백설공주 거울과 인공지능 이야기

Original Japanese Language Edition
Kikai Gakushu Nyumon Boltzmann Kikai Gakushu kara Shinso Gakushu Made
by Masayuki Ohzeki

볼츠만 **머신러닝**에서 **딥러닝**까지

백설공주 거울과 인공지능 이야기

1쇄 발행 2018년 10월 30일

지은이 오제키 마사유키
옮긴이 심효섭
펴낸이 장성두
펴낸곳 주식회사 제이펍

출판신고 2009년 11월 10일 제406-2009-000087호
주소 경기도 파주시 회동길 159 3층 3-B호
전화 070-8201-9010 / **팩스** 02-6280-0405
홈페이지 www.jpub.kr / **원고투고** jeipub@gmail.com
독자문의 readers.jpub@gmail.com / **교재문의** jeipubmarketer@gmail.com

편집부 이종무, 황혜나, 최병찬, 이 슬, 이주원 / **소통·기획팀** 민지환 / **회계팀** 김유미
교정·교열 장성두 / **본문디자인** 성은경 / **표지디자인** 미디어픽스
용지 에스에이치페이퍼 / **인쇄** 한승문화사 / **제본** 광우제책사

ISBN 979-11-88621-44-6 (03000)
값 19,800원

볼츠만 머신러닝에서 딥러닝까지

백설공주 거울과 인공지능 이야기

오제키 마사유키 지음 / 심효섭 옮김

Jpub 제이펍

※ 드리는 말씀

- 이 책에 기재된 내용을 기반으로 한 운용 결과에 대해 저자, 역자, 소프트웨어 개발자 및 제공자, 제이펍 출판사는 일체의 책임을 지지 않으므로 양해 바랍니다.
- 이 책에 기재한 회사명 및 제품명은 각 회사의 등록 상표(또는 상표)이며, 본문 중에는 ™, ©, ® 등의 기호를 생략하고 있습니다.
- 이 책에서 사용하고 있는 실제 제품 버전은 독자의 학습 시점에 따라 책의 버전과 다를 수 있습니다.
- 책의 오류와 관련된 사항은 옮긴이나 출판사로 연락해 주시기 바랍니다.
  - 옮긴이: flourscent@gmail.com
  - 출판사: readers.jpub@gmail.com

우리 주변에서 머신러닝이라는 용어를 많이 듣게 됩니다. 이를 응용했다는 제품이나 서비스를 한두 번 정도는 사용해 본 경험이 대부분 있을 겁니다. 신기하지만 아직은 조금 어설픈 구석도 있는 제품이나 서비스를 사용해 보면서 이건 어떤 원리로 된 걸까 궁금했던 분들도 있으리라 생각합니다. 이 책은 이런 궁금증을 해소해 주는 책입니다. 모처럼 머신러닝이 무엇인지 궁금증이 생겼는데, 서점에는 수식으로 가득한 책 아니면 암호 같은 프로그램 코드만 나오기가 일쑤입니다. 일반인을 위한 머신러닝 안내서! 이것이 바로 이 책의 정체성이라고 할 수 있겠습니다.

백설 공주에 나오는 왕비와 마법의 거울을 화자로 삼아 머신러닝이 무엇인지부터 시작하여 주요 개념을 알기 쉽게 풀어 설명해 줍니다. 그러나 알기 쉬운 말로 수식도 없이 풀어 썼다고는 해도, 내용 없이 쉽게만 쓰인 책은 아닙니다. 대화체로 된 에피소드 형태로 상황을 진행하며 그 상황에 맞게 필요한 개념을 설명하고, 그 이론을 실제로 적용해 나가는 과정을 보여주기도 합니다. 그래서 재미있는 이야기를 읽으면서 그리 짧지 않은 머신러닝 연구 역사와 함께 요즘 유명한 딥러닝이 무엇인지까지 알 수 있게 해 줍니다.

아서 클라크가 말하기를 "충분히 발달한 기술은 마법과 구별할 수 없다."고 했습니다. 물론, 머신러닝은 아직 이 정도로 성숙한 기술은 아닙니다. 하지만 우리 생활에 점점 가까이 다가오는 이 기술의 원리를 이해하는 것도 즐거운 시간이 되리라 생각합니다. 감사합니다.

심효섭

아이가 성장하는 모습을 보면 '어떻게 세상과 사물을 이해하는지'에 대한 궁금증이 생깁니다. '여러 가지를 시도하고 실패하는 과정을 반복하면서 지식을 얻게 된다'라고 간단히 표현하기는 합니다. 그렇다면 '지식이란 무엇인가?'에 대한 답을 알게 된다면 사람과 같은 지능을 가진 컴퓨터도 가능하지 않을까요? 최근 회자되는 머신러닝이 바로 이를 실현하기 위한 기술입니다.

머신러닝은 다양한 시행착오를 통해 세상의 원리와 규칙을 학습하는 기술입니다. 그렇다면 이 학습이란 건 어떻게 이루어지는 것일까요? 머신러닝은 지금도 연구가 거듭되어 발전 중인 기술입니다만, 이 책에서는 그중에서도 볼츠만 머신을 이용한 머신러닝을 다룹니다. 볼츠만 머신을 이용한 머신러닝은 다양한 이미지를 보면서 실제 세계의 풍경을 기억하는 기술입니다. 사람은 이미지 일부가 초점이 맞지 않았거나 다른 사물에 가려져 보이지 않는 이미지에서도 이미지에 나타난 사물을 바로 인식할 수 있습니다. 이게 가능한 이유는 과거의 경험이나 현재 상황에 대한 정보를 추가로 고려하기 때문입니다. 이런 능력을 컴퓨터에도 탑재시킬 수 있게 된 것입니다. 이런 기술의 기반이 되는 것이 볼츠만 머신을 이용한 머신러닝입니다. 말하자면, 컴퓨터가 뇌의 기억을 담당하는 부분과 눈을 함께 갖출 수 있도록 한 것입니다.

딥러닝은 이 기술을 좀 더 발전시킨 것으로, 뇌의 판단 능력 부분을 추가하여 주변의 상황을 통해 판단하는 식별과 예측 능력을 컴퓨터에 부여하는 기술입니다. 뭔진 모르겠지만 참 대단한 기술이 나온 것 같습니다.

이러한 기술의 발전을 통해 동화 속 상상의 산물이라고 여겨졌던 '인공지능'이 현실화될

지도 모른다는 기대가 전 세계에서 높아지고 있습니다. 그리고 딥 마인드의 알파고 뉴스를 보고 컴퓨터의 새로운 능력에 전 세계가 깜짝 놀랐을 것입니다. 다양한 최신 머신러닝 기술을 사용하여 컴퓨터가 사람을 바둑에서 이긴 겁니다. 정말 대단한 기술입니다만, 그만큼 인공지능 기술에 대해 호기심을 가진 분들도 많을 겁니다. 그래서 책을 한번 살펴보고자 서점에 가 보면 정말로 다양한 책이 나와 있음을 알 수 있습니다. 거기다 요즘은 웹 검색을 해 보면 어지간한 지식은 바로 얻을 수 있는 세상이지요. 이들 정보를 접하고 좌절한 분들도 많으리라 생각합니다. 수식투성이에 어려워 보이니까요. 어떻게 해서 컴퓨터가 지능을 갖게 된다는 것인지, 그 원리를 대강이라도 알고 싶은 분들에게는 오르지 못할 산처럼 보였을 겁니다.

