Hanbit RealTime 127

머신러닝을 이용한

알고리즘 트레이딩 시스템 개발

안명호, 류미현 지음

한빛미디어 Hanbit Media, Inc.

머신러닝을 이용한

알고리즘 트레이딩 시스템 개발

안명호, 류미현 지음

한빛미디어 Hanbit Media, Inc.

표지 사진 **임희진**

이 책의 표지는 임희진님이 보내 주신 풍경사진을 담았습니다.
리얼타임은 독자의 시선을 담은 풍경사진을 책 표지로 보여주고자 합니다.

사진 보내기 **ebookwriter@hanbit.co.kr**

머신러닝을 이용한 알고리즘 트레이딩 시스템 개발

초판발행 2016년 4월 27일

지은이 안명호, 류미현 / **펴낸이** 김태헌
펴낸곳 한빛미디어(주) / **주소** 서울시 마포구 양화로 7길 83 한빛미디어(주) IT출판부
전화 02-325-5544 / **팩스** 02-336-7124
등록 1999년 9월 30일 제10-1779호
ISBN 978-89-6848-803-0 15000 / **정가** 14,000원

총괄 전태호 / **책임편집** 김창수 / **기획·편집** 정지연
디자인 표지/내지 여동일, 조판 최송실
마케팅 박상용, 송경석, 변지영 / **영업** 김형진, 김진불, 조유미

이 책에 대한 의견이나 오탈자 및 잘못된 내용에 대한 수정 정보는 한빛미디어(주)의 홈페이지나 아래 이메일로 알려주십시오.
한빛미디어 홈페이지 www.hanbit.co.kr / **이메일** ask@hanbit.co.kr

Published by HANBIT Media, Inc. Printed in Korea

지금 하지 않으면 할 수 없는 일이 있습니다.
책으로 펴내고 싶은 아이디어나 원고를 메일(ebookwriter@hanbit.co.kr)로 보내주세요.
한빛미디어(주)는 여러분의 소중한 경험과 지식을 기다리고 있습니다.

저자 소개

지은이_ **안명호**

KAIST SW석사과정을 마쳤다.

어느 날 알게 된 머신러닝에 흠뻑 빠져 그동안 애지중지하던 클라우드를 버리고 머신러닝으로 전향하였다. 이제 더는 다른 기술은 관심을 두지 않고 머신러닝 한길만으로 정했기에 머신러닝을 공부하며 어려운 수식들을 다시 보느라 고생하고 있지만, 하루하루 배워가는 지식에 행복해하며 지내고 있다. 머신러닝으로 가마우지를 만들어 인생을 즐기려 노력하고 있으며, 그 결실이 완성되는 날 완벽한 경제적 자유를 누리고자 한다.

- **Facebook** https://www.facebook.com/james.ahn.9
- **Homepage** http://www.deepnumbers.com

지은이_ **류미현**

동국대 정보공학석사를 마쳤다.

머신러닝 책을 집필할 때만 해도 쉬엄쉬엄하는 마음으로 했는데, 딥마인드DeepMind의 데미스 하사비스Demis Hassabis가 '알파고AlphaGo'를 서울에 데려와 파문을 일으키고 가는 바람에 왠지 모를 조바심이 생겼다. 그동안 딥러닝에 대해 갸우뚱하던 생각도 무지의 소치로 치워두고 급상승한 호기심을 발판으로 깊숙이 들어가 보려 한다. 확신할 수 없을 때는 불안하고 머뭇거리게 되지만 다행히 누군가 그 길을 보여주면 그때라도 놓치지 말고 따라가는 게 낫지 않을까 생각한다.

