중간고사 보고서
KOSPI 기업의 수익률 예측

<1조>

IT공학과 1812797 이정민

통계학과 1816793 장은조

통계학과 1816001 홍주리

2021.11

목 차

Ⅰ. 서론

1. 분석 프로젝트 개요

□ 개요

▷ 프로젝트명 : KOSPI 기업의 수익률 예측

▷ 프로젝트 도구 : R studio

□ 목적

SR을 극대화 할 수 있는 효율적이고 정확한 모델 찾기

Ⅱ. 데이터 특성

1. 데이터 설명

한국거래소(KRX)에 있는 KOSPI 기업의 월별 관측값 데이터를 분석에 사용하였다.
관측기간은 초기에는 1987년 2월부터 2018년 12월까지 였지만, IMF 구제 금융 이후 상황에 초점을 맞춰 분석하고자 2007년 1월부터 2018년 12월까지의 데이터를 사용하였다.
총 73개의 설명변수들이 있으며, 각 설명변수들은 수익률 관측 시점으로부터 한 달 전까지의 정보로 만든 것이다.

▷ 예측변수(Y) : 수익률 (simple monthly return)
각 기업의 이전 시점 대비 다음 시점의 수익률을 의미

즉, Y가 삼성 기업의 2018년 12월 수익률이라면, X1~X73은 삼성의 2018년 11월 까지 수집된 정보를 바탕으로 만들어진 변수들을 의미한다.
Y = 0.5 의 의미는 전달 대비 수익률이 50% 올랐다는 것을 의미한다.

2. 평가(투자) 전략

▷ 직전 5년치 데이터로 그 다음 1년을 예측하고 1년씩 time window rolling을 적용 [그림1]

▷ [그림2]와 같이 각 기업별 실제수익률과 예측수익률을 구한 후 예측 수익률이 낮은 순서대로 정렬한다.

▷ 정렬된 기업들을 K개의 Group으로 분류한 후 각 그룹별 실제 수익률 평균을 구한다.
* 각 그룹의 평균을 구할 때 기업별 가치가 다름을 감안해서 mv1m 를 weight로 가중평균

▷ 상위그룹과 하위그룹의 맞교환을 통해 얻을 수 있는 실제수익률 이득을 계산한다.

▷ 월별 실제수익률 이득을 계산해서 SR 공식에 대입한다.
이 때, K개의 그룹은 분석가가 정하는 값으로 해당 프로젝트에서는 10개, 20개 선정
각 모델별로 그룹을 10개, 20개로 나눠서 최대의 SR을 갖는 모델을 최종 모델로 선정하려고 한다. 이 때, SR이 비슷하다면 소요시간이 빠른 것을 선정한다.

※ 투자전략의 평가기준 (SR)

$$\text{Shape Ratio} = \frac{\text{평균수익률}}{\text{수익률의표준편차}}$$

위험대비 수익률을 확인할 수 있는 지표로 투자 상황에서 변동성을 감수하는 대신, 얼마나 수익을 얻을 수 있는지를 알아볼 때 사용된다.

[그림 1]

기업	실제수익률	예측수익률
A		
B		
C		
....		
...		

[그림 2]

Ⅲ. 모델 선정 과정 및 결과

1. 회귀 모형

1-1. 다중선형회귀

다중선형회귀는 여러 개의 독립변수(X)들을 가지고 종속변수(Y)를 예측하기 위한 회귀모형이다. 기업을 10개로 분류하였을 때 SR값은 향후 분석과정에서 기준이 될 0.45가 나오고 기업을 20개의 그룹으로 분류하였을 때 SR값은 0.57이 나와 기준값보다 커짐을 확인 할 수 있었다. 다중선형회귀모델의 경우 모델 실행시간이 약 13초로 오래걸리지 않는다는 장점을 갖고 있다.

1-2. KNN regression

독립변수들이 유사한 K개의 이웃을 찾고 유사한 레코드들의 평균을 찾아 새로운 레코드에 대한 예측값으로 사용하는 방법론이다. KNN regression의 경우 k값이 너무 작으면 over fitting, 너무 크면 under fitting한 경향을 보이게 된다. 따라서 적절한 k값을 선택하는 것은 KNN의 성능을 결정하는 중요한 요소이다.

