Arcydzieło!
— **Geoffrey Hinton**, noblista, ojciec chrzestny sztucznej inteligencji

Dlaczego maszyny się uczą?

O pięknie matematyki i działaniu współczesnej sztucznej inteligencji

Anil Ananthaswamy

Helion

Tytuł oryginału: Why Machines Learn: The Elegant Math Behind Modern AI

Tłumaczenie: Tomasz Walczak

ISBN: 978-83-289-3002-5

This edition published by arrangement with Dutton, an imprint of Penguin Publishing Group, a division of Penguin Random House LLC.

Portions of chapter 12 and the epilogue appeared in *Quanta Magazine*. The illustration in chapter 6 on PCA done on EEG data adapted with permission from John Abel. The illustrations in chapter 12 on the bias-variance and double descent curves adapted with permission from Mikhail Belkin. Illustrations about properties of penguins in chapter 4 created courtesy of data made freely available by Kristen Gorman, Allison Horst, and Alison Hill. The illustrations of biological neuron (p. 19), paddy fields (p. 56), and the map of Manhattan (p. 112) by Roshan Shakeel.

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/dlamas
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Printed in Poland.

Spis treści

ROZDZIAŁ 1

W desperackim poszukiwaniu wzorców

Gdy austriacki naukowiec Konrad Lorenz był dzieckiem, zafascynowały go opowieści z książki *Cudowna podróż*, historii przygód chłopca wędrującego z dzikimi gęsiami napisanej przez szwedzką noblistkę w dziedzinie literatury Selmę Lagerlöf[1]. W efekcie młody Lorenz marzył o tym, by stać się dziką gęsią. Ponieważ nie mógł spełnić tej fantazji, zadowolił się opieką nad jednodniowym kaczątkiem, które dostał od sąsiada. Ku radości chłopca, kaczątko zaczęło za nim chodzić, co było wynikiem procesu „wdrukowania". Wdrukowanie dotyczy zdolności wielu zwierząt, w tym małych kaczek i gęsi, do tworzenia więzi z pierwszym poruszającym się obiektem, który zobaczą po wykluciu. Lorenz został później etologiem i pionierem badań nad zachowaniem zwierząt, przede wszystkim nad wdrukowaniem. Udało mu się sprawić, że kaczęta w wyniku wdrukowania uznawały go za matkę[2]. Podążały za nim, gdy chodził, biegał, pływał, a nawet gdy odpływał kajakiem. W 1973 roku wspólnie z innymi etologami, Karlem von Frischem i Nikolaasem Tinbergenem, otrzymał Nagrodę Nobla w dziedzinie fizjologii lub medycyny. Ci trzej naukowcy zostali uhonorowani „za odkrycia związane ze strukturą i wywoływaniem indywidualnych i społecznych *wzorców* zachowań"[3].

Wzorce. Podczas gdy etolodzy dostrzegali je w zachowaniu zwierząt, te ostatnie również wykrywały własne wzorce. Świeżo wyklute kaczątka muszą mieć zdolność rozróżniania cech obiektów poruszających się wokół nich. Okazuje się, że wdrukowanie u kaczątek może dotyczyć nie tylko pierwszej żywej istoty, którą zobaczą w ruchu, ale także przedmiotów nieożywionych. Na przykład u młodych kaczki krzyżówki wdrukowanie może dotyczyć pary poruszających się obiektów, które mają podobne kształty lub kolory. Wdrukowanie wiąże się w tej sytuacji z relacjami reprezentowanymi przez te obiekty[4]. Jeśli więc po wykluciu kaczątka zobaczą dwa poruszające się czerwone przedmioty, będą później podążać za dwoma obiektami takiego samego koloru (nawet jeśli te obiekty będą niebieskie, a nie czerwone), ale już nie za dwoma obiektami różnych kolorów[5]. W tym przykładzie wdrukowanie u kaczątek dotyczy idei *podobieństwa*. Młode wykazują również zdolność rozpoznawania *różnic*. Jeśli jako pierwsze poruszające się obiekty kaczątka zobaczą na przykład sześcian i prostopadłościan, to później będą podążać za dwoma innymi obiektami o odmiennych kształtach (na przykład za piramidą i stożkiem), ponieważ rozpoznają, że te obiekty różnią się od siebie kształtem. Będą natomiast ignorować dwa obiekty o takim samym kształcie.