패러다임의 변화, 소프트웨어에서 데이터로

바야흐로 데이터의 시대로 발전해가고 있다. 얼마 전까지만 하더라도 소프트웨어의 중요성을 강조하고 소프트웨어가 세상을 먹어치우고 있다는 다소 도발적인 문구가 많이 회자했다. 언론에서도 이에 발맞추어 소프트웨어가 우리의 삶과 산업에 미치는 영향을 보도하고 새로운 소프트웨어 기술을 소개하며 영향력 있는 개발자들의 인터뷰를 실었다. 당연히 소프트웨어가 중요하다는 점에서는 이견이 없다. 지금도 그렇고 앞으로도, 말할 나위 없이 소프트웨어는 중요하다.

이야기하고 싶은 것은 경쟁력과 가치다. 오픈소스의 확산은 소프트웨어 발전에 기념비적인 사건이라 할 수 있다. 이는 오픈소스가 소프트웨어에서 데이터로 패러다임을 변화시키는 핵심 동인이기 때문이다. 과거에는 소프트웨어 자체가 경쟁력이 될 수 있었다. 지금처럼 소프트웨어의 수가 많지도 않았고, 소스 코드가 공개된 소프트웨어의 수도 많지 않았다. 또한, 사용하는 소프트웨어의 기능과 품질이 우수할수록 경쟁력이 있었다.

현재는 오픈소스로 인해 사용할 만한 소프트웨어가 넘쳐나고 있다. 간단한 기능을 수행하는 오픈소스에서, 거대한 IT 서비스를 운영하는 데 사용할 정도로 강력한 오픈소스에 이르기까지 다양한 분야에 걸쳐있다.

더구나 최근의 오픈소스 프로젝트는 과거의 것과 비교할 수 없을 정도로 수준이 높아졌다. 구글, 페이스북처럼 전 세계 IT 산업을 이끄는 회사에서 자사의 필요로 개발한, 자사의 서비스에 사용하는 고품질, 고기능의 소프트웨어를 오픈소스로 앞다퉈 공개하고 있다. 얼마 전 구글에서 오픈소스로 공개한 TensorFlow[01]는 딥러닝Deep Learning 라이브러리로, 구글에서 머신러닝 관련 작업, 예를 들어 이미지 인식 등에 사용하는 것으

01 https://www.tensorflow.org/

로 기능이 떨어지거나 학습결과물의 품질이 떨어지는 조악한 것이 절대 아니다.

이처럼 오픈소스의 확산은 소프트웨어 자체가 더는 경쟁력이 될 수 없는 환경을 만들었다. 머신러닝을 구현하고 싶을 때 구글의 TensorFlow를 이용하면 세계 최고 수준의 머신러닝 라이브러리를 사용할 수 있다. 실시간 빅데이터 분석을 하고 싶을 때 링크드인의 Pinot[02]을 이용하면 역시 세계 최고 수준의 실시간 빅데이터 분석 프로그램을 쉽게 개발할 수 있다. 이렇게 몇 번의 검색만으로 최고 수준의 소프트웨어를 쉽게 얻는 환경이 되었다.

이런 환경에서 경쟁력은 이제 한 단계 끌어올려 져 '지능화'될 수밖에 없다. 기존의 소프트웨어가 어떤 기능을 제공하는 것이라면 지능화를 갖춘 소프트웨어는 자동화를 통해 번거로움을 없애주고, 판단을 통해 새로운 가치를 전달할 수 있게 된다. 이 책의 내용에 빗대어 주식으로 예를 들면, 기존의 주식 관련 소프트웨어는 "주식거래를 간편하게, 빨리, 원하는 시점에 할 수 있다"라는 기능 중심의 가치를 제공한다면, 지능화를 갖춘 소프트웨어는 "지금 xx 주식을 사면 10%의 이익을 볼 수 있다"라는 편익 중심의 가치를 제공한다.