적절한 k 값을 정하기 위해서 수행시간을 고려하여 반복문을 이용해 k를 1부터 50까지 넣어 SR을 극대화 해주는 최적의 k값을 찾았다. 기업을 10개의 그룹으로 분류하였을 때 SR을 극대화 해주는 k값은 14였고 그때의 SR은 0.92였다. 기업을 20개로 분류하였을 때 SR을 극대화 해주는 K값은 23이고 SR은 0.98이었다. 각각의 수행시간은 약 5분정도 소요 되었다.

1-3. Linear Median Regression

$$Q_\tau(y_i)=\beta_0(\tau)+\beta_1(\tau)x_{i1}+\cdots+\beta_p(\tau)x_{ip}$$

기존 회귀분석에서는 최소 제곱법을 이용하여 설명 변수에 따른 반응 변수의 조건부 평균을 추정하지만, 분위수 회귀 분석에서는 반응 변수의 조건부 분위수 값을 추정한다. 주로 통계학과 계량

경제학에서 자주 사용되는 분위수 회귀는 중위수 혹은 분위수 값을 예측하고 싶을 때, 선형 회귀의 조건이 맞지 않을 때, 이상치가 존재할 때, 잔차가 정규성을 만족하지 않을 때 혹은 결과 변수 증가에 따른 오차의 분산이 커질 때 사용하게 된다. 최적의 분위수 방정식을 찾기 위한 과정은 중위수 절대 편차 값을 최소화함으로써 찾을 수 있다.
* 이때 이상치에 영향을 받는 mean regression의 한계를 보완하고자 분위수 회귀를 수행하였다.

분위수회귀를 수행하기 위해서는 R에서 'quantreg' 패키지의 rq 함수를 이용하였다.
중앙값은 0.5분위수이므로 tau를 0.5로 설정해주었다.
기업을 10개의 그룹으로 나누었을 때는 SR값이 0.4, 20개의 그룹으로 나누었을 때는 SR값이 0.58이 나왔고 모델이 돌아가는데 약 3분의 시간이 소요되었다.

1-4. Quantile Regression Forest
R에서 random forest를 이용할 때 사용하는 ranger 함수에서 quantreg를 True로 설정하면 랜덤포레스트의 아이디어를 분위수회귀에 적용한 결과를 얻어낼 수 있다.
기업을 10개의 그룹으로 나누었을 때는 SR값이 0.56, 20개의 그룹으로 나누었을 때는 SR값이 0.69가 나왔고 모델이 돌아가는데 약 30분의 시간이 소요되었다.
분위수 회귀에서는 Quantile Regression forest를 사용하여 기업을 20개의 그룹으로 나누어 SR값을 구했을 때 가장 높은 값을 갖음을 확인 할 수 있었다. 다만 랜덤포레스트와 관련이 있는 모델인 만큼 소요시간이 다른 모델에 비해 훨씬 오래 걸렸다는 단점이 있었다.

2. Ensemble

2-1. Bagging

▷ Bagging 이란?

Bagging은 Bootstrap Aggregation의 약자로 데이터를 여러번 복원추출하여 결과물을 집계하는 방법이다. R에서는 bagging함수를 이용하여 구현할 수 있으며 여러 파라미터 중 nbagging과 coob를 튜닝하여 활용하였다.

▷ 튜닝 파라미터 및 선정 이유

nbagging은 반복추출 횟수를 의미하는 파라미터이다. 해당 파라미터의 값을 높여 반복추출 횟수를 증가시키면 정확도를 일정부분 상향시킬 수 있지만, 모델의 수행시간을 고려하여 100으로 설정하였다.
Coob=True로 설정하여 Out-of-bag(OOB) 추정치가 계산되도록 하였다. 이는 예측 오차를 추정하는 데 사용하도록 하였다. 이를 통해 모델의 정확도를 높이고자 했다.

▷ 결과

전체 데이터에 해당 모델을 적용한 결과, 10개 그룹으로 나눈 경우 SR값이 음수가 나와 유의미하지 않았으며 20개 그룹으로 나눈 경우 SR값이 0.04로 매우 작게 나와 의미있는 결과가 나오지는 않았다. 두 경우 모두 수행시간이 10분 이내로 적절한 시간이 소요되었다.

2-2. Random Forest

▷ Random Forest란?

