



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

碩士學位論文

머신러닝을 활용한 이동평균선
교차전략의 이상적 매매에 관한
연구

A Study on the Ideal Trade of Moving Average Line
Crossing Strategy using Machine Learning



國民大學校 비즈니스IT專門大學院
트레이딩시스템 專攻

金 錫 俊

2019

머신러닝을 활용한 이동평균선 교차전략의 이상적 매매에 관한 연구

A Study on the Ideal Trade of Moving Average Line
Crossing Strategy using Machine Learning

指導教授 김 선 응



이 論文을 碩士學位 請求論文으로 提出함

2020 年 02 月

國民大學校 비즈니스IT專門大學院
트레이딩시스템 專攻

金 錫 俊

2019

金錫俊 의
碩士學位 請求論文을 認准함

2020 年 02 月



審査委員長 최 홍 식 ①

審査委員 김 선 응 ①

審査委員 안 현 철 ①

國民大學校 비즈니스IT專門大學院

차 례

그림 차례	ii
표 차례	iii
국문 요약	iv
제 1 장 서론	1
1.1 연구의 배경 및 목적	1
제 2 장 문헌 및 이론 연구	4
2.1 이동평균선의 전략유형	4
2.1.1 교차점 매매	4
2.1.2 상향점/하향점 매매	4
2.2 이동평균선 기계학습 연구	6
2.3 머신러닝 이론 연구	7
2.4 연구의 방향성	13
제 3 장 실험 실습	15
3.1 실험 모형 개요	15
3.2 실험 모형	16
3.2.1 지표산정/정규화	16
3.2.2 상향점/하향점 매매	18
3.2.3 머신러닝 학습	21
3.2.4 로짓회귀 매매	24
제 4 장 결론	26
4.1 실험실습 결론	26
4.2 연구한계 및 방향성	27
참고 문헌	29
ABSTRACT	30

그림 차례

<그림 1> 교차점 매매	4
<그림 2> 상향점/하향점 매매	5
<그림 3> TLU	8
<그림 4> SVM 하드마진	9
<그림 5> 가중치 벡터와 마진	10
<그림 6> 로지스틱 함수	13
<그림 7> 실험모형 시스템 구조도	15
<그림 8> 교차점 매매 프로세스 순서도	19
<그림 9> 5일/20일 이동평균선의 교차점 구간	20
<그림 10> 알고리즘 풀 데이터 셋 훈련/테스트 구간	22
<그림 11> 학습 알고리즘 선택 및 데이터 셋 준비	22
<그림 12> 훈련/테스트의 분리	23
<그림 13> 모델 학습 및 성과 측정	23
<그림 14> 데이터 변형 및 재학습	23
<그림 15> 파라미터 설정 및 재학습	24
<그림 16> 밴드별 수익률 및 그래프	25

표 차례

<표 1> 지수별 이동평균 장기단기 교차점매매 수익률	2
<표 2> 지수별 교차일수가 10일 이하인 교차점 매매 수익률	2
<표 3> 상향점/하향점 매매 수익률	6
<표 4> 사이킷런 실험 알고리즘	13
<표 5> 실험모형 단계별 실험수행 내용	16
<표 6> 상대 강도 지수 산정예시	17
<표 7> 종가 및 이동평균 가격 정규화 된 변환가격	18
<표 8> 일일 교차점 거래내역	20
<표 9> 상향점/하향점 매매 내역	21
<표 10> 알고리즘 학습처리 절차	21
<표 11> 알고리즘별 분류예측 성과 결과표	26
<표 12> 로짓회귀 알고리즘의 매매수익률	27



머신러닝을 활용한 이동평균선 교차전략의 이상적 매매에 관한 연구

김 석 준

트레이딩시스템 전공

이동평균선의 교차전략은 현업에서 가장 많이 이용하는 기술적 지표중의 하나이다. 가장 쉽게 이해 할 수 있고 가격의 추세파악이 용이하고 이동 평균선을 따라 다양한 전략이 가능하다. 5일 이동 평균선을 하나만 이용 할 경우 가격이 평균선과 멀어 질 때는 평균선으로 회귀 전략을 구사할 수 있다. 또한 단기 이동 평균선은 장기 이동 평균선에 비하여 높은 변동성을 보이는데 이 성질을 이용해서 두 개의 곡선이 주기적으로 교차하는데 상향 돌파 시 매수, 하향 돌파 시 매도의 전략을 구사할 수도 있다.

그러나 이동평균선의 교차전략은 가격의 후행성 때문에 매매 신호가 부정확하게 발생 한다. 포지션의 방향이 설정되어서 가격이 움직이는 진행 방향과 반대의 매매 신호가 자주 발생 한다. 교차매매의 코스피 지수 수익률이 -46.48%이다. 같은 대상기간의 단순 수익률이 115%에 비하면 교차곡선의 매매 신호는 부정확 하다고 할 수 있다.

본 연구는 교차점 매매의 한 구간에서 가장 높은 점(상향점)과 가장 낮은 점(하향점)을 판단한다. 교차점에서 물리적 거리가 가장 큰 지점이 상향점 또는 하향점이

되고 상향점 에서 매도(매수청산), 하향점 에서 매수(매도청산)를 진행한다. 이와같은 매매 방식을 이동평균선의 상향점/하향점 매매라고 하고 코스피 지수의 교차점 매매수익률과 같은 기간에 2712.39% 매매 수익률을 기록한다. 극단의 수익률을 만들어 내는 이동평균선의 가장 이상적인 매매이다.

상향점/하향점 매매를 수행한 매매 내역을 매수/매도 포지션을 머신러닝의 지도학습의 선형/비선형 알고리즘을 선택하여 학습 시킨다. 본 연구에서는 퍼셉트론, SVM, 로짓회귀 알고리즘을 선택하여 각각 선형/비선형 방식으로 학습을 진행한다. 위 알고리즘 중에서 로짓회귀 알고리즘은 분류/예측 결과를 확률로 제시하므로 학습된 확률 결과 값을 일일 매매를 진행하여 수익률을 산정한다.

로짓회귀 알고리즘의 학습 결과를 매매하는 방식은 매수 시 확률 값이 35미만인 경우, 매도 시 65초과인 경우, 그 사이(35~65) 인 경우는 전 일자 포지션을 유지하는 방식인데 밴드 폭을 35-65 뿐만 아니라 10부터 5단위씩 45까지 진행하여 수익률을 파악한다.

로짓회귀 알고리즘이 학습한 테스트 구간의 매매수익률은 236% 상향점/하향점 매매수익률은 370% 교차점 수익률은 5.53 단순 수익률이 12.79인데 상당한 의미 있는 수익률을 실험하였다.

교차곡선의 이상적 매매인 상향점/하향점 매매는 현실 세계에서 존재 할 수 없다. 교차점이 최소한 한 구간이 이루어져야만 교차곡선의 변곡점을 파악해 볼 수 있기 때문이다. 현실세계에 존재 할 수 없지만 이상적 교차점을 현실 세계에서서 파악하는 것이 본 논문의 한계점이자 이후의 연구 과제이다.

키워드 : 기계학습, 이동평균선, SVM, 로짓회귀, 퍼셉트론

제 1장 서론

1.1 연구의 배경 및 목적

이동평균가격의 기술적 지표는 현업에 종사하는 트레이더들 에게 가장 많이 활용되는 기술적 지표중의 하나이다. 분석의 용이성뿐만 아니라 장기/단기의 평균 가격곡선이 교차하는 선 위(또는 아래)에 현재의 가격을 위치시키면 지난 과거의 가격의 움직임과 현재의 가격의 쉬운 비교가 가능하다. 선명하고 쉬운 지표는 움직임이 복잡한 현상일수록 그 현상을 분석하는데 뚜렷한 장점이 있다.

시계열상의 장기/단기의 평균 가격곡선의 가장 큰 특징은 임의 주기를 갖고 단기곡선과 장기곡선이 교차하면서 곡선을 형성한다는 것이다. 단기의 가격 움직임이 강세이면 장기의 가격곡선을 상향돌파하고 단기의 가격움직임이 약세이면 장기의 가격곡선을 하향돌파하는 움직임을 임의의 주기를 갖고 반복한다는 것이다. 두 곡선이 만나면서 일종의 불규칙한 파동을 만드는 것이다. 주기별 파동을 만들어 내는 비선형적 데이터 분포는 기계학습 알고리즘의 오랜 연구 분야에 좋은 재료가 된다.