그래서 이 책에서는 '머신러닝은 무엇인가?'부터 시작하여 최근 연구의 돌파구를 만든 딥러닝, 그리고 딥러닝 초기 연구의 계기가 되었던 볼츠만 머신을 이용한 머신러닝을 '수식 없이', '이야기와 함께' 설명해 보고자 합니다. 관심 있는 사람이라면 누구든지 읽을 수 있는, 그야말로 세상에 없던 새로운 머신러닝 책을 만들어 보았습니다.

이 책은 인공지능을 배우고자 하는 개발자뿐만 아니라 인공지능에 관심이 많은 일반인에게도 적합한 책입니다. 심지어 은퇴 후에 여유로운 생활을 즐기시는 어르신께서도 재미있게 읽으실 수 있는 책입니다. 부모님과 함께 초/중학교 자유 토론 주제로 삼아도 좋을 만큼 쉽게 썼습니다. 이 이야기를 읽은 사람 중에는 머신러닝의 미래를 개척하게 될 사람도 분명 있을 겁니다. 전 국민 머신러닝 시대를 맞아 이 책이 작은 역할이나마 다할 수 있기를 바랍니다.

오제키 마사유키

강찬석(LG전자)

머신러닝 이론서라고 하면 보통은 딱딱하고 수식이 복잡하게 나와 있는 책이라는 편견이 있는데, 이 책은 이를 단박에 깨줍니다. 제목에서도 알 수 있듯이, 동화에 나올 법한 주제를 바탕으로 이론을 설명하고 있어서 처음 머신러닝을 접하는 입문자에게 많은 도움이 될 것으로 생각합니다. 머신러닝 책이라서 베타리딩을 지원할 당시에는 이론을 바탕으로 증명도 해야 하는 줄 알았는데, 막상 책을 읽어 보니 이해하기 쉬운 내용으로 서술되어 있어 인공지능 이론을 복습하는 데 많은 도움이 되었습니다. 앞으로도 관련 책들이 많이 나왔으면 좋겠습니다.

김민찬(서울시립대학교)

인공지능을 대화를 이용해 재미있게 풀어쓴 책입니다. 또한, 인공지능의 전체적인 개념을 한번 훑어볼 수 있는 책입니다. 인공지능이란 개념을 잘 모르고 흥미도 별로 없는 독자도 이 책은 재미있게 볼 것 같습니다. 책을 다 보고 나서는 한 편의 동화를 본 느낌이 들었습니다. 책의 번역 또한 전체적으로 자연스러워 읽기에 부담이 적었습니다. 개인적으로 캐릭터들이 귀엽고 등장인물들의 성격이 재밌어서 더 읽기 좋았던 것 같습니다. 이 책의 후속편도 기대해 봅니다.

김상열(삼성SDS)

머신러닝, 딥러닝 등 인공지능에 대한 기본 개념을 마법 거울과 왕비의 대화를 따라가며 흥미 있게 접근할 수 있었습니다. IT 관련 분야에 대한 지식이 없는 분도 쉽게 읽어 보실 수 있으리라 생각합니다. 번역도 어색하지 않고 잘 된 것 같습니다.

박재유(LG전자)

대학원 수업도 이수했고 다양한 서적도 읽어 왔지만, 기계학습이란 여전히 저에게 복잡하고 어려운 수식투성이로 느껴졌습니다. 도대체 이 함수는 무엇이고 왜 쓰는지를 이해하지 못한 채 무작정 데이터를 학습시켜서 결과를 보는 게 목적이었습니다. 그런데 이 책을 읽고 나니 드디어 전체 퍼즐이 맞춰지는 기분이 듭니다. 기계학습의 기원과 발전사를 이토록 쉽게 일반인 수준으로 풀어냈다는 사실에 감탄했습니다. 인공지능의 문턱을 한층 낮추는 데 크게 기여할 책으로 기대합니다.

장윤하(안랩)

경사 하강법, 비선형 변환, 커널법, 희소 모델링, 합성곱과 풀링 등 인공지능 분야를 처음 공부할 때 장애물(?)이 되는 개념들을 물 흐르듯이 너무도 쉽게 설명하고 있습니다. 베타리딩 내내 '아니, 이 개념을 어떻게 이런 스토리로 엮었단 말이야?'라며 감탄에 감탄을 거듭하며 읽어나갔습니다. 초보자뿐만 아니라 인공지능 기술을 오랫동안 사용한 분들도 개념을 재정립하는 데 적잖은 도움을 얻을 수 있을 것 같습니다. 내용 자체도 너무 좋고 책 번역 및 편집이 너무 잘 되어 있습니다.

정욱재(서울시립대학교)

이런저런 이야기와 비유로 어려운 개념을 쉽게 풀어냅니다. 책 속에 등장하는 왕비와 거울의 말괄량이 같은 모습은 읽는 내내 웃음을 선사합니다. 하지만 그렇다고 쉬운 개념과 삽화만 잔뜩 담긴 실속 없는 책은 아니니 머신러닝에 관심이 있다면 마음 편히 읽어 볼 것을 권합니다.

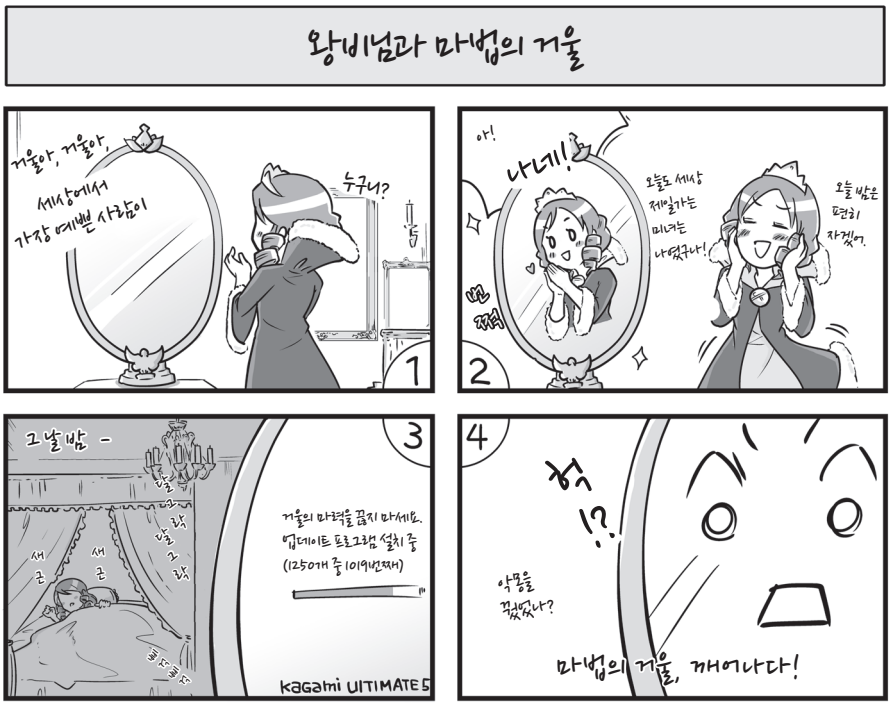
황도영(NHN)

