머신 러닝 모형을 통한
한국 주식 시장에서의 자산가격 연구

leeway00

초록

주요어 : 머신 러닝, 금융, 주식
학 번 : 2016-11612

본 연구는 금융, 특히 주식 시장에서 활용되고 있는 다양한 머신 러닝 모형들을 비교, 분석했다. 2010년에서 2021년간의 Out-of-sample에서 모형을 평가했다. 우선, 모형이 비선형적인 특성을 잘 반영할수록 좋은 예측력 및 설명력을 보여주는지를 보였다. 둘째로, 머신 러닝 모형이 통해 포트폴리오 선택과 같은 실용적인 문제에서의 활용 가능성을 보였다. 마지막으로, 과적합의 문제와 관련하여 머신 러닝 모형을 활용한 연구의 한계점을 밝혔다.

목차

1. 서론

본 연구는 금융, 특히 주식 시장에서 활용되고 있는 다양한 머신 러닝 모형들을 비교, 분석했다. 이를 통해 머신 러닝 및 딥 러닝 모형들의 금융경제학적인 측면에서 주가 수익률에 대한 위험 프리미엄을 기존 모형들보다 더 잘 드러내는지를 보였다.

또한 주가 수익률에 대한 위험 프리미엄을 측정하는 머신 러닝 모형들의 예측 정확도를 보이고, 머신 러닝 예측을 토대로 한 포트폴리오를 통해 머신 러닝 모형을 활용한 포트폴리오 전략이 단순한 선형 모형을 통한 예측 및 포트폴리오 전략보다 높은 수익률을 가져다줌을 보이고자 한다.

인공 지능 알고리즘과 컴퓨팅 기술의 발달에 힘입어 금융에서도 복잡한 머신 러닝 및 딥러닝 모형을 활용하여 투자의사 결정을 내리는 시장참여자들이 증가하고 있다. 이러한 투자는 퀀트 투자의 일종이며, 퀀트 투자란 일반적으로 컴퓨터의 계량 분석을 기반으로 투자 대상을 찾아내는 전략들을 일컫는다. 일례로 미국 헤지 펀드 업계에서는 퀀트 펀드의 운용자산 규모가 2009년부터 2017년까지 8년간 두 배 넘게 불어나며 총 9,620억 달러에 이를 정도로 최근 급성장했다.[1] 국내 금융 산업에서도 이러한 추세에 따라 고빈도 거래 및 알고리즘 트레이딩이 최근 주목받으면서 여러 퀀트펀드가 성장하고, 대형 증권사들이 프랍 트레이딩에서 퀀트팀을 운용하고 있으며, IT 기업들이 금융 산업에 뛰어들고 있다.[2]

위와 같이 한국 시장에서의 핀테크 및 머신 러닝 모형들에 관한 관심이 증가하는 상황에서, 본 연구에서는 머신 러닝 모형들을 비교 분석하여 한국 주식 시장에서의 예측력을 분석했다. 본 연구에서 활용한 머신 러닝 모형들은 이러한 인공지능 알고리즘 중 가장 보편적으로 알려져 있고, 가장 간단한 형태의 모형들이다.

확률적인 데이터에 기반한 머신 러닝 모형 연구의 필요에 따라 최근 컴퓨터공학에서도 주가 수익률 예측 모형을 연구하는 추세가 두드러지게 나타나고 있다. 이러한 컴퓨터 공학의 연구 또한 단순히 공학적으로 복잡한 모형을

1) 김재현, “미국에서 뜨거운 '퀀트' 주식투자 열기…마법공식 뭘까”, 머니투데이, 2019.05.25
2) 양선우, “단타매매로 유명세 탄 알고리즘 헤지펀드…금융사 "퀀트 인재 어디 없나요?”, 인베스트 조선, 2019.07.08

만들기보다는 금융시장과 관련된 아이디어를 바탕으로 모형을 제시하려 하고 있는데, Ryo Akita(2016)에서는 회사에 관련된 금융 뉴스데이터를 기반으로 sentiment 분석을 진행하여 주식의 미래 트렌드를 예측했고, Jaemin Yoo(2021)에서는 Attention LSTM을 사용하여 주식 간의 상관관계를 활용하여 주식 수익률을 예측하고 포트폴리오를 만들어 시장 포트폴리오보다 높은 수익률을 보였다. Chi Chen(2019)에서는 주식시장의 참여자들의 행위를 바탕으로 주가 예측을 진행했다.

위의 컴퓨터 공학 분야의 연구들이 머신 러닝 모형을 어떻게 구성하는지에 집중했던 반면, 금융경제학의 영역의 자산가격 연구에서는 이러한 머신 러닝 및 딥러닝 모형을 활용한 주가 예측 모형들이 위험 프리미엄을 설명하는지 또는 시장 참여자인 투자자들의 투자 의사결정에 어떠한 영향을 미치는지를 중심으로 연구가 진행되고 있다.

먼저 Kelly, and Xiu(2020)에서는, 미국 주식 데이터를 바탕으로 머신 러닝 모형을 비교함으로써, 머신 러닝 예측을 통해 높은 퍼포먼스를 보였고, 이러한 예측이 설명변수들의 비선형적인 영향을 반영하기 때문이라고 설명한다. 구체적으로는, 전체 시장과 개발 주식에 대한 위험 프리미엄을 측정하는 머신 러닝 모형들의 예측 정확도를 보이고, 머신 러닝 예측을 토대로 한 포트폴리오를 통해 높은 머신 러닝 전략이 높은 수익률을 가져다줌을 보였다.

자산가격 연구에서의 머신러닝 사용에 대한 Gu, Kelly, and Xiu(2020)의 입장은, 많은 자산가격결정 모형들이 설명력을 개선하기 위해 제시되었지만, 기대수익률의 횡단면 분포를 성공적으로 설명한 모형이 없다는 채준(2007)과도 보완적인 맥락을 가지고 있다. 채준(2007)에서는 모든 모형이 완전하지 못하기 때문에 항상 그 모형들이 예측하는 수익률과 실제수익률과의 차이가 존재한다고 설명한다. 이러한 문제는 Gu, Kelly, and Xiu(2020)에서 위험 프리미엄의 측정이 근본적으로는 예측의 영역이기에 머신 러닝 모형을 활용하는 것이 이상적이라는 주장으로 해소된다.