데이터 분포의 예측을 위한 기계학습의 다양한 알고리즘은 선형성 일차방정식에서부터 비선형적인 다차항의 고차방정식까지 다양하게 활용되어오고 있다. 특히 비선형적인 데이터 분포의 분류 및 예측에 좋은 성과를 보이고 있는 로짓회귀 알고리즘과 SVM의 학습알고리즘은 이동평균가격곡선이 그리는 비선형적인 가격 분포 데이터의 분류/예측에 가장 적합하다.

이동평균선이 단순하고 쉬운 지표로서 장점은 있지만 매매신호를 파악하는데 가격의 후행성 때문에 치명적인 약점이 있다. 이동평균 가격의 후행성은 가격을 평균화함으로서 가격의 변화의 폭이 현재의 가격의 움직임보다 느리게 움직이므로 매매신호로서 신호 발생이 늦게 포착된다.

<표1>은 단기(5일) 이동평균선과 장기(20일) 이동평균선이 교차할 때 매매신

호를 파악하여 상향돌파(매수, 매도청산 포지션)와 하향돌파(매도, 매수청산 포지션)시 매매를 수행한 수익률이다. 단순 시장수익률에 비교 할 수 없을 정도로 낮다. <표2>는 교차일수가 단기 10일 이내 수익률이다. 엄청난 손실을 기록하고 있는데 교차주기가 짧다는 의미는 시장의 가격이 추세가 형성 되지 않고 횡보를 한다는 것이다. 가격의 변동성이 낮을 경우에는 가격의 후행적인 매매 신호는 더 치명적일 수 있다.

이동평균수익률(%) = (진입(신규)가격 - 청산가격) * 100 / 청산가격

단순수익률(%) = (실험종료종가 - 실험시작종가) * 100 / 실험시작종가

지수명	이 동 평 균 수 익 률	실 험 데 이 터 시 작종가	실 험 데 이 터 종 료 종 가	단 순 수 익 률	실 험 데 이 터 시 작일	실 험 데 이 터 종 료일
코스피	-46.48	967.72	2080.27	111	1999-07-30	2019-10-30
다우	-87.09	1929.7	20071.42	2312	1999-12-06	2019-10-29
상해	149.58	2954.91	2939.32	-1	2007-03-12	2019-10-30
니케이	33.3	8923.41	22843.12	156	2003-06-26	2019-10-30
닥스	-12.9	3198.82	12939.62	305	2003-06-26	2019-10-29

<표1> 지수별 이동평균 장기단기 교차점매매 수익률

지수명	수익률 합계
코스피	-427.60
다우	-310.42
상해	-271.65
니케이	-268.93
닥스	-310.30

<표2> 지수별 교차일수가 10일 이하인 교차점매매 수익률

이러한 이동평균가격의 후행적인 매매신호를 조금 더 빨리 파악하는 방법으

로 단기 이동평균선의 상향점/하향점을 식별하여 그 지점에서 매매를 실현해보고 그 매매 데이터를 기계학습 알고리즘으로 학습하여 분류/예측해보는 방법으로 가격의 후행성을 극복하고 최적의 매매 타이밍을 찾아보고자 한다. 기계학습 알고리즘이 제안하는 매매신호에 기반 하여 매매를 수행해보고 수익률도 산정해본다.

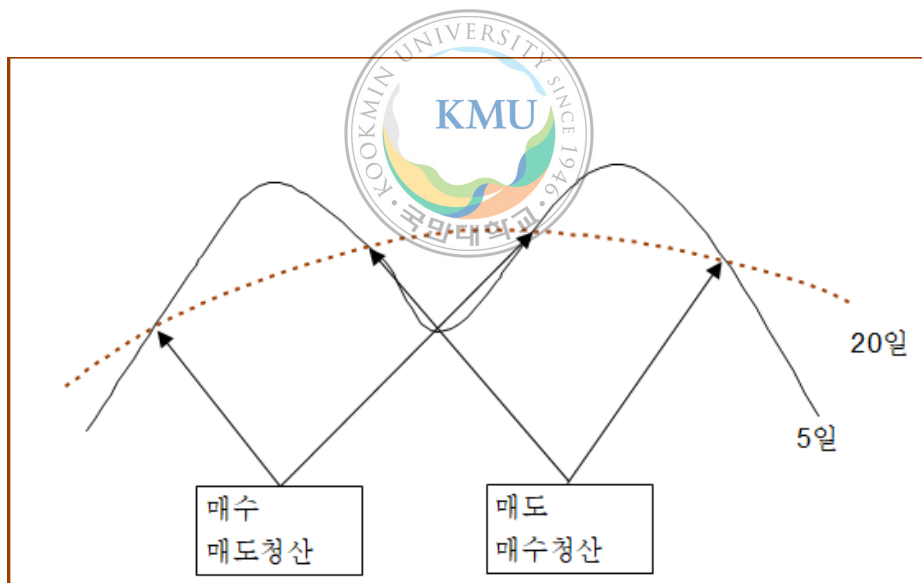


제 2장 문헌 및 이론 연구

2.1 이동평균선의 전략유형

2.1.1 교차점 매매

가장 전통적인 매매 방식으로 단기이동평균선이 장기이동평균선을 상향 돌파했을 때 매수(골든 크로스)하고 단기이동평균선이 장기이동평균선을 하향 돌파했을 때 매도(데드 크로스)하는 매매 방식을 본 연구에서는 교차점 매매라고 한다. <그림1>은 단기 5일 이동평균선과 장기 20일 이동평균선의 교차곡선을 보여주는 그림이다.

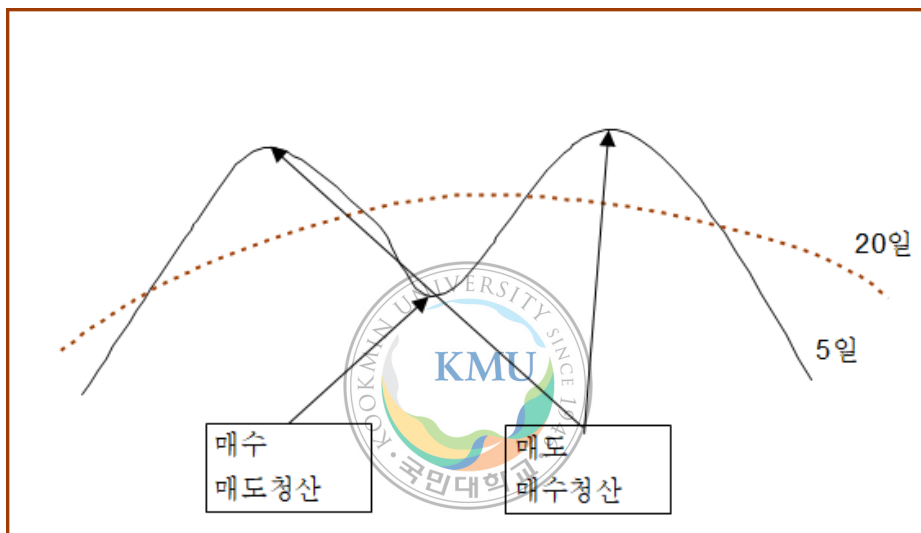


<그림1> 교차점 매매

2.1.2 상향점/하향점 매매

단기이동평균선과 장기이동평균선의 상향점에서 매도, 매수청산 포지션을 하향점에서 매수, 매도청산 포지션으로 매매하는 방식을 본 연구

에서는 상향점/하향점 매매라고 한다. 상향점/하향점은 교차점에서 가격의 물리적 거리(Distance)가 가장 큰 값이 위치한 지점이고 단기이동평균곡선의 기울기가 0에 근접하는 (평균변화율이 가장 작은 값) 위치가 되는데 물리적 거리를 산정하여 가장 큰 값을 갖는 기준점을 상향점/하향점으로 설정하였다.



<그림2> 상향점/하향점 매매

가격의 후행성을 완벽하게 극복한 매매 방식으로 현실세계에서는 존재하지 않은 이상적인 매매내역이다. 변동성이 큰 5일 이동평균선의 상향점에서는 매도신규(매수청산) 하향점에서는 매수신규(매도청산)의 포지션으로 상향점/하향점 매매를 수행해보고 그 수익률을 산정해봤다. <표3>은 지수별 상향점/하향점 매매 수익률인데 교차점 매매수익률과 비교를 할 수 없을 정도로 이상적인 수익률을 확인 할 수 있다.

지수명	수익률 합계
코스피	2712.39
다우	1848.58
상해	1842.65

니케이	2108.14
닥스	1957.56

<표3> 상향점/하향점 매매 수익률

2.2 이동평균선 기계학습 연구

기계학습 알고리즘을 이용한 주가예측의 연구는 트레이딩 분야에서 왕성하고 활발히 이루어지고 있다. 유발 하라리가 언급했듯이 “요즘은 알고리즘의 시대라”는 말이 상징하듯이 트레이딩의 지수, 상품, 주식, 옵션 등 전 분야에 가격의 움직임을 해석해보려는 경향은 기존의 시스템트레이딩을 넘어 일반적인 트레이딩 현상으로 자리 잡고 있다.