동화 속 거울과 왕비의 인공지능 이야기는 제게 큰 충격을 주었습니다. 그 흔한 알고리즘 의사 코드 하나 없이 인공지능을 재미있는 이야기로 들려주다니 정말 놀라웠습니다. 공부를 위해 읽는 개발 서적이 아닌, 두 주인공의 인공지능 이야기가 궁금해 계속 읽게 되었습니다. 인공지능이 무엇인지 궁금하지만 알고리즘에 부담을 가지고 있다면, 이 책으로 시작해 볼 것을 추천합니다. 저 또한 인공지능에 대한 배경지식이 없는 상태임에도 너무나도 재밌게 술술 읽었습니다. 인공지능에 대한 전반적인 지식을 얻을 수 있었고, 무엇보다 공주와 거울의 이야기가 매력적이었습니다. 오랜만에 정말 재미있는 개발 서적을 읽었네요!

CHAPTER

1

아는 것이 없는 거울



1-1 마법의 거울과 왕비

"거울아, 거울아, 이 세상에서 누가 제일 예쁘니?"

백설공주 이야기에서 유명한 구절이다. 백설공주가 성장함에 따라 거울은 백설공주가 세상에서 제일가는 미녀가 되었다는 것을 알게 되었다. 거울은 스스로가 세상에서 가장 예뻐야 한다고 생각하는 왕비에게 무심코 이 사실을 전달했고, 이 말을 들은 왕비는 마음이 몹시 상했다는 이야기는 잘 알려져 있다.

이 거울을 실제로 만든다면 어떻게 해야 할까? 거울은 디스플레이로 대체하면 될 터이고, 거울 위에 카메라도 달아 보자. 카메라로 찍은 영상을 그대로 디스플레이로 내보내면 거울의 기본적인 기능을 갖추게 된다. 기분전환 삼아 온 세상의 다양한 풍경을 보여주는 것도 이제는 가능하다. 이 모두가 마법이 아닌 과학의 힘이다.

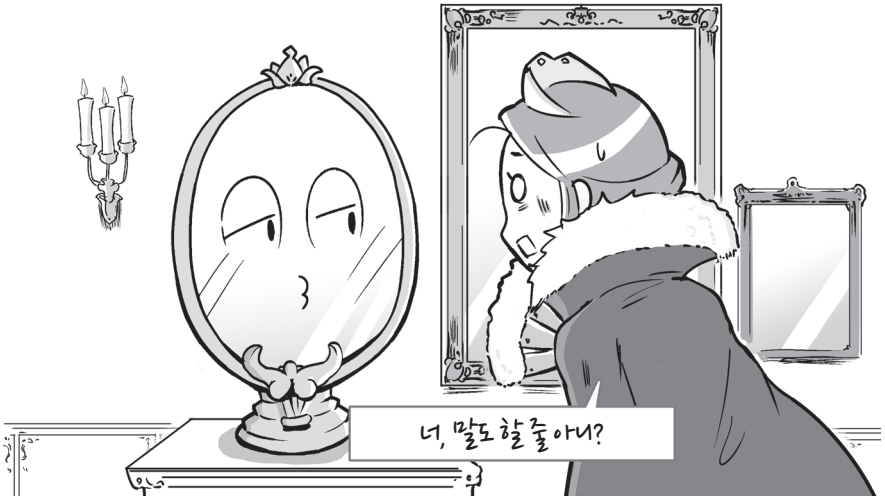
이 거울에 다시 컴퓨터를 추가해 보자. 이 컴퓨터가 머신러닝 알고리즘을 탑재한 컴퓨터라면 또 어떤 일을 할 수 있을까? 머신러닝, 현대의 마법이라 할 수 있는 이 기술이 이 책의 주인공이다. 그리고 이 기술을 탑재한 거울이 있다. 이 거울과 왕비가 만나면서 이 책의 이야기가 시작된다.


거울아, 거울아, 이 세상에서~


음… '예쁘다'는 게 무슨 뜻이죠?


너… 말도 할 줄 알아?


예. 사실은 말을 할 수 있게 됐어요. 얼마 전에 강제 업데이트가 되는 바람에 말이죠. 헤헤.



전에는 그냥 내 얼굴만 비추길래 내가 세상에서 가장 예쁘다는 줄 알았지.

아~ 예. 이것도 인연이니 잘 지내보도록 하죠. 그런데 '예쁘다'는 게 무슨 뜻인가요?

조금 시건방진 듯하지만, 뭐 됐어. '예쁘다'라는 게 무슨 뜻이냐면 말야. '아름답고, 귀엽고… 어쨌든 예쁘다'는 뜻이야.

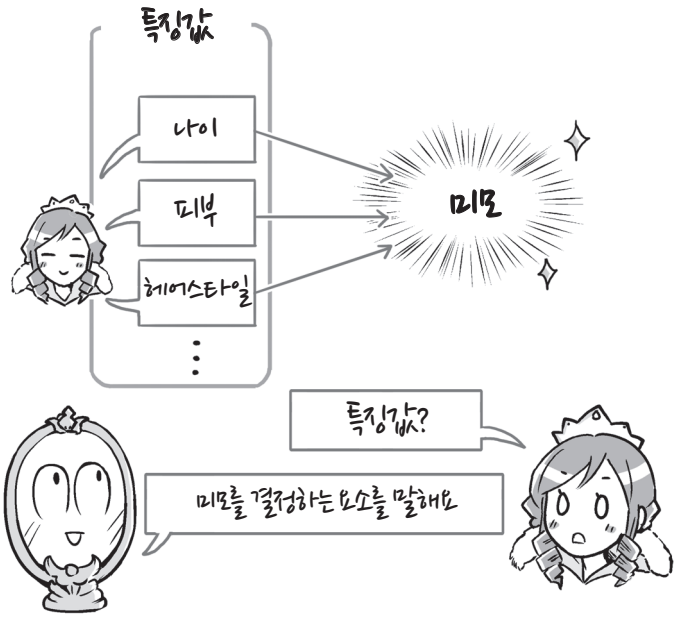
아니, 그렇게 추상적인 말은 이해할 수가 없어요. 좀 알아듣게 수치화해 주시면 좋겠어요. 왕비님이 말씀하시는 '예쁘다'는 도대체 어느 정도를 말하는 건지요?

그거야 당연히 엄청 예쁘지.

이거 참, 죄송합니다. 제가 신모델이라 예쁘다는 게 무슨 뜻인지 잘 몰라요.

예쁘다는 건 말이지. 피부가 곱고, 머리 모양도 귀엽고, 앉으나 서나 완벽하고 말이지. 무엇보다 드레스가 잘 어울려야…

아, 그래요? 그거예요! 예쁘다는 것을 수치화하기 위해서는 어떤 점을 보아야 하는지를 알아야 하는데, 그걸 특징값이라고 해요. 그걸 알려 주셔야 해요.