위의 Gu, Kelly, and Xi(2020)의 연구 결과와 유사하게 본 연구에서는 한국 데이터를 바탕으로, 예측 변수들 간의 비선형적인 관계가 머신 러닝 모형들에서 반영되어 예측의 문제에서 성능 향상이 이루어지는지 확인했다.

유사하게 최근 Leippold, Wang, and Zhou(2021)에서는 중국 시장에서의 머신 러닝 모형의 성과를 비교분석하고 변수들 간의 중요도를 평가했다. 머신 러닝 모형을 사용한 이유에 대해서는, 중국이 일련의 구조적 붕괴, 다양한 금융 개혁, 자본시장 개방 확대 등을 통해 고도로 역동적인 발전을 경험하고 있다는 점을 감안할 때 중국 시장의 특수성을 설명하기 위해서는 고도로 유연한 방법이 필요하다고 주장하고 있다.

추가적으로, Bianchi, Büchner and Tamoni(2020)에서는, 채권의 위험 프리미엄 예측에 머신 러닝을 적용했다. 구체적으로는, 깊이가 극단적으로 깊은 Decision Tree나 Neural Net을 사용하는 것이 독립변수와 종속변수 간의 비선형적인 관계를 학습하여 채권의 초과수익률을 예측하는 것에 유용하다고 설명한다. 이러한 머신 러닝 모형에 변수 간의 중요도를 판단하는 알고리즘을 적용하여 경제적 관점에서, 많은 변수들 간의 비선형적인 관계가 중요한 것인지, 또는 동일한 변수의 높은 다항식이 요구되는 것인지를 알 수 있다고 언급했다.

이러한 연구들을 바탕으로, 본 연구에서는 한국 시장에서의 머신 러닝 모형들을 분석, 비교하여 위험 프리미엄을 설명하고 있는지를 드러내고, 모형이 비선형적인 특성을 잘 반영할수록 좋은 예측력을 보여주는지를 분석했다. 또한, 과적합의 문제와 관련하여 머신 러닝의 구체적인 학습 방법 및 데이터에 따라 예측력에 큰 차이가 날 수 있음을 밝혔다.

2. 머신 러닝 모형

본 장에서는 실험에 사용된 구체적인 머신 러닝 방법들을 소개한다. 본 연구에서는 Gu, Kelly, and Xiu(2020)의 방법론을 기본적으로 따랐다. 먼저, 모형의 학습은 평균 제곱 오차(Mean squared error)를 최소화하는 방향으로 진행했다.

분석 전반에 걸쳐 주식의 초과 수익률과 그에 상응하는 예측 변수 사이의 관계를 설명하기 위해 일반적인 additive prediction error model을 사용했다.

$$r_{i,t} = E_t[r_{i,t+1}] + \epsilon_{i,t+1} \tag{2,1}$$

먼저 $i = 1, \cdots, N_t$의 인덱싱은 각각 개별 주식을 뜻하며, $t = 1, \cdots, T$는 월을 뜻한다.

$$E_t[r_{i,t+1}] = g(z_{i,t}) \tag{2,2}$$

본 연구의 목표는 최고의 예측 성능을 제공하는 후보 집합에서 예측 모형을 탐색하는 것으로, $g(\bullet)$는 연구에서 사용된 예측 모형이다. 여기서 $z_{i,t}$는 예측 변수의 P차원 벡터이고, 예측 모형 $g(\bullet)$연산을 통해 산출된 값을 기간 t에서 주식 i의 초과 수익 $r_{i,t+1}$에 대한 기댓값으로 가정한다.

이러한 머신 러닝을 통한 자산가격 연구에서 사용되는 Additive prediction 형식은 기존의 자산가격 연구와 차이점을 갖는다. 먼저, $g(\bullet)$함수는 개별 주식의 상태 i나 예측 시점인 t에 의존하지 않는다. 이는 머신 러닝 모형 $g(\bullet)$의 예측이 t이전 시점의 시계열 정보나 예측하고자 하는 주식 i 외의 다른 주식의 횡단면 정보를 사용하지 않는다는 점을 의미한다. 이는 매 기간마다 횡단면 모형을 재추정하거나 각 주식에 대한 시계열 모형을 독립적으로 추정하는, 일반적인 자산가격 연구의 횡단면 또는 시계열 모형들과 대비된다. Gu, Kelly, and Xiu(2020)에서는 시간이 지남에 따라 다른 주식에 걸쳐 동일한 형태를 유지함으로써 이 모형은 전체 패널의 정보를 활용하고 개별 자

산에 대한 위험 프리미엄 추정치에 안정성을 부여한다고 설명한다.

본 연구에서 사용된 머신 러닝 모형들은 지도학습(Supervised learning) 모형과 비지도학습(Unsupervised learning)모형들로 나누어진다. 먼저, 지도학습 모형들은, 정답이 있는 데이터를 활용해 모형을 구성하는 것으로 회귀분석의 문제를 포함한다. 지도학습에서는 최적화 알고리즘을 통해서 실제값(Label)과 예측값(Predicted value)의 간격을 좁히는 방향으로 모형의 파라미터를 조정한다. 비지도학습은, 정답이 없는 데이터를 유사한 특징에 따라 군집화하여 새로운 데이터에 대한 결과를 예측하는 방법이다.

연구에서 사용된 머신 러닝 모형은 Gu, Kelly, and Xiu(2020)와 Leippold, Wang, and Zhou(2021)와 유사한 모형을 사용하였다. 구체적으로 비지도학습으로는 Partial Least Squres(PLS), PCR을 사용했고, 지도학습으로는 Ordinary Least Squares(OLS), Elastic Net, RF(Random Forest), GBRT(Gradient Boosted Regression Tree)를 사용했다. 추가로 5개의 Neural Net 모형을 활용했다.