초기 단순한 가격의 지표데이터를 알고리즘의 특성에 맞게 예측 해보는 것 등이 주류의 연구였다면 실제 매매 수익률을 높이기 위해 핵심 가격 데이터의 특성과 시장의 경제변수들, 비가격의 정형화된 데이터의 특성을 적합한 알고리즘으로 예측해보려는 시도가 일반화 되고 있다.

안현철, 김선웅은 KOSPI200 지수 데이터를 이용하여 “기술적 지표뿐만 아니라 실무에서 활용되는 각종 비가격 지표들도 함께 사용하는” 연구를 발표한다. 입력변수들의 예측결과를 추정확률을 사용할 수 있는 변형SVM 모델을 활용하여 반환되는 확률 값을 이중 임계치 결정 방식으로, 하단부터 상단까지 예측확률에 속하는 분류는 보류(Hold)하는 방식으로 매매 시스템을 연구하였다. 이중 임계치의 상단/하단 밴드를 다시 유전자 알고리즘으로 분류하여 최적화되는 상단/하단의 밴드를 설정하도록 하였다. 임계치 방식의 상단/하단의 밴드를 이용하여 매매를 보류(그 전의 포지션을 유지)하는 연구는 알고리즘의 학습률을 높이는 기존의 연구에서 실전 매매 수익률을 파악해보는 실전 전략으로서도 의미 있는 연구였다.

이동평균가격의 지표를 활용한 연구에서 박성철은 장기와 단기이동평균의 일반 교차점 매매(상향돌파-매수, 하향돌파-매도)와 마코프 모

형(CHMM)의 지표로 사용하여 매매 수익률을 비교하였다. 단기와 장기이동평균선의 조합은 일반적인 교차점 매매 시 10일과 40일, 마코프모형을 활용한 매매 시 20일과 40일의 조합이 가장 높은 수익률을 나타냈다.

이동평균 가격의 가장 큰 특징은 추세 파악이 쉽다는 것이다. 현재 가격을 과거 평균가격과 비교는 현재가격이 평균가격과 어느 정도 떨어져서 움직이는지를 파악해 봄으로써 추세, 역 추세, 횡보구간의 정의가 비교적 쉽다. 오정환은 이동평균가격의 이러한 특성을 이용하여 VKOSPI 변동성 지수의 움직임을 파악하여 옵션의 매매전략에 활용한다. 즉 VKOSPI 추세를 장단기 이동평균선의 교차점으로 판단하여 추세적인 변동성 증가, 감소를 예측하여 그에 맞는 옵션전략 즉 변동성 증가는 양 매수 전략, 변동성 감소는 양 매도 전략을 구사하는 연구를 했다. 공포지수의 추세 파악과 그에 연동된 옵션매매 전략은 가격이나 지수의 이동평균 곡선이 그려내는 추세패턴을 변동성 분석에 활용한 연구로서 의의가 있다.

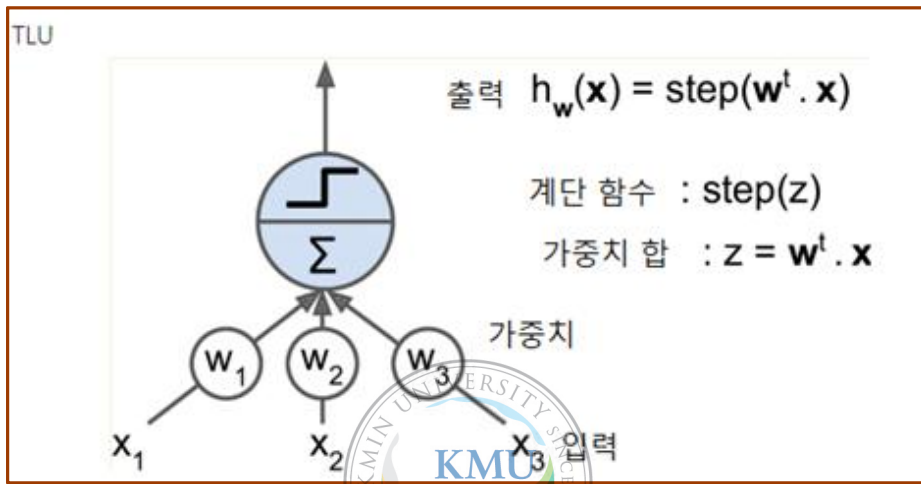
2.3 머신러닝 이론 연구

■ 퍼셉트론

퍼셉트론은 가장 간단한 인공 신경망 구조중의 하나로 1957년에 프랑크 로젠블라트가 제안하였다. 퍼셉트론은 TLU(threshold logic unit)라는 인공뉴런을 기반으로 한다. 입력과 출력이 어떤 숫자고 각각의 입력연결은 가중치와 연관되어있다.

TLU는 가중치의 합을 계산하고 ($z = w_1x_1 + w_2x_2 + w_3x_3 + \dots w_nx_n$) 계단함수 (step function)을 적용하여 그 결과를 출력한다. 즉 $h_w(x) = \text{step}(z) = \text{step}(w^t \cdot x)$ 이다. 퍼셉트론에서 가장 널리 사용되는 계단함수는 헤비사이드 계단함수이다

$$heaviside(z) = \begin{cases} 0 & z < 0 \text{ 일 때} \\ 1 & z \geq 0 \text{ 일 때} \end{cases} \quad \text{sgn}(z) = \begin{cases} -1 & z < 0 \text{ 일 때} \\ 0 & z = 0 \text{ 일 때} \\ +1 & z > 0 \text{ 일 때} \end{cases}$$



<그림3> TLU

퍼셉트론은 학습 원리는 다음과 같다.

$$w_{i,j} = w_{i,j} + \eta(y_j - \hat{y}_j)x_i$$

- ▷ $w_{i,j}$ 는 I번째 입력뉴런과 j번째 출력뉴런 사이를 연결하는 가중치
- ▷ x_i 는 현재 훈련 샘플의 i번째 뉴런 입력 값
- ▷ \hat{y}_j 는 현재 훈련 샘플의 j번째 출력뉴런의 출력 값
- ▷ y_j 는 현재 훈련 샘플의 j번째 출력뉴런의 타깃 값
- ▷ η 는 학습률

퍼셉트론의 선형적인 학습 원리는 배타적 논리합 XOR 문제를 풀어내지 못하면서 긴 암흑기를 보내고 딥러닝 알고리즘으로 화려하게 부활한다. 여러 퍼셉트론의 layer를 만들면 XOR의 제약을 줄 일수 있다는 사실이 밝혀졌다. 이런 인공 신경망을 다층 퍼셉트론 (Multi Layer

Perceptron)이라 하고 다층 퍼셉트론으로 XOR문제를 풀 수 있게 되면서 인공 신경망 알고리즘은 다시 많은 연구 분야에 활용된다.

■ SVM

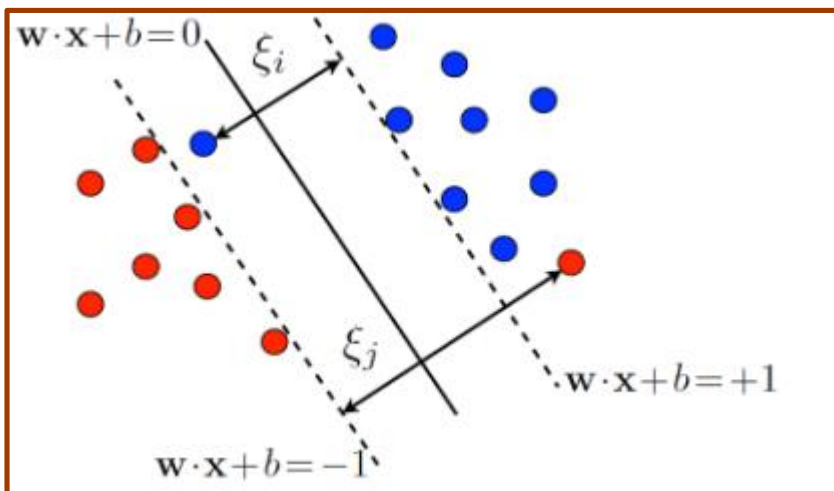
▷ 결정함수

가중치 벡터 w 편향을 b 라고 하면 선형 SVM분류기 모델은 단순히 결정함수 $w^t \cdot x + b = w_1x_1 + w_2x_2 + \dots + w_nx_n + b$ 을 계산하여 벡터로 구성된 새로운 클래스를 예측한다. 결과 값이 0보다 크면 예측된 클래스 \hat{y} 은 양성클래스(1)이 되고 그렇지 않으면 음성클래스(0)이 된다.