지능화를 위해 필요한 것이 데이터고, 머신러닝을 이용하면 데이터를 지능으로 바꿀 수 있다. 대표적인 머신러닝 기술 중 하나인 딥러닝이 이에 대한 좋은 예다. 딥러닝은 1943년에 발표된 신경망Neural Network이라는 기술에 기반을 둔 것으로 2000년대 초반까지만 하더라도 낮은 성능과 기술상의 제약으로 인해 거의 사장되었었다. 하지만 2000년대 후반에 딥러닝으로 화려하게 부활했다. 알고리즘의 발전도 이에 중요한 역할을 했지만, 컴퓨팅 파워Computing Power와 특히 데이터가 없었다면 불가능했다.

머신러닝을 위해 필요한 요소는 알고리즘, 컴퓨팅 파워, 데이터 3가지다. 알고리즘은

02 https://github.com/linkedin/pinot/wiki

인터넷을 통해 비교적 쉽게 구할 수 있고, 컴퓨팅 파워 역시 비용만 있다면 아마존과 같은 퍼블릭 클라우드를 사용해 쉽게 해결할 수 있다.

하지만 앞의 2가지와 다르게 데이터는 그렇지 않다. 신경망을 비롯한 머신러닝 알고리즘들은 많은 데이터가 있을수록 더욱 좋은 성능을 보이기 때문에 양질의 데이터를 많이 가질수록 유리하다. 인터넷으로 인해 이전보다는 데이터를 구하기 쉬운 것은 사실이지만 데이터 오너십Data Ownership의 문제로 인터넷을 통해 구할 수 있는 데이터는 한정되어 있고, 신상정보나 카드사용명세처럼 중요한 가치를 가지고 있는 정보는 공개되지 않은 경우가 대부분이다.

결국, 쓸만한 양질의 데이터를 누가 많이 가지고 있느냐에 따라 지능화의 품질이 달라지는 것이다. 관심도 없는 제품광고를 시도 때도 없이 보내는 서비스보다는 내가 사고 싶은 물건에 대한 할인 광고를 보여주는 서비스가 사람들로부터 사랑받듯이, 앞으로의 싸움은 기능이 아닌 편익 중심으로 펼쳐질 수밖에 없고 여기에서 핵심은 단연코 데이터가 된다.

금융과 IT 그리고 머신러닝

금융은 숫자를 다루는 산업으로 IT 기술이 매우 중요한 역할을 한다. 입금하고 출금하고 내 계좌의 돈이 얼마나 있는지 확인하고 다른 사람에게 송금할 때, 이 모든 것이 IT 기술로 실행된다. 내가 송금한다고 해서 실제의 물리적 존재인 돈이 송금받는 사람에게 전달되는 것은 아니다. 돈을 찾기 전까지는 IT 시스템 어딘가의 데이터베이스에 기록되어 있는 나의 예금이 송금 액수만큼 줄어들고 송금받는 사람의 데이터베이스 기록의 예금이 해당 액수만큼 늘어나는 것이다.

IT 기술이 없다면 지금 우리가 누리는 여러 가지 금융서비스는 시간이 오래 걸리거나

불가능한 것이 많을 것이다. 그동안 금융에서는 IT 시스템을 금융거래에 관련된 데이터의 기록, 삭제, 수정 등의 기능 수행 목적으로 많이 활용했다. 하지만 최근에는 금융에서 IT 시스템을 한 차원 높은 수준으로 바라보기 시작했다.

간단히 말하면 머신러닝을 이용한 지능화라고 할 수 있다. 금융기관에서 가지고 있는 막대한 양의 금융 데이터를 단순 데이터베이스처럼 사용하는 것이 아니라, 이들 데이터에서 머신러닝을 이용해 위험도를 측정하고 앞으로 주가의 방향을 예측하고 새로운 사업기회를 찾는 등 이전과는 다른 시각과 기술로 활용하기 시작했다.