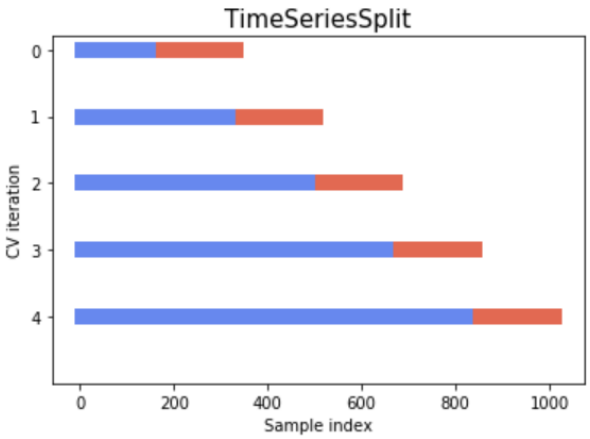
2.1. 모형 학습을 위한 데이터 분할

예측을 위한 모형을 만들기 위해서 데이터를 여러 Set으로 나누게 된다. 먼저, 모형의 파라미터를 학습시키기 위해서 Sample과 Out-of-Sample(Test set)을 나눈다. 다음은, 샘플을 Train set과 Validation Set으로 나누게 되는데, Validation set을 나누는 이유는 하이퍼파라미터의 조정 때문이다.

하이퍼파라미터는 모형의 학습 프로세스를 제어할 수 있게 하는 조정 가능한 매개 변수이다. 머신 러닝 모형의 결정은 하이퍼파라미터의 선택에 의존한다. 하이퍼파라미터의 선택에 따라 모형의 복잡성 및 학습 데이터에 대한 과적합을 조정하기 때문에 머신 러닝 모형들의 성능에 큰 영향을 미친다. 대표적인 예시로 Neural Net의 경우, 레이어의 수와 노드 수를 결정하는 데 사용될 뿐만이 아니라, Train set의 학습량(Iteraton)을 조절하여 과적합을 방지하거나, 최적화 알고리즘(Optimizer)의 하이퍼파라미터인 learning rate, decay rate 등을 결정한다.

구체적인 과정으로는, 특정 하이퍼파라미터를 가정한 후, Train set을 통해서 모형을 학습시키고, Validation set을 통해 해당 하이퍼파라미터로 구성된 모형의 결과를 산출하고 평가한다. 모든 하이퍼파라미터의 후보군으로부터 이러한 개별 학습 및 평가를 거친 다음, Validation set에서 가장 높은 평가를 갖춘 하이퍼파라미터를 선택하고, 해당 하이퍼파라미터에 기반한 모형을 최종적으로 선택하게 된다.

본 연구에서 모형 학습은 Train, Validation으로 데이터를 기간별로 나누어 진행했다. 주식 데이터가 시간 순서에 의해 드러나므로, 이후의 데이터로 학습된 모형이 이전 기간의 데이터를 예측하게 되는 것은 문제이다. 시계열 데이터인 금융 데이터의 특성 상 validation set을 만들기 위해서는 데이터를 랜덤하게 셔플하지 않고, 시간 순서에 따라서 train, validation, test set을 만들게 된다. 또한 Gu, Kelly, and Xiu(2020)와 Leippold, Wang, and Zhou(2021)과 같이 모형을 다시 학습시킬 때, Train set의 크기를 1년씩 늘려나가며 1년의 Test set을 예측했다. 같은 맥락에서, validation 과 Test set 또한 rolling basis로 다음 12개월을 포함하도록 조정되었다.



<그림 3>

2.2. 딥 러닝 모형

본 연구에서 사용된 딥 러닝 모형으로는 Fully Connected Linear layer를 사용했다. Net1은 19×16의 Linear layer를 통해 19개의 설명변수를 16차원의 latent space로 변환시킨 뒤, ReLU 함수에 통과시킨 다음, 다시 16×1차원의 Linear layer를 통해 $r_{i,t}$를 예측했다. Net2, Net3, 그리고 Net4의 경우 Gu, Kelly, and Xiu(2020)과 유사하게 Linear layer를 은닉층(Hidden Layer)에 추가하면서, 새롭게 생성되는 latent space의 차원을 8, 4, 2로 줄여나갔다. 즉, Net4의 경우, 19×16, 16×8, 8×4, 4×2, 2×1의 5개의 Fully Connected Linear Layer와 각각의 Linear layer사이에 있는 4개의 ReLU함수로 이루어져있다.

Net5는 은닉층의 차원이 변수의 개수보다 높은 경우를 상정했다. Net2에서, 32차원의 latent space를 만들기 위해 19×32, 32×16, 16×8, 8×1의 4개의 Fully Connected Linear Layer와 각각의 Linear layer사이에 있는 3개의 ReLU함수로 Net5를 구성하였다.

최적화 함수는 SGD와 Adagrad을 사용했다. Adagrad는 Neural Net의 학습과정에서 학습률을 감소시키며 학습을 진행한다. 이를 통해 데이터가 표현하고 있는 관계에 비해 learning rate가 높게 설정된 경우, 매개변수 전체의 learning rate값을 일괄적으로 낮출 수 있다. 구체적으로는 Gu, Kelly, and Xiu(2020)에서 사용한 SGD알고리즘이 계산된 기울기 경사를 그대로 매개변수에 더해가면서 모형을 학습시켰다면, Adagrad는 기울기 경사를 더해줄 때, 기울기 경사값에 제곱근 평균 제곱을 시킨 과거의 기울기를 추가로 나누어주어, 학습되는 양을 iteration에 따라 또는 누적된 학습량에 따라 낮출 수 있게 된다.

2.3. 모형 평가에 사용된 통계량

먼저 Out-of-sample에서 모형들 간의 예측력을 비교하기 위해서 Gu, Kelly, and Xiu(2020)과 Bianchi, Büchner and Tamoni(2020)과 동일한 R^2_{oos}를 사용했다.

$$R^2_{oos} = 1 - \frac{\sum_{(i,t)\in T}(r_{i,t} - \widehat{r_{i,t}})^2}{\sum_{(i,t)\in T} r^2_{i,t}} \tag{2.3}$$

위 식에서 T는 Test set에 있는 주식과 기간 상의 집합이고, $\widehat{r_{i,t}}$는 분석에 사용된 모형들의 Test set에서의 예측값을 의미한다.

또한, Gu, Kelly, and Xiu(2020)에서와 같이 머신 러닝 모형들 간의 성능을 평가하기 위해서 Diebold-Marino 검정 방법을 도입했다. 송준혁 (2011)에 따르면 해당 검정은 각 모형을 이용하여 h기간의 예측오차를 계산한 후 이를 이용해서 계산된 손실함수를 비교하는 방식으로 이루어진다. 본 연구에서는 예측오차들의 이차함수를 손실함수로 사용했고, 기간 h는 12개월을 사용했다.