이목구비가 또렷하다든가, 젊은 편이라든가, 얼굴이 작은 편이라든가 하는 것도 있지.

'~한 편'이라고만 하면 제가 알 수가 없어요. 젊다는 건 또 몇 살을 말하는 건가요? 정확한 수치를 알려 주셔야죠.

여자의 나이를 묻다니, 실례야!

그게 아니고요. 거짓을 고하지 않는 게 제 신조이거든요. 그러니까 있는 그대로 확실히 대답해 주세요.

그래도 명색이 '있는 그대로 비추는 거울'이다, 이거니?

그럼요. 이목구비가 또렷하다는 것도 제대로 수치화해 주셔야 해요.

그걸 어떻게 숫자로 매길 수가 있니?

그럼 저도 알 수가 없어요. 예쁘다는 말이 뭔지도 잘 모르겠고요.

그런 걸 알아야 '얼마나 예쁜지'를 계산할 수 있다는 거야?

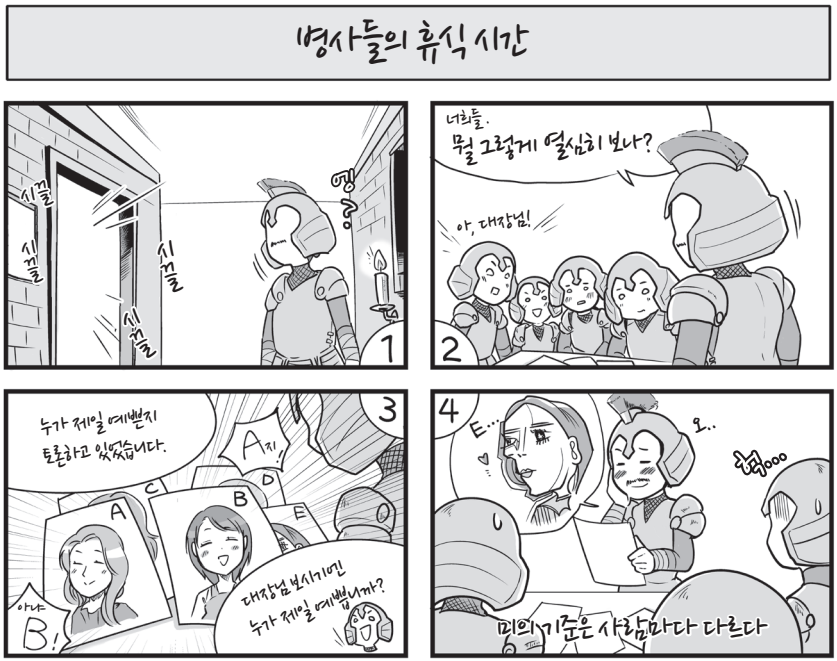
그럼요. 제가 새로 탑재한 머신러닝 시스템의 위력을 지금부터 보여드리도록 하죠.

뭐? 러닝머신?

CHAPTER

3

최적화 문제 풀어 보기



3-1 왕비의 전력질주

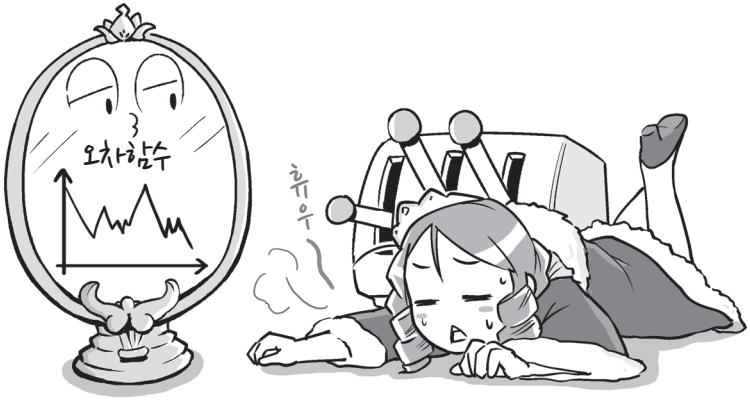
국가적인 프로젝트를 통해 수집한 데이터와 모순을 일으키지 않도록 미모 점수를 구하는 함수를 만들기 위해 왕비는 모형에 포함된 가중치를 바꿔보는 작업을 하고 있다.

데이터의 숫자 값과 모형에서 계산한 미모 점수의 차이를 가능한 한 줄이는 것이 왕비의 사명이었다. 가중치를 조금씩 바꿀 때마다 오차함수의 값도 조금씩 변한다. 이 오차를 가능한 한 작게 하는 최적의 가중치를 찾아야 한다.

이 문제는 **최적화 문제(optimization problem)**라는 유형의 문제다. 대다수의 머신러닝 문제에서 등장하는 기본적인 과제라고 할 수 있다. 그럼, 이 문제를 푸는 방법은 무엇일까?

하아…, 하아…

아직 오차가 많이 나네요.




아니, 이 레버를 만져봐도 오차가 나고 저 레버를 만져봐도 오차가 계속 나니 어떻게 해야 할지 모르겠네!


왕비님, 요령이 없으시네요.


시끄러워! 내가 왜 이렇게 힘들게 해야 하는 거냐고!


최적화 문제를 쉽게 푸는 요령이 있어요.


최적화 문제? 오차를 작게 하는 것 말이지?


맞아요. 오차가 적은 최적의 함수를 찾는 문제를 말해요. 오차를 줄이기 위해 지금은 오차함수의 최솟값을 찾아야 합니다. 이런 문제를 최적화 문제라고 해요.


그럼, 그 요령이란 걸 빨리 알려달라고~


레버를 이것저것 재밌게 만지작거리시길래 그냥 있었죠.


재미없거든!


먼저, 중요한 건 말이죠. 레버를 조금씩 움직이시라는 거예요.


조금씩?


레버를 살짝 움직여 보고 오차가 줄어드는지 확인합니다. 이 과정을 미분(derivative)이라고 해요.


학교에서 배웠던 미분이 이런 데 쓰는 거였구나!


맞아요. 학교에서 배운 것도 다 언젠가는 써먹을 날이 온다니까요.


이 레버를 살짝 밀어 보니까 오차가 줄어들었어.


오차가 줄어드는 걸 보니 그 레버가 맞네요. 그대로 조금 더 밀어 보세요.


맞다니?

우리의 목적은 오차를 줄이는 거였죠. 우리가 목적하는 대로 가고 있으니 방향이 맞는 거죠. 그 레버를 그대로 좀 더 밀어 보세요.

음… 얼마나 밀어야 해?

오차가 줄어드는 게 멈출 때까지요! 좀 더 밀어도 될 거 같은데요? 아, 거기까지요!

줄어드는 데까지는 줄이라는 거구나.

네, 맞습니다. 이렇게 오차가 줄어드는 방향으로 가중치를 조정해 가는 과정을 경사 하강법(gradient descent)이라고 해요.

이제 다 된 거야?

다른 레버도 있잖아요. 다른 레버도 움직여 보세요.

좀 감질나네. 이 레버를 밀어 보니깐 오차가 늘어났어.

그럼, 반대로 당겨 보면 어떻게 되나요?