의사결정 나무를 여러 개 만든 다음 앙상블을 하여 학습성능을 높이는 방법으로 R에서는 randomforest 함수와 ranger 함수를 이용하여 모델을 생성할 수 있다. randomforest 함수보다 ranger 함수가 시간적인 측면에서 효율이 뛰어나기 때문에 ranger 함수를 이용하여 랜덤포레스트 모델을 구현하였다.

▷ 튜닝 파라미터 및 선정 이유

default 값만으로도 충분히 성능을 낼 수 있다고 생각하여 따로 튜닝을 실시하지 않았다.

▷ 결과

전체 데이터에 해당 모델을 적용한 결과, 10개 그룹으로 나눈 경우 SR값이 0.54로 목표했던 0.4보다 크게 도출되었으며 20개 그룹을 나눈 경우에는 SR값이 0.39로 목표했던 0.4에 미치지 못했다. 해당 모델은 기업을 10개 그룹으로 나눈 것이 더 효과적이라고 할 수 있으며 두 경우 모두 수행시간이 20분을 넘어가 시간적인 측면이 효율적이라고 할 수 있을지는 미지수이다.

2-3. Gradient Boosting Machine (GBM)

▷ Boosting 이란?

주요 앙상블 알고리즘은 bagging과 boosting으로 나눌 수 있다. 여러 개의 분류기가 순차적으로 학습을 수행하되, 앞에서 학습한 분류기가 예측이 틀린 데이터에 대해서 올바르게 예측할 수 있도록 다음 분류기에게 가중치(weight)를 부여하면서 학습과 예측을 진행한다.

이 때, 경사하강법(Gradient descent)를 이용하여 가중치를 업데이트 하는 방법을 GBM이라 한다.

GBM의 파라미터는 표[1]과 같이 다양하다. 모든 파라미터를 튜닝하는 것은 비효율적이라 판단이 되어, 일부의 파라미터값만 튜닝을 하기로 하였다.

▷ 튜닝 파라미터 및 선정 이유

n.trees는 클수록 training set의 error는 줄일 수 있으나, 과적합을 일으킬 수 있다.

interaction.depth는 클수록 모델의 복잡도가 증가하며, 과적합을 일으킬 수 있다.

shrinkage는 너무 크면 overshooting & 과적합 문제 발생하며, 너무 작으면 학습 시간이 오래 걸린다. 과적합의 이슈를 해결하기 위해 3개의 파라미터를 선정하였다.

▷ 파라미터 튜닝 후보값

n.trees의 값은 (3,5,7), interaction.depth의 값은 (1,3,5), shrinkage의 값은 (0.1,0.05,0.01) 값을 후보값으로 정해서 총 27가지 파라미터 조합을 생성시켰다.

▷ 파라미터 튜닝 과정

튜닝 데이터 : 2002년 1월 ~ 2006년 12월 총 5년의 데이터

튜닝 과정 : R의 gbm함수와 for문을 통해 27가지 파라미터 조합을 CV(교차검증) = 5로 설정하고 돌림. cv.error가 가장 작은 파라미터 조합을 최종 파라미터로 선정

위의 과정을 통해 나온 최종 파라미터를 데이터가 바뀔 때 마다 적용시켰다.

훈련 데이터는 time window rolling 의해 매번 바뀌는데, 초반 5년의 데이터를 학습해서 나온 파라미터를 모든 데이터에 고정적으로 사용하였다. 이유는 time window rolling 될 때마다 파라미터를 학습시키는 시간이 오래걸린다는 판단이었다. 시간적 여유가 있다면 이 부분을 보완해서 바뀌는 데이터셋마다 튜닝을 새롭게 해보려고 한다.

고정된 파라미터를 전체 데이터에 적용한 결과, 기업을 10개의 그룹으로 나누었을 때 SR값이 0.93, 20개의 그룹으로 나누었을 때는 0.38이 나왔다. 27개의 파라미터 조합을 튜닝하는데 걸린 시간은 약 5.7분이 걸렸고, 모델이 돌아가는데는 2.8분의 시간이 소요되었다.

2-4. eXtreme Gradient Boosting
Gradient descent boosting 기법에 병렬 학습이 지원되도록 구현한 라이브러리이다.
GBM 대비 수행시간이 빠르며, 표준 GBM 경우 과적합 규제기능이 없으나, XGBoost는 자체적으로 과적합 규제 기능으로 강한 내구성을 지닌다. 또한 Early stopping 기능이 있어 더 이상 성능의 발전이 없을 경우 멈춰추는 기능도 있다.