3. 데이터

데이터는 한국의 주식시장 데이터로, KOSPI, KOSDAQ의 가격 및 재무 데이터를 활용하였다. 데이터의 출처는 한국거래소의 KRX 정보데이터시스템을 활용하였다. 재무데이터는 주당순이익(EPS)과 주당순자산가치(BPS)로, 연간 사업보고서를 기준으로 갱신되었다. 분석에 사용된 기간은 2000년에서 2021년 9월로, 총 21년 9개월 분의 주가 데이터가 사용되었다. 금리 데이터는 한국은행 경제통계시스템의 시장금리 추이를 사용했다. 아래 시장 팩터 계산의 과정에서 사용된 무위험 이자율로는 CD 91일물을 사용했다.

위 장에서 서술한 것처럼, 2000년에서 2010년 사이의 데이터를 Train set 및 Validation set으로, 그리고 2010~2021년의 데이터를 Test set으로 사용했다.

모형의 학습 및 예측에 사용된 팩터는 시장팩터(RM-RF), 배당수익률, 주당순이익(PER), 주당순자산가치(PBR), 시가총액(Size), Share Turnover, Momentum, Beta, Ideosyncratic Volatility, transaction turnover, 장단기 금리 스프레드, 신용 스프레드를 사용했다.

먼저, 시장 포트폴리오 수익률을 뜻하는 RM변수는, 코스피와 코스닥의 수익률을 가중평균하여 계산했다. Idiosyncratic Volatility는 각 주식의 수익률을 시장 포트폴리오 값에 회귀시켜 나오는 잔차를 활용했다. Aabo, Pantzalis and Park(2017)에 따르면 이러한 Idiosyncratic volatiltiy는 자산가격 모형으로 설명할 수 없는 수익률의 변동분을 뜻하며, mispricing을 드러낸다.

추가적인 거시 변수로 장단기 금리 스프레드와 신용 스프레드를 사용했다. 장단기 금리 스프레드는 국고채 10년물에서 국고채 3년물 간의 금리 차를 사용했고, 신용 스프레드는 회사채 3년(평균)에서 국고채 3년물 간의 금리 차를 사용했다. 단기금리는 주로 통화정책의 영향을 받는 반면, 장기 금리는 기간 프리미엄의 영향을 포함하고 있기에, 장단기 금리스프레드에는 현재의 통화정책, 미래 통화정책의 변화, 미래 경기변동에 대한 기대 등의 정보가 포함되어 있다. 신용 스프레드는 기업의 자금조달과 관련된 지표로, 기업의 재무변수와 거시경제 변수에 따른 변동을 반영한다.

4. 한국 주식시장에서의 연구 결과

먼저, 모형 간의 Out-of-sample에서의 예측결과를 비교하고, 다음으로 머신 러닝 예측을 기반으로 한 포트폴리오의 수익률을 분석하였다. 마지막으로는 Neural Net의 과적합문제와 연도별 R^2 값에 대한 분석을 다루었다.

4.1. 모형 간의 예측력 비교

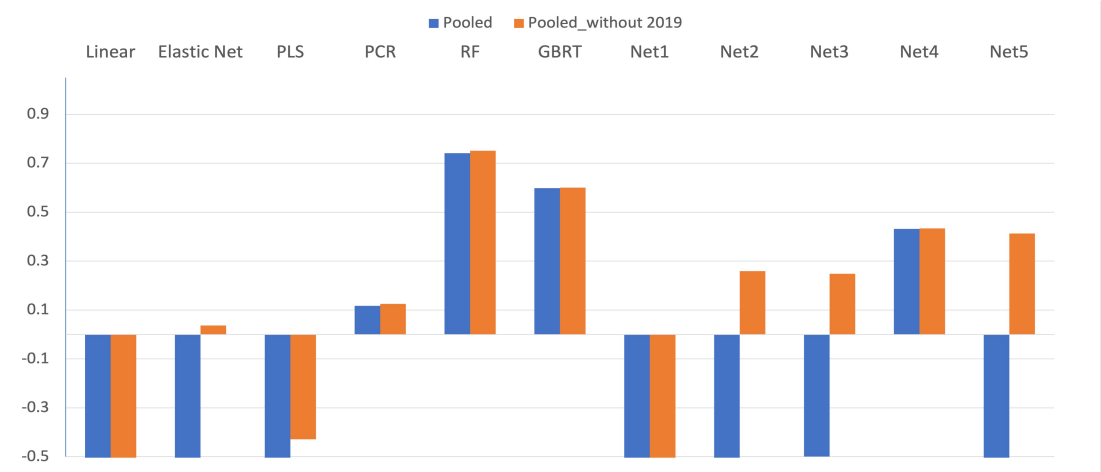
2장에서 소개된 Additive prediction error model에 따라, Pooling된 전체 셈플에서의 성능 비교와 연도별 셈플에서의 예측 가능성을 연구했다.

2019년에 대한 예측 실패가 이 비교 불가능하게 만들었기 때문에 전체 시점에 대한 Out-of-sample R^2 값과 동시에 2019년의 데이터를 제외한 R^2_{oos} 값을 계산하여 비교했다. <표 1>의 첫 번째 행의 R^2는 전체 기간의 결과값이며, 두 번째 행의 값은 2019년의 예측을 제외한 값이다.

<표 1>

	OLS	Elastic Net	PLS	PCR	RF	GBRT
Pooled	-1.2109	-1.8040	-0.5497	0.1175	0.7418	0.5982
Without 2019	-0.6992	0.0373	0.4290	0.1250	0.7514	0.6004