오차가 줄어들었어. 그럼, 이번에는 이 레버를 당겨보면 되는 거지?


네, 그래요. 레버를 조금씩 움직일 때 오차가 어떻게 변하는지를 알아보는 방법이 미분이고요, 이 미분 결과를 이용해서 최소가 될 때까지 오차를 점점 줄여나가는 거죠.


이 과정을 계속 반복한다는 거지?


오차가 줄어드는 게 멈출 때까지 레버를 당겨보세요.


알았어! 오차가 더 줄어들었네! 괜찮은데?


그럼, 이번에는 이 레버를 조금 움직여 보세요.


미분한 후에 오차가 줄어드는 방향으로 레버를 움직인다!


맞아요. 그렇게 하면 돼요. 잘 하시는데요?


정리하자면, 다음 페이지의 그림과 같은 단순한 과정을 반복하면 되는 거군.


네, 바로 그거예요. 이런 내용으로 명령서를 작성하면 다른 사람한테 일을 시킬 수 있어요!

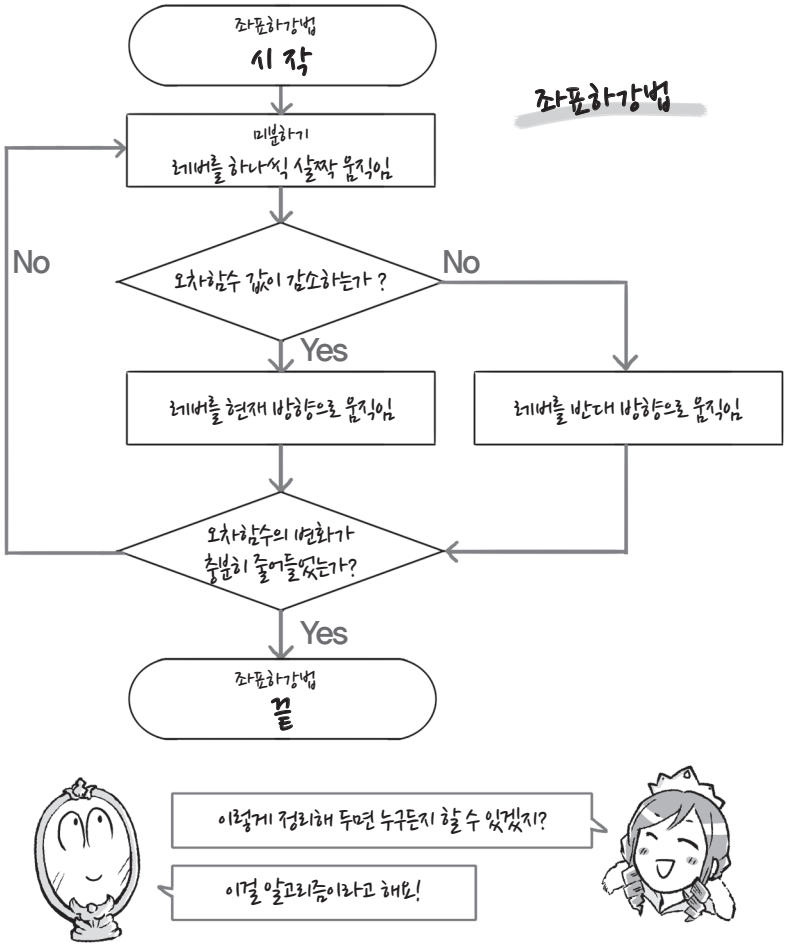

맞아, 맞아. 꼭 내가 할 필요는 없잖아.


이 명령서를 알고리즘(algorithms)이라고 합니다. 이 알고리즘은 좌표 하강법(coordinate descent)이라고 해요.


알고리즘?


알고리즘이란, 어떤 규칙을 정해서 그 규칙대로 따라 하면 문제를 풀 수 있도록 하는 과정을 나타낸 것을 말해요. 이렇게 작성한 명령서를 일을 맡길 사람에게 주면 되지요.


그렇다면 알고리즘을 직접 고안해서 그걸 일을 시킬 사람한테 주면 된다는 얘기네!



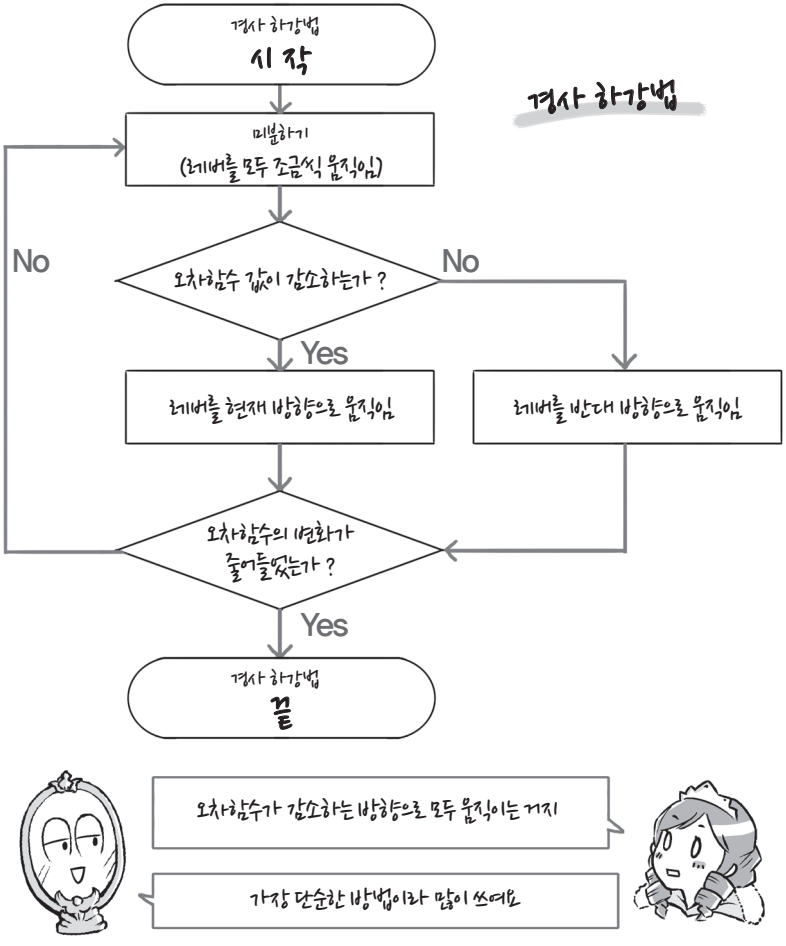
그렇지요. 효율적이고 기발한 알고리즘은 궁리를 거듭해서 오랜 시간 끝에 개발됩니다. 알고리즘을 한번 잘 만들어 두면 이 알고리즘을 다른 사람한테 전달하기만 하면 되거든요.

레버를 하나하나 만져보는 건 너무 번거로우니까 각각의 레버를 어느 방향으로 움직여야 하는지 알아본 다음, 모든 레버를 한 번에 움직이면 빠르지 않을까?

네, 그렇게 여러 개의 레버를 한 번에 움직여서 오차가 적어지도록 하는 방법을 경사 하강법(gradient descent, steepest descent)이라고 해요.

