the prediction models are ranking stocks based on point predictions and do not incorporate the uncertainty in the point estimates, there is a high level of noise that is likely confounding the ranking of stocks per day. Classification models are ranked on probabilities of outperformance which indirectly incorporates the reliability of the estimates being above a given point - in our case, that point is the median of the following days' expected returns.

Classification models outperform their corresponding prediction models at most portfolio concentrations with few exceptions in terms of annualized return, however the annualized deviations are nearly identical between analogous models. In fact, the annualized deviations are higher than that of the benchmark at most tested portfolio concentrations, yet downside deviation is not notably different from the benchmark. This is indicative that the machine learning models have higher upside deviation than the market while not substantially improving on the downside deviation. Thus, lending evidence for a need of better portfolio weightings after asset selection.

In terms of ranking the models by risk-adjusted return, the gradient-boosted trees were the lowest performing models with the lowest average returns, despite the encouraging distributions of those returns with relatively low kurtosis and high skewness. The ensembles performed better than the gradient-boosted tree models, but since both ensembles are based on the averaging of other models, their output is not impressive in terms of returns but did manage to minimize the downside deviation slightly better than nearly all other models. Deep neural networks performed very well for classification, and moderately well for prediction. The classification deep neural network outperformed all models in terms of annualized returns and lower downside deviation, and was reliable in maintaining high performance over a wide range of asset concentrations. There is some contention in the regression and random forest models but the random forests tend to outperform the regression models at most portfolio concentrations. Random forests remain robust to whether the dependent variable is a classification or value prediction.

LIMITATIONS AND FUTURE STUDY

This study focused on the comparison of machine learning techniques for asset selection on the KOSPI 200 constituents over an out-sample period of twelve years. The focus on asset selections in the specified market leaves several notable limitations, while the results indicate several questions for further study. By using the KOSPI 200 as the target market, there are limitations in generalizability to other markets. We expect that machine learning models such as the DNNs would perform better in less homogenous markets, such as the KOSDAQ, while the performances of more simplistic models like the regressions would suffer in such markets. While extant studies have shown the effectiveness of similar models on the S&P500 and Nikkei for example, more research is needed to corroborate the findings in other markets.

Similarly, using the KOSPI 200 is limited in terms of the asset type. An equity market was chosen due to the high liquidity and generally high returns; however, the models were not tested on other asset types nor other transaction types, such as shorting. There is an expectation that these models can be adapted to other asset types with relatively minor adjustments. It is important to keep in mind that the models should be considered as a ceiling on performance since they do not consider friction costs, realized transaction costs or other associated fees. While Table 1b attempts to estimate a realistic transaction cost, it is not a realized transaction cost, so real results may differ. Furthermore, using the daily adjusted close price is a proxy for what daily returns are expected to be. However, critical investors buying or selling extremely large amounts of shares using such techniques may have an impact on the prices of those assets and create a positive or negative bias in actuality. A similar effect can occur from many investors using related algorithms, which can potentially result in a positive or negative feedback loops as can be seen with high frequency trading algorithms that cause flash crashes occasionally.

Finally, results may improve with better training of the models and by introducing asset weight allocation techniques to the selected stocks, which is typical in the field. Models for this study were trained using reasonable assumptions about parameter estimations informed by previous research and extant literature. The chosen models were selected based on their diversity from one another and intended to be relatively simple for purposes of generalization of the results. It is expected that variants on the presented models offer higher performance, such as improving the ensemble models by using weighted soft voting or weighted averaging based on the other models' performances. Asset weight optimization should also be executed on the backend of the model, such as mean-variance or CVaR optimizations. The potential improvements for weight optimization is expected to be significant, particularly in terms of minimizing the downside risk, since the main goal of training the machine learning algorithms was simply to maximize return. The results of this research will help push the understanding of asset selection models using machine learning algorithms ever so slightly forward in hopes that it will help both academics and practitioners in the field.

ACKNOWLEDGEMENTS

This research was supported by the Ministry of Science and ICT (MSIT) Korea, under the Information Technology Research Center (ITRC) support program (IITP-2018-2015-0-00369) supervised by the Institute for Information & Communications Technology Promotion (IITP).

REFERENCES

Arora A, Candel A, Lanford J, LeDell E, Parmar V (2015). Deep Learning with H2O. http://h2o.ai/resources.

Asness CS, Moskowitz TJ, Pedersen LH (2013). Value and momentum everywhere. J. Finan. 68(3):929-985.

Atsalakis GS, Valavanis KP (2009). Surveying stock market forecasting techniques - Part II: Soft computing methods. Exp. Syst. Appl. 36(3):5932-5941.