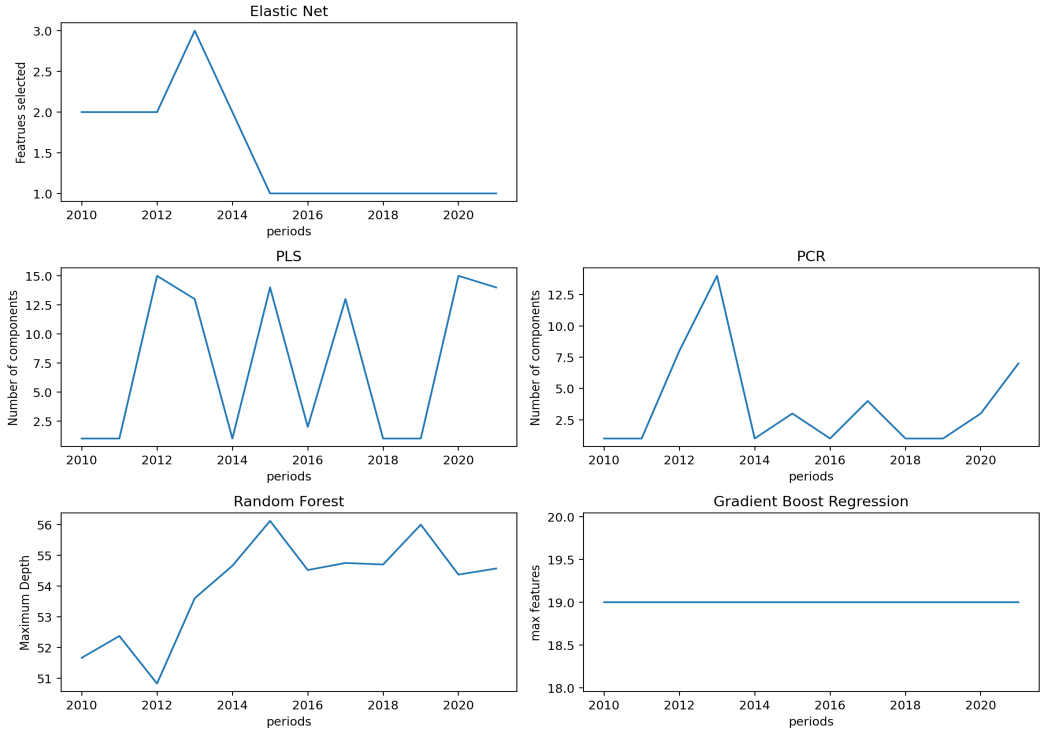
	Net1	Net2	Net3	Net4	Net5
Pooled	-271.1715	-1.8466	-0.4991	0.4317	-0.8215
Without 2019	-4.0310	0.2588	0.2482	0.4335	0.4127



먼저 OLS와 Net1은 전체 기간과 2019년이 제외된 기간 모두 R^2_{OOS}가 음의 값을 보였고, 이는 모든 주식 및 시점에서 0으로 예측하는 것에 비해 열등하다는 의미이다. 이러한 문제는 두 모형이 데이터의 비선형적인 관계를 잘 반영하지 못했음을 드러낸다. 반면에 PCR, RF, GBRT, Net4 에서는 두 셈플 모두에서 양의 R^2값을 보이며 모형의 활용가능성을 보여주었다.

둘째로는 2019년이 제외된 셈플에서는 OLS와 Net1을 제외한 모든 모형이 양의 R^2값을 보였다. 특히 Net1의 Out-of-sample R^2가 -4.03 것과, Net2가 0.358, Net3이 0.248를 보이며 충분한 Node개수가 있어야 Neural Net을 통해서 단순히 0을 예측하는 것보다 효과적인 예측을 할 수 있음을 보였다, 또한 Net4는 0.433, Net5의 0.413의 R^2값을 보였는데, 이들 값과 Net2, Net3의 R^2값을 비교함으로서, 복잡한 모형에서 자산 가격이 위험 프리미엄에 미치는 비선형적인 영향이 더 잘 드러났음을 알 수 있다.

마지막으로, 두 셈플에 걸쳐 Decision Tree의 모형들이 높은 성과를 보였다. 아래 그림5는 각 모형의 Complexity를 예측 연도에 따라 나타낸 것이다. RF의 경우 19개의 변수가 사용된 것에 비해 51에서 56사이의 높은 값의 Tree Depth가 하이퍼파라미터로 선택되었음을 알 수 있다.



<그림 4>

Random Forest의 Maximum Depth는 Random Forest가 Decision Tree의 다양한 예측을 바탕으로 앙상블(Ensemble)을 통해 최종 결과값을 보여주는 것으로부터, 전체 Decision Tree의 Maximum Depth들의 평균을 계산했다.

Elastic Net으로부터 선택된 Feature의 수는 1,2,3개였다. 이로부터 선형 회귀의 경우에서는 19개의 변수들이 위험 프리미엄에 미치는 영향이 직접적으로 드러나는 변수가 거의 없었다는 점을 알 수 있고, 이러한 이유로 낮은 R^2값을 보였음을 알 수 있다. PLS와 PCR의 Complexitiy에서는 1에서 15 사이의 n_component가 하이퍼파라미터로 선택되었던 것을 보아, 시점에 따라 예측을 위해 필요한 압축된 정보량이 달랐음을 알 수 있다.

아래 표2는 Diebold-Mariano Test의 결과물이다. 양수는 해당 열의 모형이 해당 행의 모형보다 좋은 성과를 보여주었다는 뜻이다. 값 뒤의 별표는 5% 유의수준에서 해당 차이가 유의미했음을 의미한다.

<표 2>

	Elastic Net	PLS	PCR	RF	GBRT	Net1	Net2	Net3	Net4	Net5
OLS	-0.091	1.407*	1.621*	2.316*	2.143*	0.013	0.375*	1.026*	1.959*	0.337*
Elastic Net		0.454	1.062	1.511	1.403	-1.016	-0.693	1.301	1.287	0.574
PLS			1.638*	2.554*	2.313*	-1.014	-0.577	0.634	2.081*	0.526*
PCR				3.556*	2.610*	-1.016	-0.932	-0.804	2.326*	0.868
RF					-1.790	-2.018*	-1.227	-1.600	-3.04*	-1.146
GBRT						-2.532	-1.157	-1.406	-1.270	-1.080
Net1							1.017	1.017	1.017*	1.017*
Net2								0.991	1.080*	0.145
Net3									1.209	-0.885
Net4										-1.007

표2의 결과는 ElasticNet과 Net1를 제외한 모든 모형이 OLS보다 우세했음을 보여준다. 또한 Neural Net의 경우, 더 깊은 layer가 있는 모형일수록 통계적으로 유의미한 결과값을 자주 보임을 알 수 있었다. 또한, Random Forest와 Gradient Boosting Regression Tree의 경우 선형 모형과 PLS, PCR에 비해 높은 수익을 가져다준 점이 5% 수준에서 유의미했음을 알 수 있다.

4.2. 머신 러닝 포트폴리오

본 절에서는 포트폴리오의 수익률을 예측하는 성능을 비교했다. 이러한 포트폴리오 분석은 단순히 개별 주가 수준의 분석 뿐 만이 아니라, 추가적으로 모형의 종합적인 성능을 볼 수 있다.