선형 SVM 분류기의 예측

$$\hat{y} = \begin{cases} 0 & w^t \cdot x + b < 0 \text{ 일 때} \\ 1 & w^t \cdot x + b \geq 0 \text{ 일 때} \end{cases}$$

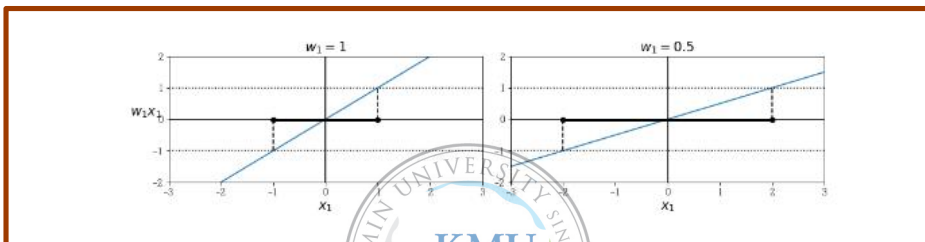
오렐리앙 제롱의 [핸즈온 머신러닝]에 따르면 “선형 SVM 분류기를 훈련한다는 것은 마진 오류를 하나도 발생하지 않거나 (하드 마진) 제한적인 마진오류를 가지면서(소프트 마진)가능한 한 마진을 크게 하는 w 와 b 를 찾는 것이다”라고 정의한다.



<그림4> SVM 하드 마진

▷ 목적함수

목적함수의 정의는 “결정함수의 기울기는 가중치 벡터의 노름 $\|w\|$ 과 같다. 이 기울기를 2로 나누면 결정함수 값이 ± 1 이 되는 점들이 결정경계에서 2배만큼 멀어진다. 즉 기울기를 2로 나누는 것은 마진에 2를 곱하는 것 과 같다”라고 정의하고 아래 그림처럼 가중치 벡터 w 가 작을수록 마진은 커진다.



<그림5>가중치 벡터와 마진

즉 마진을 크게 하기 위해 $\|w\|$ 를 최소화 하려고 한다. 그러나 마진 오류를 하나도 만들지 않으려면 (하드마진), 결정함수가 모든 양성 훈련샘플에서는 1보다 커야하고 음성 훈련 샘플에서는 -1보다 작아야 한다. 음성 샘플($y^{(i)} = -1$)일 때 $t^{(i)} = -1$ 로, 양성 샘플($y^{(i)} = 1$)일 때 $t^{(i)} = 1$ 로 정의하면 제약조건을 모든 샘플에서 $t^{(i)}(w^T \cdot x^{(i)} + b) \geq 1$ 로 표현 할 수 있다.

▷ 하드 마진 선형 SVM 분류기의 목적함수

$$\text{minimize}_{w,b} \frac{1}{2} w^T \cdot w$$

$$\text{subject to } i = 1, 2, \dots, m \text{ 일 때 } t^{(i)}(w^T \cdot x^{(i)} + b) \geq 1$$

소프트 마진 분류기의 목적함수를 구성하려면 각 샘플에 대한 슬랙 변수 $\zeta^{(i)} \geq 0$ 을 도입해야 한다. $\zeta^{(i)}$ 는 I번째 샘플이 얼마나 마진을 위

반할 지를 정한다. 이 문제는 2개의 상충되는 목표를 갖고 있는데 마진오류를 최소화하기 위해 가능한 슬랙 변수의 값을 작게 만드는 것과 마진을 크게하기 위해 $\frac{1}{2}w^T \cdot x$ 을 가능한 작게 만드는 것이다. 여기에 하이퍼파라미터 C가 등장한다. 이 파라미터는 두 목표사이의 트레이드 오프를 정의한다.

▷ 소프트 마진 선형 SVM 분류기의 목적함수

$$\text{minimize}_{w,b,\zeta} \frac{1}{2}w^T \cdot x + C \sum_{i=1}^m \zeta^{(i)}$$

subject to $i=1,2,\dots, m$ 일 때 $t^{(i)}(w^T \cdot x^{(i)} + b) \geq 1 - \zeta^{(i)}$ 이고 $\zeta^{(i)} \geq 0$

▷ 커널SVM

데이터 셋의 차원을 한 차원 높여주는 것을 커널 트릭이라고 한다. 다음 2차 다항식 매핑 함수는 $\phi(x)$ 이다.

$$\phi(x) = \phi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix}$$

변환된 벡터는 2차원이 아니고 3차원이 된다. 2개의 2차원 벡터 a와 b에 2차 다항식 매핑을 적용한 다음 변환된 벡터로 점곱을 하면 다음과 같다.

$$\begin{aligned} \phi(a)^T \cdot \phi(b) &= \begin{pmatrix} a_1^2 \\ \sqrt{2} a_1 a_2 \\ a_2^2 \end{pmatrix}^T \cdot \begin{pmatrix} b_1^2 \\ \sqrt{2} b_1 b_2 \\ b_2^2 \end{pmatrix} = a_1^2 b_1^2 + 2a_1 b_1 a_2 b_2 + a_2^2 b_2^2 \\ &= (a_1 b_1 + a_2 b_2)^2 = \left(\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}^T \cdot \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}\right)^2 = (a^T \cdot b)^2 \end{aligned}$$

변환된 벡터의 점곱이 원래 벡터의 제곱과 같다. 모든 훈련샘플에 변

환 ϕ 를 적용하면 실제로 훈련 샘플을 변환할 필요가 없다. 즉 점곱을 제곱으로 바꾸기만 하면 된다. 결과 값은 실제로 훈련샘플을 어렵게 변환해서 선형 SVM알고리즘을 적용하는 것과 완전히 같다. 이것이 커널 트릭이다. 함수 $K(a,b) = (a^T \cdot b)^2$ 을 2차 다항식 커널이라고 부른다. 머신러닝에서 커널은 변환 ϕ 를 계산하지 않고 원래 벡터 a 와 b 에 기반 하여 점곱 $\phi(a)^T \cdot \phi(b)$ 를 계산 할 수 있는 함수이다. 다음은 SVM에서 널리 사용되는 커널 함수이다

선형 : $K(a,b) = a^T \cdot b$

다항식 : $K(a,b) = (\gamma a^T \cdot b + r)^d$

가우시안 RBF : $K(a,b) = \exp(-\gamma \|a-b\|^2)$

시그모이드 : $K(a,b) = \tanh(\gamma a^T \cdot b + r)$

■ 로짓 회귀

로지스틱함수는 변수변환에 따라 의해 선형 또는 비선형으로 바꾸기 용이하기 때문에 널리 사용된다. 즉, P_i 가 X_i 가 어떤 값을 취하든 0에서 1사이의 확률 값을 갖는다. 이러한 로지스틱 함수를 따른다고 가정한다.

$$P_i = \frac{\exp(\beta_0 + \beta_1 X_1)}{1 + \exp(\beta_0 + \beta_1 X_1)}$$

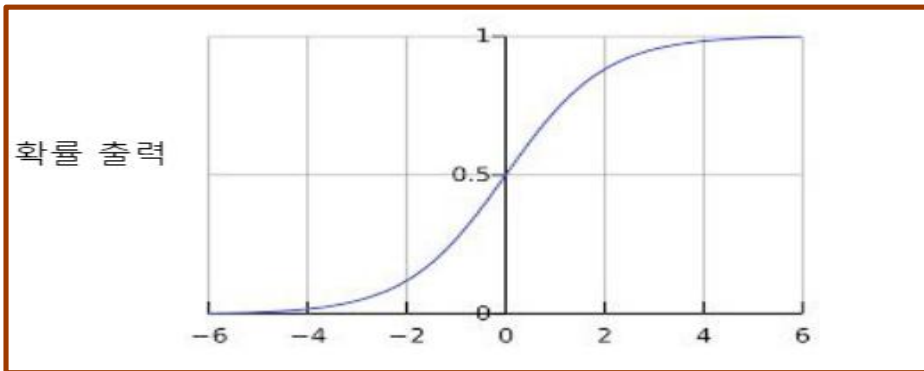
위의 로지스틱 함수는 <그림6>에서 보듯이 X값에 상관없이 항상 확률P가 0과 1사이를 갖게 된다.