머신러닝은 많은 데이터가 필요하고 강력한 컴퓨팅 파워가 뒷받침되어야 하는데, 금융계는 이러한 조건들을 만족하는 환경을 이미 가지고 있다. 금융 데이터는 수치데이터고 여타의 데이터보다 정확해서 머신러닝에 적용하기 매우 좋은 조건을 가지고 있다. 예를 들어, 소셜분석을 하고자 하면 사람들이 작성한 글을 분석해 적절한 수치로 변환한 후에 머신러닝 알고리즘을 이용해 분석하거나 예측할 수 있는데, 분석하는 과정과 수치로 변환하는 과정에서 어느 정도의 손실과 오류는 필연적이다. 더구나 소셜분석으로 글에 쓰인 내용이 글쓴이의 생각을 100% 대변하는 것인지, 아니면 다른 이유로 그렇지 않은지를 정확하게 알기 힘들다.

이러한 흐름을 잘 보여주는 것이 세계적인 투자은행인 골드만삭스다. 비지니스인사이더의 2015년 4월 12일 기사를 보면 'Goldman Sachs is a tech company'[03]라는 제목으로 골드만삭스의 변신에 대해 자세히 소개하고 있다.

투자은행인 골드만삭스의 전 세계 직원 수는 33,000명인데, 이 중 9,000명이 엔지니어와 프로그래머라고 한다. 즉, 전 직원의 30%가 IT 관련 직종이다. 페이스북의 IT 관련 전체 직원 수가 9,199명이고 트위터의 전체 직원 수가 3,638명, 링크드인이

03 Jonathan Marino (2015). Goldman Sachs is a tech company. http://goo.gl/UruxtF

6,897명이라는 것을 생각해보면 골드만삭스의 IT 관련 인원이 얼마나 많은 수인지를 쉽게 짐작할 수 있다.

금융계에서의 대표적인 머신러닝 활용은 알고리즘 트레이딩Algorithmic Trading이다. 알고리즘 트레이딩은 사람이 아닌 프로그램에 의해 거래하는 것으로 이미 미국에서는 2012년에 전체 거래의 85%를 차지할 정도로 일반화되었다. 우리가 흔히 주식시장에서 얘기하는 외국인은 사람이 아닌, 기관이나 헤지펀드에서 사용하는 알고리즘 트레이딩 프로그램이라고 생각해도 결코 과언이 아니다.

금융계에서의 IT 활용 확대에 따라 이제는 전 세계 금융의 메카인 월스트리트에서도 MBA나 재무, 경제를 전공한 전통적인 이미지의 금융맨들이 아닌, 너무도 이질적으로 보이는 수학자, 천문학자, 물리학자, 컴퓨터과학자를 찾는 것이 쉬운 일이 되어버렸다. 머신러닝과 같은 IT 기술을 활용하는 것이 금융기관의 수익을 높여줄 수 있을 뿐만 아니라, 새로운 사업기회를 찾을 수 있고 일의 효율성을 극대화한다는 것이 오랫동안 증명되었기 때문에 금융과 머신러닝 같은 IT 기술의 조우는 더욱 많아질 것이다.

이 책을 집필한 이유

처음 머신러닝을 접했을 때 환상적인 개념과 놀라운 결과에 무척 흥분했었다. 예전에 수행한 프로젝트에서 물체를 인식하는 프로그램을 개발했었는데, 무척이나 힘들고 괴로운 과정이었다. 사람의 눈으로는 너무 쉽게 구별되는 물체를 프로그램으로 컴퓨터에 인식하게 하려니 정말 많은 수학 지식이 필요했고, 이를 코드로 구현하기도 쉽지 않았다.

물체 인식을 위한 모든 것을 하나하나 코드로 프로그램에 넣어야 했고, 발생할 수 있는 예외상황과 이미지 특성들도 역시 포함해야 했다. 다른 형태를 지닌 물체를 인식하게

하려면 앞서 했던 고생을 새롭게 반복해야 했다. 이런 고생을 하더라도 결과가 좋으면 행복했겠지만, 현실은 냉정했다. 특정한 조건에서는 인식률이 높았지만, 그 조건을 조금이라도 벗어나는 환경이 되면 인식률은 곤두박질하기가 부지기수였다.