아래 표3은 Fama-French의 Size와 BE/ME factor를 기준으로 만든 6개의 포트폴리오와 Out-of-sample R^2 값이다. Gu, Kelly, and Xiu(2020)에서는 모든 Neural Net 모형들이 양의 값을 가졌던 것과 별개로, 본 연구 결과에서는 모든 포트폴리오에서 양의 Out-of-sample R^2를 보여준 모형은 존재하지 않았다. 대신, Random Forest와 Gradient Boosting Regression Tree의 경우 Big size 포트폴리오들에서 높은 예측력을 보여주었다. Net4 또한 양의 Out-of-sample R^2을 보여주며, 대형주들의 경우에 머신 러닝의 예측력이 존재함을 보였다.

반면 Small size 포트폴리오들에서는 모든 모형이 음의 R^2_{oos}를 보였다. 이를 통해 소형주들의 예측에서는 머신 러닝 모형들이 단순히 0으로 예측하는 것에 비해서 효과적이지 못했음을 알 수 있다. 이러한 결과의 주된 원인은 기간 및 주식을 고려하지 않고 전체 데이터를 pooling하여 머신 러닝 모형들을 학습시켰기 때문에, 모형들이 학습한 데이터와 위험 프리미엄 간의 관계가 소형주들의 위험 프리미엄을 잘 설명하지 못하기 때문이다.

<표 3>

	OLS	Elastic Net	PLS	PCR	RF	GBRT	Net1	Net2	Net3	Net4	Net5
Big Growth	-0.15	-0.34	-0.21	-1.37	0.85	0.74	-0.12	0.00	-0.18	0.08	-0.54
Big Neutral	-0.53	0.19	0.78	-3.15	0.63	0.51	0.20	-0.16	-0.55	0.50	0.06
Big Value	-0.49	-0.33	1.72	-0.41	0.87	0.86	-0.22	-0.31	-1.08	0.02	-0.32
Small Growth	-0.05	-0.04	-0.06	-0.94	-0.90	-1.85	0.00	-0.01	-0.02	-1.06	-0.01
Small Neutral	-0.22	-0.80	-0.42	-2.15	-0.17	-0.39	-0.21	-0.33	-0.21	-0.60	-0.25
Small Value	-0.08	-0.17	-0.13	-1.29	-0.08	-0.10	-0.08	-0.46	-0.38	-0.16	-0.11

아래 표 4는 머신 러닝 포트폴리오이다. 직접적으로 머신 러닝의 결과에 따른 포트폴리오들의 성과를 비교하기 위해, 각각의 머신 러닝 모형의 예측값에 따라 10분위 포트폴리오를 만들었다. 또한 가장 높게 예측된 포트폴리오(10분위)를 매수하고 가장 낮은 값의 예측으로 구성된 주식(1분위)들을 매도하는 zero net 포트폴리오를 구성했다.

<표4>

	OLS				Elastic Net				PLS			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.222	-0.059	0.158	-0.374	0.000	0.016	0.173	0.095	-0.167	-0.054	0.142	-0.379
2	-0.083	-0.035	0.122	-0.283	0.014	0.022	0.215	0.101	-0.063	-0.035	0.130	-0.268
3	-0.041	-0.021	0.103	-0.206	0.015	0.013	0.165	0.081	-0.032	-0.021	0.099	-0.214
4	-0.020	-0.016	0.111	-0.143	0.016	0.019	0.196	0.097	-0.014	-0.012	0.107	-0.111
5	-0.004	-0.009	0.116	-0.076	0.018	0.007	0.156	0.045	0.001	-0.005	0.111	-0.049
6	0.009	0.002	0.121	0.017	0.018	0.015	0.151	0.099	0.014	0.003	0.120	0.024
7	0.023	0.018	0.131	0.141	0.019	0.003	0.155	0.020	0.028	0.015	0.130	0.116
8	0.040	0.039	0.142	0.271	0.022	0.013	0.249	0.052	0.043	0.036	0.148	0.243
9	0.064	0.075	0.178	0.419	0.028	0.014	0.219	0.062	0.066	0.077	0.189	0.407
High(H)	0.280	0.190	1.117	0.170	0.053	0.062	1.062	0.058	0.285	0.180	1.119	0.161
H-L	0.251	0.124	0.800	0.155	0.026	0.023	0.762	0.030	0.226	0.117	0.800	0.146

	PCR				Random Forest				GBRT			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.126	-0.039	0.145	-0.268	-0.130	-0.117	0.146	-0.798	-0.096	-0.115	0.143	-0.805
2	-0.030	-0.012	0.120	-0.100	-0.057	-0.048	0.102	-0.465	-0.042	-0.045	0.103	-0.442
3	-0.007	-0.006	0.110	-0.057	-0.033	-0.027	0.095	-0.290	-0.024	-0.024	0.098	-0.243
4	0.006	-0.005	0.116	-0.039	-0.017	-0.014	0.091	-0.154	-0.012	-0.008	0.095	-0.084
5	0.014	-0.004	0.114	-0.035	-0.004	-0.002	0.089	-0.026	-0.002	0.002	0.092	0.018
6	0.021	0.011	0.134	0.081	0.008	0.008	0.093	0.084	0.006	0.011	0.092	0.114
7	0.029	0.019	0.157	0.122	0.021	0.021	0.099	0.210	0.016	0.022	0.100	0.218
8	0.041	0.030	0.171	0.174	0.038	0.037	0.112	0.329	0.030	0.037	0.114	0.321
9	0.060	0.050	0.215	0.234	0.065	0.065	0.134	0.487	0.053	0.062	0.137	0.455
High(H)	0.203	0.139	1.113	0.125	0.269	0.261	1.122	0.233	0.222	0.243	1.126	0.216
H-L	0.165	0.089	0.796	0.112	0.200	0.189	0.804	0.235	0.159	0.179	0.805	0.223