위의 식은 변수 X 및 계수들에 대하여 선형이 아니므로 추정이 어렵다. 그러나 P를 다음과 같이 변환하면 선형 회귀식으로 바꿀 수 있다

$$\text{logit}(P) = \ln \left[\frac{P}{1-P} \right]$$

위의 변환을 로짓(logit)변환이라 한다. 이때 로지스틱 회귀모형은 로

짓 변환을 통하여 다음과 같은 선형모형 형태로 표현 된다.



<그림6> 로지스틱 함수

$$\text{logit}(P) = \beta_0 + \beta_1 X$$

위 식 β_1 은 X가 한 단위 증가할 때 P의 함수, 즉 승산비(odds ratio)의 로그값의 증가분을 말하므로 승산비가 e^{β_1} 배로 증가함을 의미 한다.

이상과 같이 알고리즘별로 특성과 이론을 고찰해 봤다. 사이킷런에서 구현하는 알고리즘은 아래 표와 같다. 하이퍼파라미터를 통해서 모델의 복잡도를 제어 할 수 있고 또한 그리드서치(gridsearch)를 통해서 파라미터의 조합을 통해서 모델의 우수한 성능을 찾아가는 방식으로 실험을 한다.

알고리즘	클래스명	선형/비선형	하이퍼파라미터
퍼셉트론	Perceptron	선형	없음
SVM	SVC	선형	마진, 코스트
		비선형	마진, 코스트, 커널
로지스틱회귀	LogitRegression	선형	없음
		비선형	복잡도, 차원증가

<표4> 사이킷런 실험 알고리즘

2.4 연구의 방향성

이동평균 가격의 범용적인 활용에도 불구하고 매매신호를 활용하는

지표로서는 큰 수익률을 거둘 수 없다. 가장 일반적인 교차점 매매에서도 알 수 있듯이 평균 가격의 후행성 때문에 교차주기가 짧아지는 횡보구간에는 매매신호를 활용 할 경우 큰 손실이 예상된다. 이러한 가격의 후행성은 현업에서는 여러 가지 보조지표를 활용하거나 비추세 구간은 감안하여 활용하기도 한다.

한편, 기계학습의 다양한 알고리즘은 연구/개발의 편의성 증대를 위해 단일하고 통합(integrated)된 환경으로 제공 된다. 인공지능의 퍼셉트론의 선형 알고리즘에서부터 현재의 딥러닝 까지 통합된 라이브러리로 제공되고 있고 본 연구에서는 아나콘다의 개발환경에서 사이킷런(scikit-learn)의 라이브러리를 이용하여 분석한다.

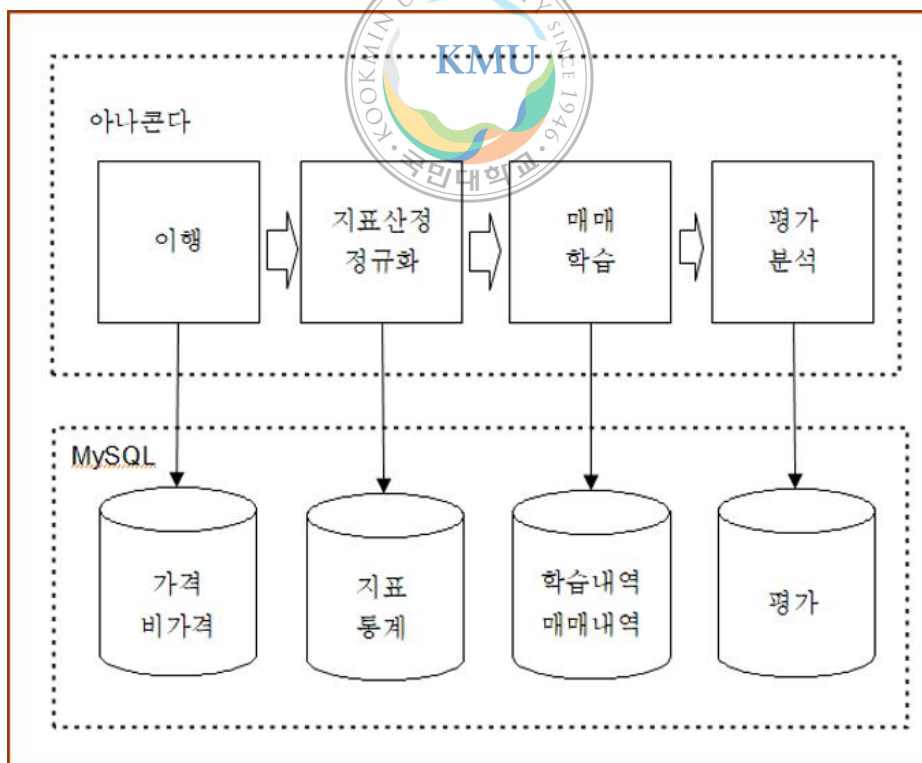
먼저 본 연구에서 제안하는 실험 모형을 소개하고 교차적 이상적 매매인 상향점/하향점 매매를 수행한다. 상향점/하향점 매매 데이터를 머신러닝 알고리즘의 학습 데이터를 만들고 위에 언급한 3개의 알고리즘으로 학습을 한다. 학습한 결과 성과가 가장 우수한 로짓회귀 비선형 방식의 테스트 데이터를 기준으로 일일 실전매매를 수행하여 수익률을 평가해 보고 이상적 매매의 한계점을 파악한다.

제 3장 실험실습

3.1 실험 모형 개요

실험을 위한 시스템의 구축은 DBMS는 MySQL과 아나콘다의 Spyder와 Jupyter Notebook을 이용하여 구현하였다. 아래 <그림7>은 전체적인 시스템 도메인이고 가격데이터의 이행적재 -> 지표산정 및 데이터 정규화 -> 매매 및 학습 -> 평가 및 Report의 과정을 거쳐서 구현하였다.

<그림7> 실험모형 시스템 구조도



실험모형은 크게 4개의 트랙으로 전개 했다. <표4>는 트랙별로 실험한 내용을 요약했으며 실험 내용은 모두 시스템화 구현과정을 거쳐서 진행한다.

실험트랙	실험내용
① 지표산정 및 정규화	가격데이터 적재 수익률 산정 입력변수 산정(이동평균가격,RSI, 거리) 정규화 – 가격데이터 Min-Max Scaled
② 교차점 매매 및 상향점/하향점 매매	매매 일자 별 교차점 매매 거래내역 일일 교차점 거래내역 매매 일자 별 상향점/하향점 거래내역 일일 상향점/하향점 거래내역
③ 알고리즘 학습	알고리즘 학습 설계 학습 알고리즘 전체 데이터 셋 검증 퍼셉트론, SVM , 로짓회귀 알고리즘 학습 학습 결과 기록
④ 실전매매 및 평가	로짓회귀 학습결과 테스트 데이터 검증 로짓회귀 일일 매매 수행 매매 수행 시 보류확률을 변경하면서 수행 매매 수행 내역 기록

<표5> 실험모형 단계별 실험 수행 내용

3.2 실험 모형

3.2.1 지표산정 및 정규화

■ 상대 강도 지수 (RSI)

상대강도 지수는 과거 일정한 기간 동안에 상승에 의한 변동분과 하락의 의한 변동분의 비율을 지수화 한 지표로 상승 변동분의 비율이 높으면 현재의 가격구간이 과매수 구간이고 낮으면 과매도 구간으로

매매신호를 활용하여 사용하는 지표이다. 이동평균의 상향구간과 하향구간에 잘 대응하는 보조지표로 본 연구에서는 교차점매매의 평균 주기에 근거해서 13일(매도평균일수), 16일(전체평균일수), 19일(매수평균일수)의 상대강도지수를 입력변수로 사용 한다.

$$RSI = \frac{\sum_i^n (\text{가격의 일일 상승분})}{\sum_i^n (\text{가격의 일일 상승분}) + \sum_i^n (\text{가격의 일일 하락분})}$$

상대강도 지수를 아래의 표에 맞춰서 산정 한다

거래일	일일상승	일일하락	상승합계	하락합계	RSI13
2019-05-02	9.16	0	5.25	6.84	43
2019-05-03	0	16.43	4.52	8.1	35
2019-05-07	0	19.33	4.08	9.59	29
2019-05-08	0	8.98	4.08	10.07	28
2019-05-09	0	66	4.08	12.68	24
2019-05-10	6.03	0	4.36	12.68	25

<표6> 상대 강도 지수 산정 예시

■ 가격간의 거리(distance)

머신러닝에서는 두 객체간의 유사성 정도를 나타내기 위해 척도가 필요한데 가장 일반적으로 많이 사용할 수 있는 것이 객체간의 거리(distance)이다. 거리가 클수록 유사정이 작아진다.

$$\text{민코프스키 거리 } d(x_i, x_j) = \left(\sum_{a=1}^p |X_{ai} - X_{aj}|^m \right)^{1/m}$$

여기에서 이차원일 때 유클리드거리를 나타내며 일차원일 때 맨하탄 거리(rectilinear distance)을 나타낸다. 일반적인 기계학습에서는 특별한 언급이 없을 때 유클리드 거리를 사용하는데, 본 연구에서는 가격의 절대 값의 차이 즉 객체의 1차원의 정량적 비교만 하기 때문에 맨하탄 거리를 입력변수로 사용한다.