그러려면 레버를 움직여 줄 사람이 여러 명 필요하겠어.

레버를 모두 한 번에 움직일 수 있다면 효율적인 알고리즘이 되겠죠. 그렇게 여러 방법을 궁리해 보고 궁리한 방법을 시험해 보면 좋은 알고리즘을 만들 수 있어요. 지금 말씀하신 방법은 다음과 같은 알고리즘으로 정리할 수 있겠네요.

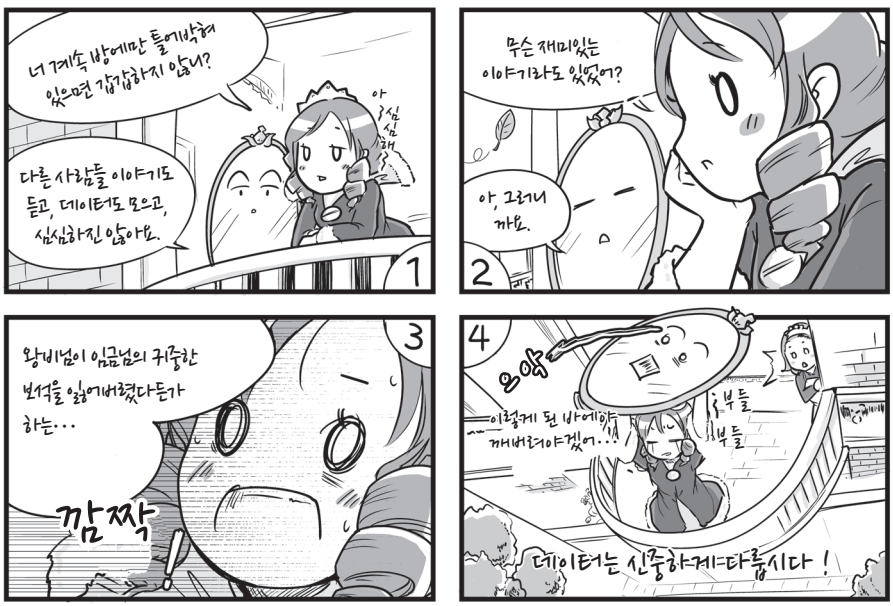


CHAPTER

4

딥러닝에 도전하기

왕비님과 너무 많은 것을 아는 거울



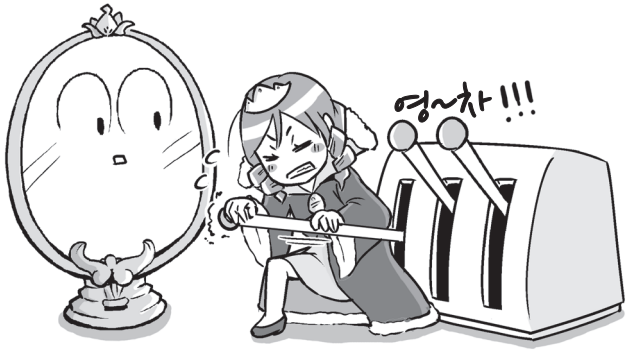
4-1 레버가 안 움직여!

다층 신경망은 단순히 목적하는 출력이 되도록 특징값을 조합하고 가중치를 곱해 더해 나가는 것이 아니라, 특징값을 조합하여 또 새로운 특징값을 만들고 다시 새로운 특징값에서 원하는 출력을 조합해 내는 구조로 되어 있다. 언뜻 생각하면 어떤 함수이든지 근사(approximate)해 낼 수 있는 만능 기법이 탄생했다고 생각하겠지만, 아쉽게도 이를 활용하는 데는 여러 문제가 있었다. 그러나 이들 문제는 이제 대부분 해결되었다고 한다. 대체 어떤 방법으로 문제를 해결한 것일까?

그럼, 이제 다층 신경망을 이용하여 얼마나 예쁜지 알아보는 함수를 만들어 볼까요?

레버를 조작해서 최적화 문제를 풀기만 하면 되는 거지? 오차가 줄어들게끔 말이야.

네, 맞아요. 어쨌든 오차함수의 값이 작아지게만 하면 되는 건 마찬가지죠. 똑같은 알고리즘이에요.




간단하겠네! 나도 도와야겠어. 에잇!


그만두시는 게 좋을걸요.


이거 왜 이래? 레버가 전혀 안 움직이잖아. 어떻게 된 거야?


뭐가 어떻게 된 건 아니고, 원래 그렇습니다.


무슨 말이야? 이래서는 가중치를 바꿀 수가 없잖아!


네. 기울기 소실(vanishing gradient) 문제 때문에 그래요.


기울기 소실? 기울기라니?


지금까지는 레버를 살짝 움직여서 오차가 늘어나는지 줄어드는지를 보았었죠? 이게 미분이에요. 이때 오차가 얼마나 변화하는지 나타낸 값을 기울기라고 해요.


기울기가 있다는 건 오차가 잘 줄어든다는 얘기겠지?


맞아요. 그 기울기가 사라지는 문제가 기울기 소실 문제예요. 그만큼 오차를 줄이기도 어렵게 되지요. 그렇기 때문에 레버가 움직이지 않았던 거예요.


다른 사람들에게 부탁해야겠어.


그 사람들도 마찬가지일걸요!


대체 왜 이렇게 되는 거야, 정말.


레버를 움직이면 출력값이 바뀌겠지요? 이 출력값이 데이터와 맞도록 하고 싶은데 다층 신경망에서는 이 출력까지 가는 길이 엄청 멀어지거든요.


확실히 출력까지 가려면 조합도 여러 번 거쳐야 하고 비선형 변환이랬나? 그것도 여러 번 통과해야 하겠네.

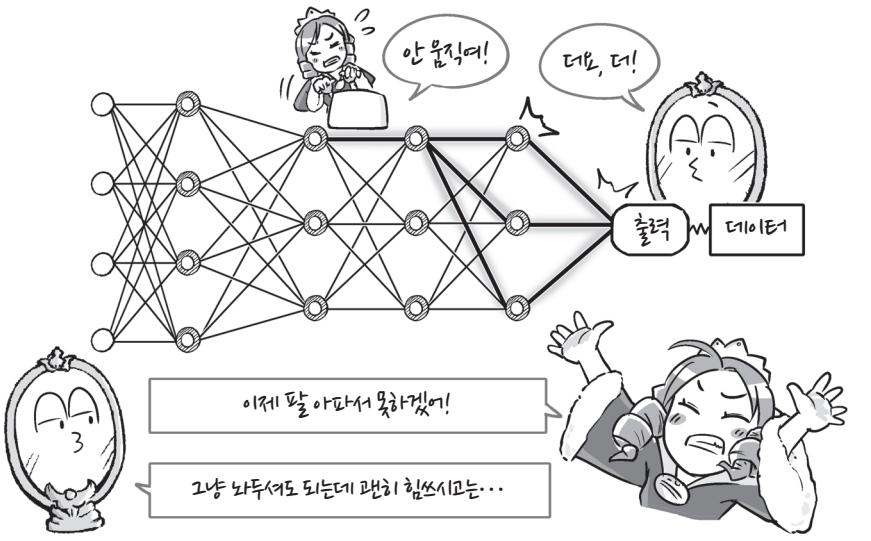

신경망 구조를 보면 아시다시피 끈이 얽히고설킨 것 같은 구조이거든요.