	Net1				Net2				Net3			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.505	-0.076	0.411	-0.186	-0.225	-0.102	0.150	-0.685	-0.161	-0.106	0.188	-0.564
2	-0.075	-0.043	0.119	-0.360	-0.079	-0.049	0.106	-0.461	-0.068	-0.048	0.105	-0.462
3	-0.040	-0.026	0.103	-0.253	-0.044	-0.025	0.102	-0.246	-0.038	-0.026	0.096	-0.273
4	-0.020	-0.011	0.096	-0.111	-0.020	-0.009	0.277	-0.033	-0.017	-0.010	0.095	-0.106
5	-0.003	0.000	0.101	0.003	-0.002	0.000	0.099	-0.001	0.000	0.000	0.097	0.003
6	0.012	0.012	0.103	0.118	0.016	0.011	0.097	0.117	0.018	0.014	0.102	0.134
7	0.030	0.025	0.111	0.225	0.033	0.024	0.107	0.224	0.036	0.026	0.110	0.233
8	0.052	0.040	0.124	0.319	0.055	0.039	0.119	0.330	0.058	0.043	0.124	0.345
9	0.086	0.064	0.153	0.419	0.091	0.071	0.155	0.454	0.090	0.069	0.153	0.452
High(H)	0.319	0.199	1.062	0.187	0.305	0.224	1.094	0.205	0.315	0.223	1.118	0.199
H-L	0.412	0.138	0.807	0.170	0.265	0.163	0.783	0.209	0.238	0.164	0.804	0.205

	Net4				Net5			
	Pred	Avg	SD	SR	Pred	Avg	SD	SR
Low(L)	-0.098	-0.072	0.358	-0.201	-0.170	-0.082	0.210	-0.392
2	-0.060	-0.041	0.377	-0.110	-0.065	-0.045	0.126	-0.358
3	-0.035	-0.026	0.109	-0.238	-0.039	-0.034	0.104	-0.327
4	-0.016	-0.011	0.109	-0.105	-0.020	-0.017	0.104	-0.163
5	0.000	0.001	0.104	0.006	-0.006	-0.003	0.095	-0.035
6	0.016	0.010	0.109	0.092	0.008	0.011	0.101	0.105
7	0.032	0.024	0.116	0.208	0.022	0.025	0.224	0.113
8	0.054	0.039	0.126	0.306	0.041	0.041	0.125	0.328
9	0.091	0.063	0.151	0.414	0.077	0.073	0.336	0.217
High(H)	0.314	0.198	1.017	0.195	0.338	0.216	1.055	0.204
H-L	0.206	0.135	0.765	0.177	0.254	0.149	0.763	0.195

수익률의 기간은 월간 예측에 대응되는 월간 수익률로 계산했다. 먼저 10-1분위수 포트폴리오에서 가장 높은 수익률을 보여준 것은 Random Forest으로 약 19%의 수익률을 보였다. 앞선 분석에서 높은 정확도를 보여주었던 Random Forest와 Gradient Boosting Regression Tree의 경우, 10-1분위수 포트폴리오의 수익률이 가장 높았을 뿐만 아니라, 다른 모형과 비교했을 때, 높은 샤프 지수값을 보였다. 이로부터 비선형 구조를 잘 반영한 머신 러닝 모형의 경우에는 위험 대비 높은 수익률을 기록 할 수 있음을 알 수 있었다.

또한 5개의 Neural Net의 경우에서도 13%에서 16% 사이의 수익률과 0.17에서 0.2 사이의 샤프지수 값을 보여주며, 선형 모형들인 OLS와 ElasticNet에 비해서 효과적이었다는 결론을 도출할 수 있었다.

4.3. 과적합 문제

머신 러닝 모형의 예측 문제는 과거의 데이터를 통해 설명변수 간의 관계를 학습한 모델이, 일어나지 않은 미래의 값을 예측하는 것이다. 따라서 효과적인 예측을 위해서는 모형이 과거의 데이터에 과적합(Overfitting)되는 것을 막아야 한다. 이 과정에서 정규화와 Validation 등의 개념이 도입된다. 하지만 과적합을 방지하는 여러 방법에도 불구하고, 특히 Neural Net에서는 Train set에 과적합되거나, 또는 설명변수의 관계를 완전히 잘못 학습하는 경우가 빈번하게 일어난다.

먼저, 표 5는 연도별, 머신 러닝 모형별 Out-of-sample R^2값이다. 이들 중에서 2019년도의 경우 대부분의 머신 러닝 모형들의 예측력이 좋지 않았음을 알 수 있는데, Neural Net의 경우 이 기간에 너무나 낮은 값의 Out-of-sample R^2값이 도출되어, 전체적으로 모형들을 비교하는 것에 어려움이 존재했다. 2019년도의 결과값을 살펴보면, 선형 모형인 OLS와 Elastic Net의 R^2_{oos}가 각각 -14.38, -35.83인 것에 비해서도, Net1은 -714, Net2는 -56.04, Net3은 -19.74, 그리고 Net5는 -59.33으로 극단적으로 열등한 예측을 보였다. 4장 1절에서 2019년도를 제외한 데이터를 통해 R^2_{oos}를 구하고 모형들을 비교분석 했던 이유는 2019년도의 예측값들이 전체 Out-of-sample

에서의 모형 간 비교를 불가능하게 만들었기 때문이다.

<표 5>

	OLS	Elastic Net	PLS	PCR	RF	GBRT	Net1	Net2	Net3	Net4	Net5
2010	-0.23	0.03	-0.07	0.11	0.90	0.67	0.24	0.32	0.27	0.45	0.26
2011	-2.35	0.00	-1.48	0.01	0.56	0.58	-1.37	0.40	0.61	0.11	0.75
2012	0.36	0.03	0.37	0.22	0.91	0.85	-2.45	-0.12	0.17	0.21	0.38
2013	-0.11	-0.03	-0.11	-0.17	0.79	0.67	0.38	0.52	0.66	0.45	0.56
2014	-0.08	-0.08	-0.01	-0.01	0.70	0.71	0.57	0.51	0.55	0.35	0.37
2015	0.33	0.11	0.33	0.31	0.93	0.88	0.93	0.72	0.64	0.89	0.66
2016	-0.29	0.01	-0.15	-0.02	0.67	0.60	0.57	0.43	0.52	0.40	0.62
2017	0.17	0.00	0.18	0.14	0.76	0.68	0.65	0.68	0.67	0.65	0.71
2018	-25.09	-0.01	-18.36	-1.27	0.51	0.44	-3.89	-0.52	0.34	0.42	-0.66
2019	-14.38	-35.83	-11.68	-0.07	0.49	0.54	-714.0	-56.04	-19.73	0.39	-59.33
2020	0.07	0.00	0.08	0.05	0.77	0.68	-10.35	-0.49	-1.04	0.08	0.57
2021	0.32	0.01	0.32	0.31	-0.37	-1.11	-3.13	-0.05	0.04	-0.05	-0.06