맨하탄 거리 = 정규화 된 이평가격 - 정규화 된 종가

■ 가격 데이터의 정규화

머신러닝 알고리즘의 비선형의 특성은 데이터 단위를 표준화 하거나 단위를 동일한 의미로 식별하도록 일련의 표준화 작업을 사전에 해야 한다. 가령 가격지수 같은 경우 동일한 10%라고 하더라도 지수가 500이었을 때 510과 500의 차이와 2000과 2010의 동일한 10에 대한 차이가 발생된다. 이러한 의미와 단위를 통일하는 작업은 데이터 특성공학에서 아주 중요한 요소로 선형성을 극복하는 하나의 방법으로 일반화되고 있다. 본 논문에서는 가격데이터는 데이터 모두 Min-Max Scaled를 거쳐서 통일화 시켜서 진행한다.

$$\text{Scaled 가격} = \frac{\alpha - \min(\alpha)}{\max(\alpha) - \min(\alpha)}$$

<표7>은 종가 및 이동평균가격의 Min-Max Scaled 가격으로 변환하는 예시이다. 입력변수의 가격데이터는 이러한 정규화 변환 과정을 거쳐서 머신러닝 데이터로 입력 된다

거래일	종가	5일이평	scaled 종가	scaled 5일이평
2019-05-02	2212.75	2200.52	81.6	81.03
2019-05-03	2196.32	2201.68	80.84	81.09
2019-05-07	2176.69	2201.22	79.93	81.07
2019-05-08	2168.01	2191.53	79.52	80.61
2019-05-09	2102.01	2171.22	76.44	79.67
2019-05-10	2108.04	2150.27	76.72	78.69

<표7> 종가 및 이동평균가격 정규화 된 변환 가격

3.2.2 상향점/하향점 매매

상향점/하향점 매매를 수행하기 위해서는 교차점 매매와 교차점 매매에서 생성한 일일 교차점 매매내역을 기준으로 진행한다. 교차점 매매와 상향점/하향점 매매의 프로세스를 아래의 그림으로 요약하였다.



<그림8> 교차점 매매 프로세스 순서도

상향점/하향점의 매매를 진행하기 위해서 한 구간의 교차점 구간을 검색하여 교차점의 시작 구간부터 가격의 이동거리가 가장 큰 값이 매수 구간 시 상향점이 되고 매도 구간 시 하향점으로 결정한다.

가령 <그림9>에서 2018-09-18일부터 2018-10-08일이 교차점 매수구간일 경우 매수청산일(매도신규)은 2018-10-08일이 아니라 <표8>에서 보듯이 2018-09-18일을 기준으로 가격간의 거리가(sdailydis) 가장 큰 2018-09-27일로 빨라진다. 그 다음은 2018-11-14일이 교차점 매도청산일이지만 2018-10-08일을 기준으로 가격간의 거리가 가장 큰 2018-10-29일이 매도청산일(매수신규)이 된다.

상향점/하향점은 교차점의 한 구간을 먼저 검색하여 교차시작구간의 종가를 기준으로 이동평균가격 곡선이 가장 먼 거리를 산정하는 방식으로 식별하며 교차점의 매수구간은 매수청산(매도신규)이 되고 매도구간은 매도청산(매수신규)이 된다. 이렇게 산정한 상향점/하향점의 거래내역이 아패 <표9>의 내용이 된다.



<그림9> 5일/20일 이동평균선의 교차점 구간

itemcode	trdate	bstragubn	bsprice	seprice	sedate	behodday	sbprice	rs13	openprice	closeprice	supenprice	sdcloseprice	sdalyds
000001	2018-09-18	1	2308.98	2253.83	2018-10-08	10	86.09	50	2287.73	2308.98	85.1	86.09	0
000001	2018-09-19	1	2308.98	2253.83	2018-10-08	10	86.09	44	2319.22	2308.46	86.57	86.07	-0.02
000001	2018-09-20	1	2308.98	2253.83	2018-10-08	10	86.09	56	2314.41	2322.45	86.35	86.77	0.68
000001	2018-09-21	1	2308.98	2253.83	2018-10-08	10	86.09	58	2332.04	2339.17	87.17	87.9	1.41
000001	2018-09-27	1	2308.98	2253.83	2018-10-08	10	86.09	75	2331.7	2355.43	87.15	88.26	2.17
000001	2018-09-28	1	2308.98	2253.83	2018-10-08	10	86.09	70	2356.13	2343.07	88.29	87.68	1.59
000001	2018-10-01	1	2308.98	2253.83	2018-10-08	10	86.09	71	2349.64	2338.88	87.99	87.49	1.4
000001	2018-10-02	1	2308.98	2253.83	2018-10-08	10	86.09	56	2338.28	2309.57	87.46	86.12	0.03
000001	2018-10-04	1	2308.98	2253.83	2018-10-08	10	86.09	47	2311.06	2274.49	86.19	84.48	-1.61
000001	2018-10-05	1	2308.98	2253.83	2018-10-08	10	86.09	46	2269.94	2267.52	84.27	84.16	-1.93
000001	2018-10-08	2	2253.83	2068.05	2018-11-14	26	83.52	42	2258.73	2253.83	83.75	83.52	0
000001	2018-10-10	2	2253.83	2068.05	2018-11-14	26	83.52	27	2256.03	2228.61	83.62	82.34	1.18
000001	2018-10-11	2	2253.83	2068.05	2018-11-14	26	83.52	18	2176.16	2129.67	79.9	77.73	5.79
000001	2018-10-12	2	2253.83	2068.05	2018-11-14	26	83.52	25	2131.66	2161.85	77.82	79.23	4.29
000001	2018-10-15	2	2253.83	2068.05	2018-11-14	26	83.52	24	2155.34	2145.12	78.92	78.45	5.07
000001	2018-10-16	2	2253.83	2068.05	2018-11-14	26	83.52	20	2156	2145.12	78.96	78.45	5.07
000001	2018-10-17	2	2253.83	2068.05	2018-11-14	26	83.52	22	2169.44	2167.51	79.58	79.49	4.03
000001	2018-10-18	2	2253.83	2068.05	2018-11-14	26	83.52	17	2158.8	2148.31	79.09	78.6	4.92
000001	2018-10-19	2	2253.83	2068.05	2018-11-14	26	83.52	20	2130.06	2156.26	77.75	78.97	4.55
000001	2018-10-22	2	2253.83	2068.05	2018-11-14	26	83.52	21	2143.08	2161.71	78.35	79.22	4.3
000001	2018-10-23	2	2253.83	2068.05	2018-11-14	26	83.52	20	2147.3	2106.1	78.35	76.63	6.89
000001	2018-10-24	2	2253.83	2068.05	2018-11-14	26	83.52	21	2119.19	2097.58	77.24	76.23	7.29
000001	2018-10-25	2	2253.83	2068.05	2018-11-14	26	83.52	19	2046.67	2063.3	73.86	74.63	8.89
000001	2018-10-26	2	2253.83	2068.05	2018-11-14	26	83.52	18	2066.57	2022.15	74.78	72.94	10.58
000001	2018-10-29	2	2253.83	2068.05	2018-11-14	26	83.52	18	2026.68	1996.05	72.82	71.48	12.03
000001	2018-10-30	2	2253.83	2068.05	2018-11-14	26	83.52	30	1985.95	2014.69	71.02	72.36	11.16
000001	2018-10-31	2	2253.83	2068.05	2018-11-14	26	83.52	25	2022.84	2029.69	72.74	73.06	10.46