그런데 머신러닝 예제로 여기저기서 쉽게 볼 수 있는 'MNIST Digit Recognition'을 보면 너무도 쉽게 숫자를 인식한다. 예제 코드도 길지 않은데, 정확도가 95~98%에 이르니 예전 기억을 떠올려보면 놀라운 세계였다. 그래서 머신러닝에 대한 설레는 기대감으로 여행을 시작했다. 어느 여행이나 마찬가지지만 여행안내서에 나오는 그 환상적인 절경들은 그냥 쉽게 만나는 것이 아니라는 것을 깨닫는 데 오랜 시간이 걸리지 않았다. 역시 예제는 예제일 뿐이라는 것을 다시 한 번 확인하는 계기가 되었다. 하지만 머신러닝에 대한 강한 끌림으로 여행을 포기하고 싶지 않았기 때문에 본격적인 공부를 시작했다.

필자의 기준으로 보면 머신러닝 책들은 크게 2가지 분류로 할 수 있었다. 첫 번째는 머신러닝에 대한 소개 차원의 책과 두 번째는 머신러닝의 수학적 배경을 설명하는 책이다. 전자의 책은 머신러닝 알고리즘에 대한 소개와 사용방법에 치중하여 머신러닝 사용법을 학습하는 데는 도움이 되었지만, 머신러닝의 개념을 이해하고 무엇을 하기에는 내용이 부족하다는 생각을 지울 수 없었다. 후자의 책은 반대로 너무 전문적이어서 머신러닝 알고리즘에 사용된 수학적 개념들과 각종 정리를 소개하는 데 치중했다. 머신러닝 알고리즘을 새로 개발하거나 혹은 기존 머신러닝 알고리즘을 개선해서 사용하려 한다면 이런 책들이 도움되지만, 머신러닝 알고리즘을 사용하는 데 중점을 둔 입장에서는 너무 지나친 감이 있었다.

목마른자가 우물을 파는 심정으로 머신러닝을 묵묵히 공부해본 결과, 머신러닝을 활용하는 데는 핵심적인 수학 개념에 대한 이해와 적용하려는 분야에 대한 도메인 지식 Domain Knowledge이 중요하다는 것을 깨닫게 되었다.

사실 머신러닝을 이용해 프로그램을 작성하는 데 머신러닝 알고리즘이 차지하는 비중은 그렇게 크지 않다. 중요한 것은 데이터에 대한 이해와 특성을 파악하는 것이다. 이런 과정을 데이터 탐색 분석EDA, Explonatory Data Analysis이라고 하는데, 이 과정을 제대로 수행하기 위해서는 통계와 확률에 대한 수학적 지식이 요구된다. EDA를 효과적으로 수행하는 데 도메인 지식이 있다면 시간을 대폭 단축할 수 있고, 문제를 단순화할 수 있다. 이런 과정을 거쳐 적용한 머신러닝이야말로 좋은 결과를 보여준다.

머신러닝을 공부하는 데 필요한 수학적 이론과 도메인 지식, 이를 구현한 코드를 한곳에서 볼 수 있는 책이라면 도움이 되지 않을까 하는 생각으로 이 책을 집필하게 되었다. 머신러닝을 제대로 활용하기 위해서는 데이터가 더욱 중요한 이슈라고 얘기했듯이, 머신러닝을 이야기하는 데 적용분야인 도메인을 정하지 않는 것은 반쪽짜리다. 그래서 적용대상으로 쉽게 데이터를 얻을 수 있으며, 데이터 자체에 대한 신뢰도가 높고 난도가 있는 주식을 선택했다.