다음 표 6에서는 Neural Net에 사용된 최적화 알고리즘 별 R^2_{oos}를 계산한 것이다. 첫 행 Adagrad는 4장 1절에서 분석한 표 1과 동일한 결과값이다. 머신 러닝은 매개변수를 효과적으로 찾기 위해서 여러 규칙을 통해서 학습을 시작할 때 임의의 값으로부터 매개변수를 출발시킨다. seed(1)는 이러한 랜덤한 매개변수의 시작값을 난수표의 1번값에 해당하는 위치로 고정한다는 것이다. seed를 명시하지 않은 3 개의 행은 seed를 0번으로 설정하여 학습한 결과이다. 최적화 알고리즘은 Adagrad, SGD, RAdam을 사용하였다. 전체적으로 5개의 모든 모형에서 일관적인 R^2_{oos}값이 도출되는 경우는 없었다. 특히 Net1의 경우에는, RAdam에서 -62.63, SGD에서 -3989로 큰 폭의 차이를 보였다. 이러한 결과값은, 머신러닝 방법론이 설명변수의 비선형적인 관계를 모형이 잘 파악하는 경우에는 위험 프리미엄을 설명하는 효과적인 모형을 구성할 수 있지만 그렇지 않을 가능성도 충분히 존재하다는 점을 의미한다. 즉, 머신러닝의 성능에 대한 논의 자체가 다소 결과론적일 수 있다.

<표 6>

	Net1	Net2	Net3	Net4	Net5
Adagrad	-271.17	-1.85	-0.50	0.43	-1.82
seed(1)	-136.53	-0.02	0.04	0.13	0.07
SGD	-444.89	0.06	-23.20	0.23	0.14
seed(1)	-3989.84	-0.41	-11.32	-0.06	-39.00
RAdam	-62.63	-0.01	-012	-0.29	-0.21
seed(1)	-2103.63	0.42	0.38	0.38	-1.11

5. 결론

본 연구는 수익률 예측의 측면에서 머신 러닝을 활용한 주식시장에서의 자산가격 연구를 진행했다. 2010년부터 2021년 간의 Out-of-sample과 R^2_{oos}를 활용해 선형 모형들에 비해서 비선형 머신 러닝 모형들이 높은 설명력을 가짐을 보였다. 특히 Random Forest와 Gradient Boosting Regression Tree의 성과가 두드러졌는데, 사용된 변수들의 개수보다 높은 값이 Random Forest의 Maximum Depth로 선택된 것으로부터, 깊이가 깊은 머신 러닝 모형을 사용하는 것이 설명변수들의 영향을 더 잘 드러낸다는 Bianchi, Büchner, and Tamoni(2021)의 연구와도 유사한 결과를 도출할 수 있었다.

둘째로, 머신 러닝 모형의 수익률 예측력을 통해 포트폴리오 선택과 같은 실용적인 문제에서의 활용가능성을 보였다. 머신 러닝 포트폴리오를 구축하여 선형 모형에 비해 복잡한 모형을 활용한 포트폴리오들이 높은 수익률과 샤프지수를 얻을 수 있음을 알 수 있었다.

마지막으로, 머신 러닝 모형을 활용한 연구의 한계점을 밝혔다. 본 연구에서의 Random Forest와 같이 잘 학습된 머신 러닝 모형들이 좋은 성과를 거둘 수 있는 반면, 과적합으로 인해 설명변수 간의 관계를 제대로 학습하지 못한 모형들의 경우에는 단순한 선형 모형들보다 열등한 예측을 제시할 수 있음을 보였다. 특히, Neural Net의 경우에는 최적화 알고리즘과 학습 에 따라 결과가 크게 차이 날 수 있음을 보였다.

참고자료

Aabo, Pantzalis, and Part(2017): “Idiosyncratic volatility: An indicator of noise trading?,” *Journal of Banking & Finance*, **75**, 136−151

Bianchi, Büchner, and Tamoni(2021): “Bond Risk Premiums with Machine Learning,” *The Review of Financial Studeis,* **34**, **2**, 1046−1089

Chen, et al.(2019): “Investment Behaviors Can Tell What Inside: Exploring Stock Intrinsic Properties for Stock Trend Prediction,” *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2376−2384

Gu, Kelly, and Xiu(2020): “Empirical Asset Pricing via Machine Learning ,” *The Review of Financial Studies*, **33**, **5**, 2223−2273

Jaemin Yoo, et al.(2021): “Accurate Multivariate Stock Movement Prediction via Data−Axis Transformer with Multi−Level Contexts,” *KDD '21: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2037-2045

Leippold, Wang and Zhou(2021): “Machine learning in the Chinese stock market”, *Journal of Financial Economics,* DOI

Ryo Akita, et al.(2016): “Deep learning for stock prediction using numerical and textual information,” *IEEE ICIS*, 26−29 June 2016

박기영, 고정원(2019): “머신 러닝을 이용한 경제분석,” 한국경제학보, **26**, **2**, 368−408

송준혁(2011): “KOSPI200 변동성의 장기기억 모형과 국면전환 모형간 예측력 비교 분석”, 한국은행 경제연구원 「經濟分析」, **17**, **4**, 99−127

이기영(2020): “Term Spread의 분해와 구성요소의 경기예측력 분석,” 중국지역연구, **7**, **14**, 211−241

채준(2008): “Which idiosyncratic factors can explain the pricing errors from asset pricing models in the Korean stock market?”, Asia−Pacific Journal of Financial Studies, **37**, **2**, 297−342

김재현, “미국에서 뜨거운 '퀀트' 주식투자 열기…마법공식 뭘까”, 머니투데이, 2019.05.25.

서혜진, “변동장에도 흔들림없는 판단… 로보·AI 매니저에 돈 몰린다 [간접투자도 AI가 대세]”, 파이낸셜뉴스, 2021.11.09

양선우, “단타매매로 유명세 탄 알고리즘 헤지펀드…금융사 "퀀트 인재 어디 없나요?”, 인베스트 조선, 2019.07.08

최공필, “금융회사 로보어드바이저 시장의 성장 전망”, 코스콤뉴스룸, 2021.8.27