<표8> 일일 교차점 거래내역

itemcode	trxdate	bstrtype	bstrxgubn	bstregubn	bsprice	seprice	sedate	bshoday	prratio
000001	2018-09-27	6	2	1	2355.43	1996.05	2018-10-29	21	18
000001	2018-10-29	6	1	1	1996.05	2131.93	2018-12-03	26	6.81
000001	2018-12-03	6	2	1	2131.93	1993.7	2019-01-03	21	6.93
000001	2019-01-03	6	1	1	1993.7	2234.79	2019-02-27	37	12.09
000001	2019-02-27	6	2	1	2234.79	2128.1	2019-03-28	21	5.01
000001	2019-03-28	6	1	3	2128.1	2248.63	2019-04-16	14	5.66
000001	2019-04-16	6	2	1	2248.63	2023.32	2019-05-29	30	11.14
000001	2019-05-29	6	1	1	2023.32	2134.32	2019-06-27	21	5.49
000001	2019-06-27	6	2	1	2134.32	1909.71	2019-08-07	30	11.76
000001	2019-08-07	6	1	1	1909.71	2101.04	2019-09-24	32	10.02
000001	2019-09-24	6	2	3	2101.04	2021.73	2019-10-07	9	3.92
000001	2019-10-07	6	1	3	2021.73	2093.6	2019-10-28	15	3.55
000001	2019-10-28	6	2	3	2093.6	2093.6	2019-10-30	3	0.64
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

<표9> 상향점/하향점의 매매 내역

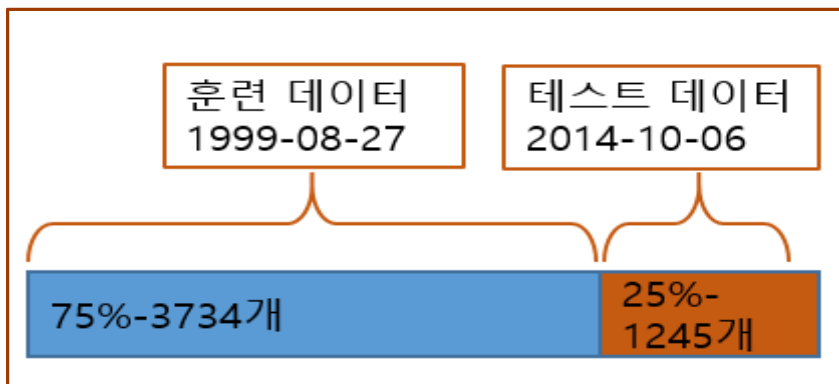
3.2.3 머신러닝의 학습

일반적으로 권고 하는 싸이킷 런 알고리즘 학습절차에 따라 학습을 수행했다. 대상 알고리즘은 퍼셉트론, SVM, 로짓회귀를 선택했다. <표10>는 작업 수행 절차에 대해서 요약했다.

학습 절차	학습 수행 내용
1. 학습 알고리즘 & 데이터 준비	전체 학습 데이터 검색
2. 훈련/테스트 데이터 분리	train_test_split 이용
3. 모델 학습 및 성과 측정	Perceptron, SVC, LogitRegression
4. 데이터 변형 및 재학습	데이터 차원 증가 비선형으로 전환
5. 파라미터 설정 재학습	마진, 커널, 복잡도 파라미터 설정

<표10> 알고리즘 학습 처리 절차

실험대상 데이터 구간은 아래 그림에서 나타난 것처럼 훈련 데이터 (75%)와 테스트 데이터 (25%)로 나누어서 지도학습으로 진행하였다. 지도학습은 1-매수, 2-매도로 설정하여 입력 데이터를 학습하고 그 결과를 싸이킷 런에서 제공하는 함수 score를 이용하여 측정하였다.



<그림10> 알고리즘 풀 데이터 셋 훈련/테스트 구간

아래 그림은 작업 수행한 내용을 제시한다. 표에서 제시한 절차대로 단계마다 수행한 작업내역을 그림으로 제시 하였다.

```
In [4]: from sklearn.linear_model import Perceptron, LogisticRegression

In [4]: moavr1[:5], moavr6[:5]

Out[4]: ( itemcode    trxdate bstrxgubn    s5closeprice    s5moavrpr    s20moavrpr    rsi13  #
0    000001    1999-08-27         1         23.17         22.54         21.87    47.0
1    000001    1999-08-30         1         22.35         22.77         21.83    49.0
2    000001    1999-08-31         1         22.13         22.82         21.75    47.0
3    000001    1999-09-01         1         20.62         22.28         21.58    47.0
4    000001    1999-09-02         1         20.31         21.72         21.49    47.0

    rsi16    rsi19    caratio    sdailydis
0    54.0    50.0         0.00         0.00
1    50.0    44.0        -1.84        -0.82
2    47.0    43.0        -2.33        -1.04
3    37.0    44.0        -5.70        -2.55
4    40.0    43.0        -6.38        -2.86 ,
```

<그림11> 학습알고리즘 선택 및 데이터 셋 준비

```
In [11]: from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, shuffle=False, test_size=0.25)

In [13]: X.shape, X_train.shape, X_test.shape, y.shape, y_train.shape, y_test.shape
Out[13]: ((4979L, 8L), (3734L, 8L), (1245L, 8L), (4979L,), (3734L,), (1245L,))
```

<그림12> 훈련/테스트 데이터의 분리

```
In [19]: perceptron = Perceptron(max_iter=100).fit(X_train, y_train)
perceptron.score(X_train, y_train), perceptron.score(X_test, y_test)

Out[19]: (0.7712908409212641, 0.6056224899598394)

In [18]: logreg = LogisticRegression(C=1.0).fit(X_train, y_train)
logreg.score(X_train, y_train), logreg.score(X_test, y_test)

Out[18]: (0.8077129084092126, 0.7775100401606426)
```

<그림13> 모델학습 및 성과측정

```
In [27]: from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=2)
poly.fit(X_train)

Out[27]: PolynomialFeatures(degree=2, include_bias=True, interaction_only=False)

In [28]: XX_train = poly.transform(X_train)
XX_test = poly.transform(X_test)

In [29]: X_train.shape, XX_train.shape
Out[29]: ((3734L, 8L), (3734L, 45L))

In [30]: logreg = LogisticRegression(C=1.0).fit(XX_train, y_train)
logreg.score(XX_train, y_train), logreg.score(XX_test, y_test)

Out[30]: (0.9268880557043385, 0.8329317269076305)
```

<그림14> 데이터 변형 및 재학습


```

In [32]: C_range = [0.001, 0.01, 0.1, 1., 10., 100., 1000.]
          tr_result = {}
          for C in C_range:
              logreg = LogisticRegression(C=C).fit(X_train, y_train)
              tr_score = logreg.score(X_train, y_train)
              te_score = logreg.score(X_test, y_test)
              tr_result[C] = {'train': tr_score, 'test': te_score}

In [33]: tr_result_tbl = pd.DataFrame(tr_result).T
          tr_result_tbl

```

Out[33]:

	test	train
0.001	0.806426	0.900643
0.010	0.829719	0.922871
0.100	0.846586	0.928227
1.000	0.832932	0.926888
10.000	0.834538	0.925013
100.000	0.842570	0.929566
1000.000	0.842570	0.926888

<그림15> 파라미터 설정 및 재 학습

3.2.4 로짓회귀 매매

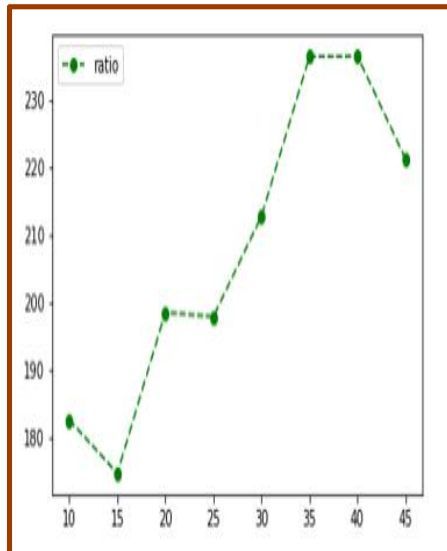
로짓회귀 알고리즘에서 테스트 데이터의 성과내역을 입력으로 일일 실전 매매하여 수익률을 평가했다. 로짓회귀 알고리즘은 학습결과로 확률 값이 제공되므로 확률 값에 근거해서 매매를 결정하거나 전일 매매를 유지(hold)할 수 있다. 전일 매매를 유지하는 방식은 상향/하향 밴드를 설정하여 그 사이에 확률 값이 들어오면 전일의 포지션을 유지하고 밴드 바깥에는 매수 또는 매도 포지션을 취하면서 매매를 진행했다.