주식시장은 예측이 어려운 대표적인 분야로, 머신러닝을 공부하는 데 필요한 모든 요소를 가지고 있다고 할 수 있다. 수학이론을 이용한 예측 모델의 작성, 작성한 모델을 위한 데이터 처리, 주가를 예측하기 위한 학습, 학습결과의 해석과 이를 개선하는 방법 등 머신러닝 전체 흐름을 경험하기에 좋다.

통계, 시계열Time Series, 알고리즘 트레이딩 등 책에서 다루는 주제는 각각 한 권의 책으로도 분량이 모자랄 만큼 방대한 내용을 가지고 있으나, 이 책의 목적상 알고리즘 트레이딩과 직접 연관되고 반드시 알아야 하는 사항들을 중심으로 서술했다.

부족한 내용은 별도의 책을 통해 지식을 습득해야 한다. 아울러 프로그래밍을 할 줄 아는 독자들을 대상으로 하므로 프로그래밍에 관련된 설명은 특별히 하지 않았다.

이 책의 구성

이 책은 크게 3부분으로 구성되어 있다.

Part 1은 머신러닝의 개요로, 머신러닝이 무엇인지와 머신러닝이 할 수 있는 것은 무엇인지, 머신러닝의 종류는 무엇이 있는지 등 머신러닝의 전반적인 개요를 설명한다.

1장 머신러닝이 첫 번째 Part인데, 특히 중요한 의미가 있는 내용은 머신러닝에 대한 개념 파악과 머신러닝이 해결할 수 있는 문제, 머신러닝 프로세스, 마지막으로 'No Free Lunch Theorem'이다.

Part 2는 알고리즘 트레이딩을 위한 수학적 배경지식으로 통계와 시계열을 다룬다. 알고리즘 트레이딩을 하려면 주식의 매도와 매수를 결정하는 '모델'이라는 것을 만들어야 한다. 이 모델을 만들기 위한 최소한의 통계 개념과 시계열 개념을 설명한다.

2장 통계는 모든 머신러닝의 가장 기초가 되는 것으로, 머신러닝을 올바르게 사용하고 활용하기 위해서 반드시 이해하고 넘어가야 하는 개념들인 표준편차, 히스토그램, 정규분포 등을 설명했다.

3장 시계열 데이터는 알고리즘 트레이딩의 이론적 토대가 되는 수학적 개념들을 소개한 것으로, 마지막 알고리즘 트레이딩 구현을 위해 필요한 내용이 최소한으로 담겨 있다.

Part 3은 실제로 간단한 알고리즘 트레이딩을 파이썬을 이용해 구현해본다. 머신러닝에 기반을 둔 모델과 시계열 이론에 기반을 둔 모델 2가지를 구현해보고, 구현된 결과에 대한 해석과 개선방법에 대해 다룬다.

4장 알고리즘 트레이딩에서는 국내에서 낯선 개념인 알고리즘 트레이딩이 무엇인지 소개하고, 알고리즘 트레이딩에 사용할 2가지 모델을 설명한다.

5장 알고리즘 트레이딩 시스템 구현은 파이썬과 라이브러리를 이용해 4장에서 설명한 2가지 모델을 실제로 구현해본다.

6장 성능 평가와 최적화는 5장에서 구현한 모델의 예측 성능을 평가하는 방법과 더 높은 예측 성능을 위해 어떻게 최적화해야 하는지를 설명한다.

이 책의 내용을 실습하는 데 필요한 소프트웨어와 하드웨어

머신러닝을 하려면 당연히 소프트웨어와 하드웨어가 필요하다. 머신러닝을 위한 하드웨어는 되도록 빠른 속도를 가진 CPU를 사용하는 것이 좋다. 머신러닝 알고리즘은 다른 소프트웨어보다 계산을 많이 하기 때문에 실행시간이 적게는 수 초에서 길게는 수 개월이 걸릴 수 있어서 CPU가 빠를수록 결과를 빨리 얻을 수 있다.