▷ 튜닝 파라미터 및 선정 이유
max_depth는 클수록 training set의 error는 줄일 수 있으나, 과적합을 일으킬 수 있다.
colsample_bytree 는 낮을수록 과적합을 방지해준다.
eta는 너무 크면 overshooting & 과적합 문제가 발생하며, 너무 작으면 학습 시간이 오래 걸린다. 과적합의 이슈를 해결하기 위해 3개의 파라미터를 선정하였다.
추가적으로 어떤 부스터 구조가 적합한지 선택하기 위해 booster 파라미터도 추가하여
총 4개의 파라미터를 선정하였다.

▷ 파라미터 튜닝 후보값
max_depth의 값은 (3,4,5), colsample_bytree의 값은 (0.5,0.7,1), eta의 값은 (0.1,0.05,0.01), booster는 ('gbtree', 'gblinear', 'dart')를 후보값으로 정해서 총 81가지 파라미터 조합을 생성시켰다.

▷ 파라미터 튜닝 과정
Gradient descent boosting 파라미터 튜닝 과정과 동일

고정된 파라미터를 전체 데이터에 적용한 결과, 기업을 10개의 그룹으로 나누었을 때 SR값이 14.68, 20개의 그룹으로 나누었을 때는 13.28이 나왔다. 81개의 파라미터 조합을 튜닝하는데 걸린 시간은 약 10.4분이 걸렸고, 모델이 돌아가는데는 1.2분의 시간이 소요되었다.
GBM보다 성능과 속도면에서 압도적으로 향상시킨 모델임을 한번 더 확인할 수 있었다.

▷ GBM의 최대 SR은 0.93 VS XGB의 최대 SR 14.68 차이가 많이 나는 이유는?
GBM은 가지치기 방식이 Leaf-wise(Best-first) 방식, 즉 보통은 Infomation Gain을 기준으로 가지를 뻗쳐나간다. 그러나 XGB는 반대로 지정된 최대 깊이(max_depth)까지 분할 한 다음, 이득이 없는 가지를 쳐낸다. 때로는 Information Gain이 없는 가지 밑에도 Information Gain이 있는 가지가 생겨날 수 있다는 가정이다. 또한 GBM은 regularization이 없지만 XGB는 있어서 GBM보다 과적합을 방지해준다. 이런 2가지 요소가 XGB가 압도적으로 GBM보다 SR이 높게 나왔던 이유이지 않을까 생각한다.

Ⅳ. 분석 결과 요약

종류	SR	수행시간
Linear Mean Regression	0.57	12.81 secs
KNN Regression	0.98	5.49 secs
Linear Median Regression	0.58	3.17 mins
Quantile Regression Forest	0.69	28.03 mins
Bagging	0.04	8.66 mins
Random Forest	0.54	22.95 mins
Gradieent Boosting	0.93	2.88 mins
eXtreme Gradient Boosting	14.68	1.17 mins

eXtreme Gradient Boosting(XGB) 모델의 SR이 14.68로 다른 7개의 모델들보다 압도적으로 높은 값을 보였다. 소요시간 또한 1.17분으로 오래 걸리지 않는 장점이 있다. 초반의 분석 목표였던 최대 SR, 빠른 소요시간 2가지 요소를 만족한 XGB를 최종 모델로 선정하였다. 그 다음으로는 회귀모형 중 KNN Regression 모델이 0.98로 GBM과 비슷한 SR값을 가지지만, GBM보다 압도적으로 빠른 시간을 보였다.

Ⅴ. 아쉬운 점

▷ GBM, XGB 파라미터 튜닝 과정을 time window rolling에 의해 매번 바뀌는 훈련 데이터셋에 맞춰 time window rolling 될 때마다 파라미터를 학습시켰다면 좀 더 좋은 성능을 보였을 것 같다.

▷ Bagging 파라미터 중 nbag는 의사결정 트리의 개수를 의미하는 파라미터인데, 이 파라미터를 조정해보지 못하고 모델을 튜닝했던 점이 아쉽다.

▷ 랜덤포레스트 모델을 구현하면서 파라미터에 대한 이해가 부족하여 튜닝을 못한 점이 아쉽다. 다음에 기회가 된다면 num.trees의 값을 조정하여 의사결정 나무의 수가 모델의 정확도에 미치는 영향에 대해서 공부해보고 싶다.