매매를 진행하기 위해 확률 값 밴드를 10에서부터 5씩 증가하여 /10-90/에서/45-55/까지 위 밴드별로 매매를 진행하였고 수익률을 산정하였다.

```
In [51]: tr_result_tbl = pd.DataFrame(tr_result).T
tr_result_tbl
```

Out[51]:

	ratio
10	182.52
15	174.60
20	198.47
25	197.82
30	212.65
35	236.35
40	236.37
45	221.17



<그림16>밴드별 수익률 및<그래프



제 4장 결론

4.1 실험실습 결론

■ 알고리즘 기계학습 성과 측정

3개의 알고리즘 지도 학습의 결과표 이다. 로짓회귀의 비선형 알고리즘이 SVM의 비선형 알고리즘보다 지도학습의 분류 성과가 우수하다. 선형적인 형태의 성과는 3개의 알고리즘 모두 낮지만 데이터 특성과 자체의 비선형 옵션으로 (SVM의 커널함수) 전환한 후에는 상당한 차이가 있음을 알 수 있었다.


알고리즘	알고리즘클래스	커널함수유형	Complexity	훈련	테스트	비고
퍼셉트론	Perceptron			77.12	60.56	선형
SVM	SVC	RBF		100	61.47	선형
			0.001	58.35	59.27	비선형
			0.1	76.19	71.72	
			1	79.03	72.2	
			10	83.47	71.72	
			100	87.92	75.26	
			1000	91.8	79.51	
			10000	93.25	78.63	
		poly	0.001	58.35	59.27	비선형
			0.1	58.35	59.27	
			1	76.88	72.2	
			10	77.42	71.8	
			100	81.7	71.48	
			1000	85.67	72.44	
			10000	90.19	77.75	
로짓회귀	LogitRegression			80.77	77.75	선형
			0.001	90.06	80.64	비선형
			0.1	92.28	82.97	
			1	92.82	84.65	
			10	92.5	83.45	
			100	92.95	84.25	
			1000	92.68	84.25	

<표11> 알고리즘별 분류예측 성과 결과표

알고리즘에서는 SVM보다 로짓회귀의 비선형 분류가 훨씬 높은 수익률을 보여줬다. 테스트 구간에서 82%이상의 안정적인 분류는 상향점/하향점을 식별하는데 의미 있는 성과이다.

■ 로짓회귀 알고리즘의 매매 수익률

<표12>는 로짓회귀 알고리즘의 기계학습의 결과로 분류/예측된 데이터를 입력으로 실전 매매한 수익률 결과표이다. 일반 단순수익률과 상향점/하향점 매매 수익률과 비교를 해 봤다. 의미 있는 수익률 지표를 얻을 수 있었다. 2014년 10월 24일 이후 테스트 구간의 상향/하향점 수익률, 교차점 수익률, 단순 수익률(일일 수익률의 합계)을 비교하여 작성하였다. 로짓회귀 알고리즘에서 5일 이동평균선과 20일 이동평균선의 상향점/하향점 데이터 분포를 학습한 후에 일일 매매한 수익률이다. 밴드구간은 40-60의 확률값이 제시되면 매매를 보류하는 방식으로 포지션을 유지 하였다.



연도	로짓알고리즘매매수익률	상향하향점수익률	교차점수익률	단순수익률
2014	-0.21	5.53	-2.31	0.90
2015	60.43	95.28	-16.96	3.14
2016	27.01	60.44	-15.62	4.02
2017	32.39	56.42	15.54	20.05
2018	47.38	83.39	-4.20	-17.90
2019	69.00	69.28	18.02	2.58
합계	236.00	370.34	-5.53	12.79

<표12>로짓회귀 알고리즘의 매매 수익률(2019/10/30일 기준)

4.2 연구 한계 및 방향성

이동평균선 교차적 전략의 이상적 매매는 엄청난 승률과 수익률에도 현실 세계에 존재 할 수 없는 한계점이 있다. 즉 한 구간의 교차구간

이 식별 되어야만 그 구간에서 가장 높거나 낮은 구간을 찾을 수 있기 때문에 오늘을 트레이딩 하는 우리는 지금 현재의 상향점/하향점인지를 파악하는데 제약이 있다.

즉 오늘이 교차점이었다면 상향점/하향점은 과거의 어느 시점이 될 것이다. 오늘 현재의 기준에서는 확인을 할 수 없는 한계가 있다. 바로 이 부분이 본 연구 논문의 한계이자 새로운 과제의 시작점이라고 생각한다. “상향점/하향점을 현실세계에서 식별하는 문제”로 좁혀 본다면 이후 연구과제의 방향은 명확해진다.

기술적 지표를 활용하여 상향점/하향점을 식별해 볼 수 있다. 볼린저 밴드의 상단/하단의 가격이 이탈하는 경우와 얼마나 유의적인 관계가 있는지? 수급주체별 거래량도 어떤 관계가 있을까? MACD의 기울기가 정점은 어떤가? 일단 상향점/하향점의 기준(좌표)은 정해 졌는데 현실의 가격변화를 가지고 움직이는 변수가 어느 정도 관계를 갖고 있는지 파악하는 영역이 기술적 지표를 활용해 볼 수 있는 요소이다.

통계적 데이터 분석 방법도 생각해 볼 수 있다. 상향점/하향점의 변곡의 데이터의 표본을 충분히 확보하여(다우, 니케이, 닥스, 상해등) 현재의 분포구간이 상향점과 하향점에서 얼마나 벗어나 있는지를 확률적 방법으로 파악해 볼 수 있고 분산분석의 방법도 생각해 볼 수 있다.

트레이딩의 다양한 지표를 활용하여 더 고도화된 머신러닝 알고리즘과 통계적 방법론을 활용하여 현재의 시간에서 이동평균선의 상향점/하향점을 식별하는 과제가 본 연구의 한계점이자 새로운 연구의 출발점이 될 수 있다.

참고문헌

- [1] 김선웅, 안현철 (2010), Support Vector Machine와 유전자 알고리즘을 이용한 지능형 트레이딩 시스템 개발
- [2] 박성철 (2010), KOSPI200 선물투자에서의 CHMM기반 지능형 매매시스템 개발
- [3] 오정환 (2013), 코스피200 변동성 지수 이동평균선을 이용한 옵션투자 전략 모형
- [4] 장재건 외, 기술적 분석지표를 이용한 선물투자기법, p.105 - p.126
- [5] 안드레아스 뮐러, 세라 가이도(2019), Introduction Machine Learning with Python
- [6] 오렐리앙 제롱(2018), Hands-On Machine Learning with Scikit-Learn & TensorFlow, p.214 - p.220, p.332 - p.334
- [7] 전치혁(2018), 데이터마이닝 기법과 응용, p.193 - p.195, p.336 - p.339

Abstract

A Study on the Ideal Trade of Moving Average Line Crossing Strategy using Machine Learning

by Kim, Seok Joon

Department of Trading Systems
Graduate school of Business-IT,
Kookmin University,
Seoul, Korea



The cross-strait strategy of moving averages is one of the most frequently used technical indicators in the field. It is the easiest to understand, easy to break the trend of prices, and a variety of strategies along the moving average line. If only one five-day moving average is used, a regression strategy can be employed as the average line when prices move away from the average line. Short-term moving averages are also highly variable compared to long-term moving averages, which can be used to periodically cross two curves, which can be purchased in case of an upward breakthrough or sold in case of a downward break.

However, the cross-strait strategy of the mobile mean line is incorrect for the sale signal because of time lag. The direction of the position is set, which often produces a trade signal opposite

to the direction of progress in which the price moves. The return on the KOSPI index for cross-trading is -46.48% (target period: 1999-07-30 to 2019-10-30). Compared to the simple return on the same target period of 115%, the cross-curve sales signal is inaccurate.

This study determines the highest (upward) and lowest (downward) points in a section of cross-point sales. The point at which the physical distance from the intersection is the upper or lower point, and the lower point is the selling (purchasing) and the buying (purchasing) is the lower point. This method of trading is called the "up/downward selling" of the mobile average and records the return on sales of 2,712.39 percent during the same period as the cross-point selling rate of the KOSPI. It is the most ideal sale of a moving average that produces extreme returns.

The purchase/ask position is studied by selecting the linear/nonlinear algorithm of machine learning. In this study, we select the Perceptron, SVM, and Logit-return algorithms to proceed with learning in a linear/nonlinear fashion. Among the above algorithms, the logit return algorithm presents the classification/forecast results as probabilities, so the learned probability result values are sold daily to calculate the return.

The trade-return rate of the test section studied by the logit regression algorithm was 236%, the bottom-up/lower-down-point-selling rate was 370%, the cross-point return was -5.53 and the simple return rate was 12.79 and a significant meaningful return was tested.

The ideal sale of cross-curves, the up/down-market sale, cannot exist in the real world. This is because the inflection point of the intersection must be at least one segment before the inflection point of the intersection can be identified. Identifying ideal intersections in the real world, which cannot exist in the real world, is the limit of this paper and a later research task.

Keyword : Machine Learning, Logit Regression, SVM