머신러닝을 본격적으로 해보고 싶다면 CPU보다는 GPU 사용을 강력히 추천한다. GPU는 수학과 과학 계산에 특화된 장치로, 매우 빠른 계산 속도를 보여준다. CPU는 직렬처리를 하지만, GPU는 병렬처리를 하므로 동일한 시간이라도 병렬처리하는 GPU의 수행속도가 훨씬 빠르다. 최신 CPU는 4코어, 8코어를 가지고 있지만, GPU는 수천 개의 코어를 가지고 있어 병렬로 처리했을 때 GPU는 동일한 시간에 수천 개의 코어로, 수천 개의 계산을 동시에 처리할 수 있다.

소프트웨어의 선택은 역시 하드웨어만큼 중요하다고 할 수 있다. 하드웨어는 사용하다가 더 좋은 성능이 필요하다면 업그레이드로 쉽게 해결할 수 있지만, 소프트웨어는 사용하는 라이브러리와 개발한 머신러닝 코드가 언어와 라이브러리에 종속되므로 신중한 선택이 필요하다.

과거에는 소프트웨어의 선택에서 속도라는 측면을 중요하게 여겼다. 엄청나게 많은 계산을 해야 하는 머신러닝은 실행시간이 길어서 오랜 시간 기다리는 것보다는 코드

의 결과를 되도록 빨리 보는 것이 의미가 있었기 때문이었다. 그래서 C, C++과 같은 언어를 많이 사용했다.

하지만 지금은 컴퓨팅 파워가 예전보다 많이 향상되었고, 단시간에 강력한 컴퓨팅 파워가 필요하다면 클라우드를 사용할 수도 있다. 결정적으로 GPU의 등장으로 속도의 매력이 이전보다는 상실되었다(이미지 인식 같은 분야는 아직도 많은 수행시간이 필요해 C++를 선호하기도 한다) 따라서 최근에는 라이브러리가 소프트웨어 선택에 중요한 이유가 되고 있다.

머신러닝 라이브러리는 필요한 머신러닝 알고리즘을 지원하고 데이터 처리를 위한 기능이 있는지, 업데이트가 지속해서 이뤄지는지 등을 고려해 선택하면 된다.

머신러닝 알고리즘을 직접 구현하는 것은 수학적 지식이 필요하고, 많은 시험과 최적화 등으로 검증해야 하는데, 새로운 머신러닝 알고리즘을 만들거나 기존의 머신러닝 알고리즘 자체를 개선할 목적이 아니라면 굳이 직접 개발할 필요는 없다.

머신러닝을 이용하는 데는 머신러닝 알고리즘에 대한 개념 이해와 사용법을 아는 것으로 충분하다. 머신러닝 프로그램 작성의 전체 프로세스를 생각해보면, 머신러닝 알고리즘 자체가 차지하는 비중과 시간은 10~20% 정도로 그다지 크지 않다. 데이터를 머신러닝에 사용할 수 있게 조작하는 전처리Pre-processing, 머신러닝의 결과를 분석하고 개선하기 위한 후처리Post-Processing 등의 과정이 오히려 더 중요하고 많은 시간이 필요하므로 사용하려는 라이브러리에서 이러한 기능들도 포함하고 있는지를 보는 것이 중요하다. 결국 전처리와 후처리에 의해 머신러닝 결과물의 품질이 결정되기 때문이다.

머신러닝 라이브러리를 선택할 때 GPU를 지원하는지 확인하는 것도 빠질 수 없는 점검사항이다. CPU와 GPU의 계산속도 차이는 작게는 수 배에서 크게는 수백, 수천 배 차이가 나기 때문에 머신러닝에 사용하려는 데이터의 크기가 크고 딥러닝 같은 계산

량이 많은 알고리즘을 사용한다면 GPU는 더는 선택이 아닌 필수가 될 것이다.

이 책에서 알고리즘 트레이딩에 추천하는 하드웨어 환경과 소프트웨어를 정리하면 다음과 같다.