레버를 움직이는 게 그 끈과 무슨 상관이 있는 거야?

비유하자면, 레버를 움직이면 끈이 팽팽해지거나 느슨해지거나 하면서 출력값이 바뀌게 되는 거예요. 출력값이 데이터와 비슷하다면 오차함수 값이 작아지겠죠. 이렇게 끈에 달린 멀리 있는 물체를 휘둘러서 원하는 대로 옮기는 일을 상상해 보세요.

어느 끈을 당겨야 원하는 대로 움직일 수 있는지 모르니까 어려운 건데, 그래도 조금씩 움직여 보면 원하는 방향을 찾을 수 있지 않을까?

네. 원칙적으로는 그렇게 하면 됩니다. 그런데 출력과 연결된 끈도 여러 개이잖아요? 여러 가닥의 끈이 모두 영향을 미치고 있기 때문에 레버를 당겨 끈 한 가닥을 이렇게 당겨 보면…



아, 안 움직인다!

이런 상태라는 거죠. 모형이 복잡한 만큼 이 끈도 복잡하게 얽혀 있어요. 그래서 끈 한 가닥을 당겨도 엄청 많은 끈이 끌려오게 되는 거예요. 이렇게 중간에 복잡하게 엉킨 끈

을 조정해서 오차함수를 최적화하는 방법을 오차 역전파법(back-propagation)이라고 합니다. 이게 어려운 부분이에요.

이거 참 곤란하네… 그런데 그래도 무슨 방법이 있다고 하지 않았어?

레버가 안 움직인다고 하셨는데, 전부 다 그렇던가요?

그래? 아까 저건 안 움직였고… 이건 괜찮네.

그렇죠? 출력과 가까울수록 움직이기 쉬워요. 기울기가 있으니까요. 하지만 출력에서 멀어질수록 레버가 무거워지죠.

왜 그렇게 되는 거야?

잘 생각해 보세요. 출력에 가까울수록 레버가 가볍다는 건 레버를 움직였을 때 오차가 줄어드는 정도가 크기 때문이에요. 다시 말해, 레버의 움직임이 출력의 변화에 직접 영향을 미치기 때문이죠. 그런데 출력에서 먼 레버는 레버를 움직여도 그 영향이 출력까지 미치는 데 많은 과정을 거쳐야 하죠?

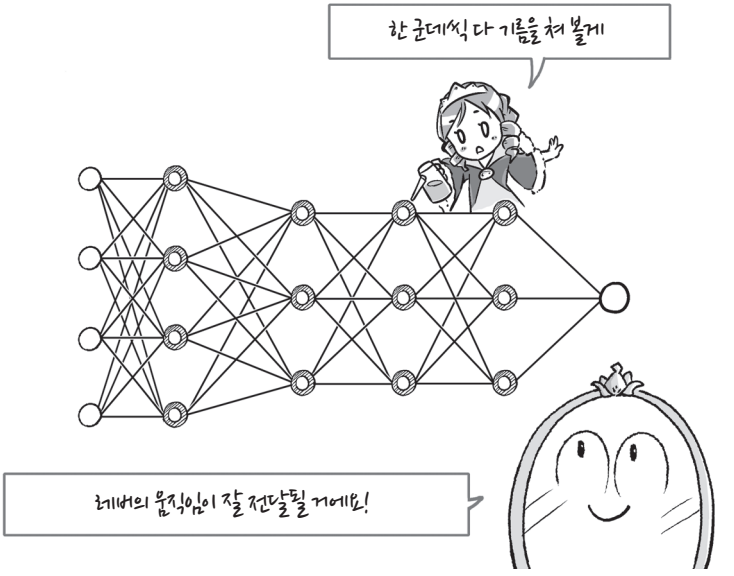
출력까지 오는 데 거치는 과정이 많으니까 잘 안 움직인다는 얘기이구나. 그만큼 끈이 복잡하게 엉켜 있으니까.

비유하자면 그렇다는 얘기예요. 끈이 복잡하게 엉켜 있는 건 그만큼 모형을 복잡하게 만들었기 때문이에요. 모형을 복잡하게 만들어서라도 출력을 데이터와 비슷하게 해야 하니까요. 그건 다른 방법이 없어요.

레버를 가능하면 쉽게 움직일 수 있도록 레버 움직임의 영향이 출력에 직접 미치게 하는 게 좋아요. 우선, 끈이 서로 엉켜 있는 곳마다 기름을 쳐 볼까요?

레버를 살짝만 움직여도 오차함수 값이 바뀌도록 하려는 거지? 지금으로써는 레버가 무거워서 살짝 움직이기도 쉽지가 않아.

이걸 더 쉽게 움직이도록 하려면 활성화 함수에 주목해야 해요. 여기서는 시그모이드 함수를 사용하지 않는 편이 좋죠.



그 비선형 변환에 쓰는 함수 말이지?

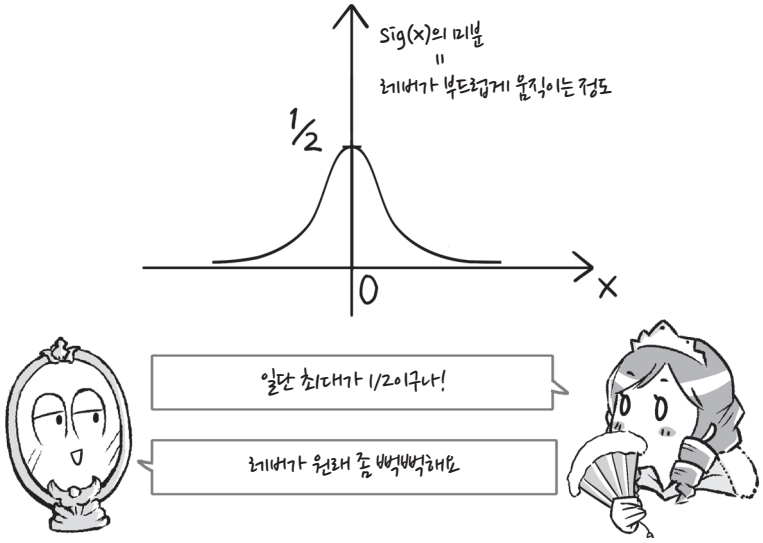
네. 안타깝지만 시그모이드 함수를 사용하는 한은 '레버를 움직이기 어려워요.' 이것도 역사적으로 보면 이유가 있어서 사용하게 된 거라 바로 들어내지는 못하지만요.

애초 시그모이드 함수를 활성화 함수로 사용하던 시절에는 레버를 움직여도 출력이 그리 크게 변하지 않았고, 오차함수 값도 마찬가지였죠. 레버의 영향이 미치기 어려웠거든요.

그건 어떻게 알 수 있어?

시그모이드 함수를 미분해 보도록 하죠. 함수 입력을 살짝 변화시켰을 때 함숫값이 변하는 정도를 볼 수 있어요.