Avellaneda M, Lee JH (2010). Statistical arbitrage in the US equities market. Quantitative Finan. 10(7):761-782.

Bogomolov T (2013). Pairs trading based on statistical variability of the spread process. Quant. Finan. 13(9):1411-1430.

Breiman L (2001). Random forests. Mach. Learn. 45(1):5-32.

Burgess AN (2000). A computational methodology for modelling the dynamics of statistical arbitrage (Doctoral dissertation, University of London).

Chiu MC, Wong HY (2015). Dynamic cointegrated pairs trading: mean - variance time-consistent strategies. J. Comput. Appl. Math. 290:516-534.

Chen T, He T, Benesty M, Khotilovich V, Tang Y (2018). xgboost: Extreme Gradient Boosting. R package version 0.6.4.1. https://CRAN.R-project.org/package=xgboost.

Clegg M, Krauss C (2018). Pairs trading with partial cointegration. Quant. Finan. 18(1):121-138.

Click C, Lanford J, Malahlava M, Parmar V, Roark H (2015). Gradient Boosted Models with H20. http://h2o.ai/resources.

Dixon M, Klabjan D, Bang JH (2015). Implementing deep neural networks for financial market prediction on the Intel Xeon Phi. In Proceedings of the 8th Workshop on High Performance Computational Finance ACM. p. 6.

Do B, Faff R (2010). Does simple pairs trading still work?. Finan. Anal. J. 66(4):83-95.

Engle R, Granger C (1987). Co-Integration and Error Correction: Representation, Estimation, and Testing. Econometrica, 55(2):251-276. doi: 10.2307/1913236.

Enke D, Thawornwong S (2005). The use of data mining and neural networks for forecasting stock market returns. Exp. Syst. Appl. 29(4):927-940.

Graham B, Dodd DL (1934). Security analysis: Principles and technique. McGraw-Hill.

Graham B, McGowan B (2005). The intelligent investor. Harper Collins.

Gray WR, Carlisle TE (2012). Quantitative Value, + Web Site: A Practitioner's Guide to Automating Intelligent Investment and Eliminating Behavioral Errors (Vol. 836). John Wiley & Sons.

Gray WR, Vogel JR (2016). Quantitative Momentum, + Web Site: A Practitioner's Guide to Building a Momentum-Based Stock Selection System. John Wiley & Sons.

Hong H, Stein JC (1999). A unified theory of underreaction, momentum trading, and overreaction in asset markets. J. Finan. 54(6):2143-2184.

Huck N (2009). Pairs selection and outranking: An application to the S&P 100 index. Eur. J. Operat. Res. 196(2):819-825.

Jacobs H, Weber M (2015). On the determinants of pairs trading profitability. Journal of Financial Markets, 23, 75-97.

Krauss C (2017). Statistical arbitrage pairs trading strategies: Review and outlook. J. Econ. Surv. 31(2):513-545.

Krauss C, Do XA, Huck N (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. Eur. J. Operat. Res. 259(2):689-702.

Lei Y, Xu J (2015). Costly arbitrage through pairs trading. J. Econ. Dyn. Control, 56:1-19.

Leung MT, Daouk H, Chen AS (2000). Forecasting stock indices: a comparison of classification and level estimation models. Int. J. Forecast. 16(2):173-190.

Liaw A, Wiener M (2002). Classification and Regression by random Forest. R News 2(3):18-22.

Peterson BG, Carl P (2015). Portfolio Analytics: Portfolio Analysis, Including Numerical Methods for Optimization of Portfolios. R package version 1.0.3636. https://CRAN.R-project.org/ package= Portfolio Analytics.

Ryan JA, Ulrich J (2017). xts: extensible Time Series. R package version 0.10-1. https://CRAN.R-project.org/package=xts.

Ryan JA, Ulrich J (2017). quantmod: Quantitative Financial Modeling Framework. R package version 0.4-12. https://CRAN.R-project.org/package=quantmod.

Sermpinis G, Theofilatos K, Karathanasopoulos A, Georgopoulos EF, Dunis C (2013). Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization. Eur. J. Operat. Res. 225(3):528-540.

Song Q, Zhang Q (2013). An optimal pairs-trading rule. Automatica, 49(10):3007-3014.

Takeuchi L, Lee YYA (2013). Applying deep learning to enhance momentum trading strategies in stocks. In Technical Report. Stanford University.

Thomaidis NS, Kondakis N, Dounias GD (2006). An intelligent statistical arbitrage trading system. In Hellenic Conference on Artificial Intelligence (pp. 596-599). Springer, Berlin, Heidelberg.

Tsai CF, Lin YC, Yen DC, Chen YM (2011). Predicting stock returns by classifier ensembles. Appl. Soft Comput. 11(2):2452-2459.