Intel i5 이상, 메모리 4GB, HDD 256GB 이상

- **OS** 파이썬을 사용할 수 있는 환경(Windows, OS X, Linux)
- **Database** MySQL, 주가 관련 데이터 저장용이다.
- **프로그래밍 언어** 파이썬, 파이썬은 강력한 언어로 웹 개발, 클라우드, 금융에 이르기까지 폭넓게 사용되고 있다. 언어가 간결하고 배우기 쉬우며 다양한 라이브러리를 가지고 있다.
- **라이브러리**

  NumPy: 고차원의 수학적 기능을 제공하는 오픈소스 파이썬 라이브러리다. NumPy의 핵심 기능은 ndarray인데, 이는 n 차원의 배열 데이터 클래스로 다차원배열을 유연하면서도 빠른 속도로 사용할 수 있다. NumPy는 각종 수학과 과학 연산에서 많이 사용되는 벡터나 스칼라로 활용할 수 있고, 데이터베이스와 연동해 사용할 수도 있다.

  SCIPY: 과학 연산에 필요한 기능을 제공하는 라이브러리로, 최적화, 선형대수, 적분, FFT 등의 기능을 제공한다.

  Pandas: 금융 데이터 처리용 라이브러리로, `DataFrame` 클래스를 이용해 시계열 금융 데이터 처리에 필요한 각종 기능이 있다.

  Matplotlib: 그래프를 그리거나 데이터를 시각화하는 데 필요한 풍부한 기능을 가지고 있는 라이브러리다. 그래프를 저장하거나 확대 또는 축소를 위한 간단한 UI도 제공한다.

  scikit-learn(sklearn): 파이썬 머신러닝 라이브러리로, 딥러닝을 제외한 현존하는 거의 모든 머신러닝 알고리즘이 구현되어 있고 데이터 처리와 머신러닝 학습결과를 분석하는 기능들도 포함되어 있다. 알고리즘과 관계없이 사용법이 일관되어 짧은 학습시간에 직관적으로 이용할 수 있는 파이썬의 대표적인 머신러닝 라이브러리다.

Statsmodels: 파이썬 통계 라이브러리로 데이터 탐색, 통계적 모델 추정, 통계 테스트 등 다양한 통계 관련 기능을 지원한다.

Anaconda를 이용한 라이브러리 설치

Anaconda는 앞으로의 실습에 필요한 패키지들과 관련 프로그램을 모두 한 번에 설치해주는 프로그램으로, Windows, OS X, Linux를 지원한다. 파이썬에 익숙하지 않은 독자라면 파이썬 설치 프로그램인 pip 등을 이용해 필요한 라이브러리를 하나씩 설치하는 것이 어려울 수도 있다. 그래서 파이썬에 익숙하지 않은 독자라면 Anaconda를 이용해 매우 쉽게 필요한 실습 환경을 만들 수 있다.

Anaconda에 대한 자세한 설명은 홈페이지에서 확인할 수 있고 다운로드하려면 https://www.continuum.io/downloads 페이지를 방문하면 된다.

Anaconda 설치는 매우 간단해 다운로드한 파일을 실행하는 것이 전부다. Anaconda는 Python 2.7과 Python 3.5 버전 2가지를 제공하는데, Python 2.7 버전을 설치하기 바란다. 이 책에서 사용하는 라이브러리와 코드들은 모두 Python 2.7 버전에서 테스트했다.

Anaconda는 OS에 따라 설치되는 라이브러리에 차이가 있는데 Linux는 Anaconda에서 지원하는 라이브러리를 모두 설치할 수 있고, OS X는 상당 부분, Windows는 가장 적다. Anaconda에서 설치하는 모든 라이브러리 리스트를 보려면 http://docs.continuum.io/anaconda/pkg-docs 페이지를 참조하기 바란다.

차례