거의 대부분 값이 작네.



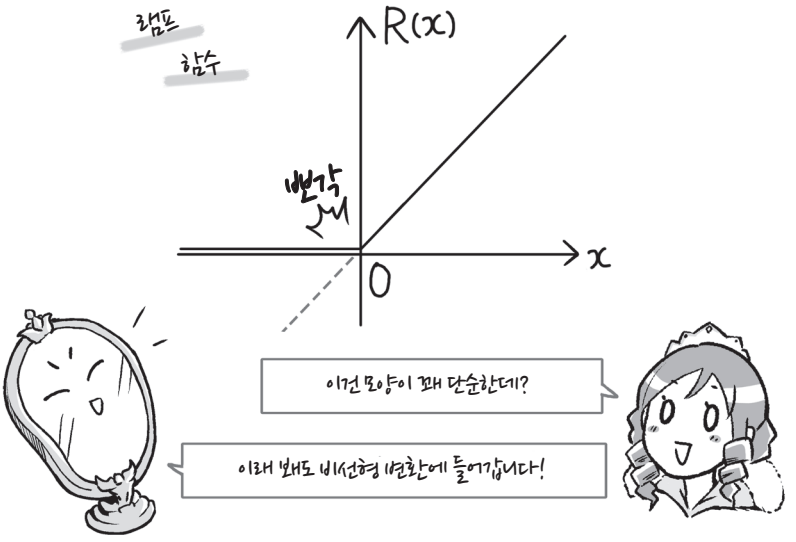
그렇지요? 미분값이 작으니까 반응도 그리 크지 않은 거예요. 그래서 레버도 무거워지는 거구요.

그럼, 레버가 쉽게 움직이도록 활성화 함수를 다른 함수로 바꾸면 된다는 얘기네?

맞아요. 꽤 오래전부터 신경망을 이용한 머신러닝, 특히 딥러닝의 시초를 연구했던 얀 르쿤(Yann LeCun)이 쌍곡선 함수 tanh(x)을 제안하였죠. 어떤 활성화 함수가 좋은지에 대한 논의가 이어졌고, 현재는 램프 함수(ramp function)가 주로 추천되고 있어요.

램프 함수? 그 함수를 쓰면 레버에 기름칠을 하듯이 부드러워진다는 거지? 그게 어떤 함수인데?

엄청 간단해요. 다음 그림과 같은 모양이에요.



이건 그냥 직선이잖아!

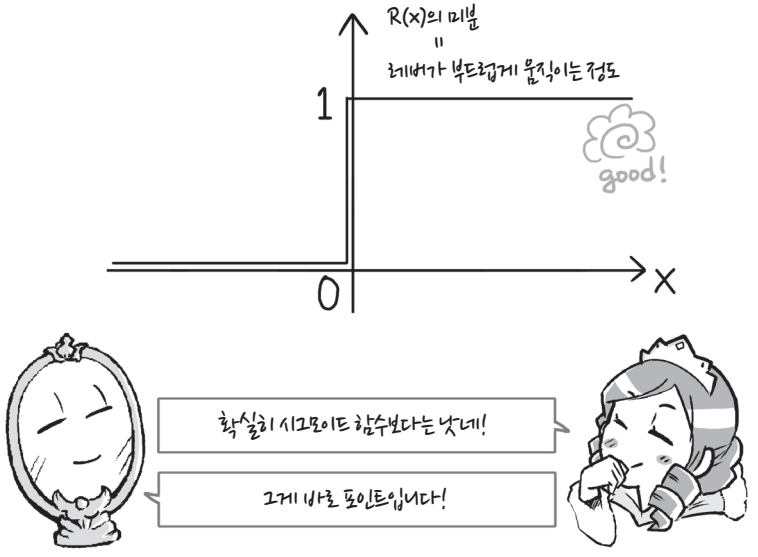
직선이라도 원점 부근에서 꺾여 있는 게 보이시죠? 이 꺾인 부분 때문에 비선형 변환이 가능해요. 거기다 미분도 전달되기가 쉽다는 특징이 있죠.

이 함수를 사용하면 레버가 잘 움직이겠네?

네. 지금은 다들 이 함수를 비선형 변환에 사용합니다.

어, 정말이네. 레버가 많이 부드러워졌어.

램프 함수를 미분해 보면 미분값이 시그모이드보다 크고, 미분값이 큰 구간도 시그모이드보다 넓은 걸 알 수 있어요. 이 미분이 레버가 얼마나 부드럽게 움직이느냐와 관계가 깊으니까 중요한 성질이죠.




램프 함수가 윤활유랑 같은 역할을 하는 거구나.


이렇게 첫 번째 문제, 기울기 소실 문제가 해결되었습니다.


응? 문제가 또 있는 거야?


과적합이라는 문제가 아직 남아 있어요.

Column

딥러닝의 융성

딥러닝에서는 다층 신경망을 기본 형태로 삼는 복잡한 신경망을 사용한다. 복잡한 신경망을 사용한다는 발상 자체는 그 역사가 깊지만 최근에서야 널리 쓰이게 되었다. 이렇게 딥러닝이 널리 쓰일 수 있었던 계기는 무엇일까? 그 계기라면 역시 데이터 집합의 규모가 매우 커진 것과 그만큼 대량의 데이터를 수집하기 쉬운 환경이 되었기 때문이다.

다층 신경망에는 많은 수의 파라미터, 다시 말해 어느 특징값이 중요한지를 나타내는 가중치가 있다. 대규모 데이터 집합이 나타내는 다양한 현상에 모형을 부합시키려면 그만큼 많은 파라미터가 필요하기 때문이다. 이를 최적화하는 데도 다양한 조건에서 시험을 거친 변화가 풍부한 데이터가 필요하다. 이를 볼 때 다층 신경망을 학습시키는 데는 대규모 데이터가 필수 불가결하다.

예전에는 대규모 데이터를 수집하기가 쉽지 않았던 점을 감안하면, 다층 신경망을 학습시키기도 어려웠을 것이다. 설사 데이터를 수집할 수 있었더라도 복잡한 신경망이 갖는 많은 파라미터를 최적화시키는 일 자체가 지극히 어려운 일이다. 왕비가 아무리 많은 도우미를 데려오더라도 야단법석이 벌어졌을 것이다.

이 때문에 알고리즘 및 신경망의 성질부터 활성화 함수의 종류에 이르기까지 다양한 방법에 대해 세세한 연구가 이루어지고 있다. 그러나 예전에는 좋은 알고리즘을 구현했어도 대규모 데이터를 다룰 컴퓨터의 성능이 이를 뒷받침하지 못했다. 하지만 지금은 휴대가 가능한 노트북 컴퓨터에서도 가볍게 이런 계산을 수행할 수 있게 되었다.

다층 신경망을 비롯해 복잡한 신경망을 사용한 딥러닝이 지금에서야 발전할 수 있었던 것은 대규모 데이터 생산 및 수집이 쉬워진 컴퓨팅 환경과 컴퓨터 성능 향상 및 알고리즘의 발전이 동시에 잘 실현되었기 때문이다.