Zastanów się nad tym przez chwilę. Nowo wyklute kaczątka po bardzo krótkim kontakcie z bodźcami zmysłowymi potrafią wykrywać wzorce w tym, co widzą, oraz tworzyć abstrakcyjne pojęcia podobieństwa i różnic, a następnie rozpoznawać te abstrakcje w późniejszych bodźcach i odpowiednio na nie reagować. Badacze sztucznej inteligencji wiele by dali, żeby się dowiedzieć, jak kaczątkom udaje się to osiągnąć.

Choć dzisiejsza sztuczna inteligencja jest jeszcze daleka od wykonywania takich zadań z łatwością i skutecznością obserwowaną u kaczątek, ma z nimi coś wspólnego — zdolność do wykrywania i uczenia się wzorców w danych. Kiedy Frank Rosenblatt pod koniec lat 50. XX wieku wynalazł perceptron, jednym z powodów, dla których wywołało to takie poruszenie, było to, że uznano go za pierwszy ważny algorytm „inspirowany mózgiem" i potrafiący uczyć się wzorców w danych na podstawie samej ich analizy. Co najważniejsze, badacze udowodnili, że przy pewnych założeniach dotyczących danych perceptron Rosenblatta zawsze znajdzie ukryty w nich wzorzec w skończonym czasie. Innymi słowy, perceptron zawsze zbiega do rozwiązania. Takie pewniki w informatyce są na wagę złota. Nic dziwnego, że algorytm uczenia perceptronu wzbudził taką sensację.

Ale co właściwie oznaczają te terminy? Czym są „wzorce" w danych? Co kryje się pod stwierdzeniem „uczenie się tych wzorców"? Zacznij od przyjrzenia się poniższej tabeli:

x1	x2	y
4	2	8
1	2	5
0	5	10
2	1	4

Każdy wiersz w tej tabeli zawiera trzy wartości dla zmiennych $x1$, $x2$ i y. W tych danych ukryty jest prosty wzorzec: w każdym wierszu wartość y jest powiązana z odpowiadającymi jej wartościami $x1$ i $x2$. Spróbuj go dostrzec, zanim przejdziesz do dalszej lektury.

W tym zadaniu za pomocą ołówka, kartki i odrobiny wysiłku można dojść do wniosku, że y równa się $x1$ plus dwa razy $x2$.

$$y = x1 + 2x2$$

Mała uwaga dotycząca notacji: rezygnuję tu ze znaku mnożenia (·) pomiędzy dwiema zmiennymi lub między stałą a zmienną. Stosuję na przykład zapis:

$2x2$ zamiast $2 \cdot x2$ i $x1x2$ zamiast $x1 \cdot x2$.

Idealny byłby zapis $2x2$ jako $2x_2$, a $x1x2$ jako x_1x_2, ze zmiennymi z indeksami dolnymi. Jednak rezygnuję z indeksów dolnych, chyba że ich użycie okaże się absolutnie konieczne. Może to razić purystów, ale taka metoda pozwala zachować przejrzystość tekstu i ułatwia jego czytanie. Gdy napotkasz indeksy dolne, czytaj x_i jako „x z indeksem dolnym i". Pamiętaj o następującej kwestii: jeśli po symbolu takim jak „x" znajduje się cyfra, na przykład „2" (czyli $x2$), taka sekwencja oznacza jeden element. Z kolei jeżeli liczba (na przykład 9) lub inny symbol (na przykład $w1$) znajduje się przed symbolem (na przykład x lub $x2$), oznacza to, że mnożenie liczby i symbolu lub dwóch symboli. Tak więc:

$$2x2 = 2 \cdot x2$$

$$x1x2 = x1 \cdot x2$$

$$w2x1 = w2 \cdot x1$$

Równanie $y = x1+2x2$ można zapisać w ogólnej postaci w następujący sposób:

$$y = w1x1+w2x2, \text{ gdzie } w1 = 1 \text{ i } w2 = 2$$

Warto zauważyć, że jest to jedna z wielu możliwych zależności między y a $x1$ i $x2$. Mogą istnieć również inne. I rzeczywiście w tym przykładzie tak się dzieje, ale w tym kontekście nie trzeba znajdować ich wszystkich. Wykrywanie wzorców nie jest tak proste, jak sugeruje ten przykład, ale daje on punkt wyjścia do dalszych rozważań.

Zidentyfikowałem tak zwaną zależność liniową między y z jednej strony, a $x1$ i $x2$ z drugiej. Określenie „liniowa" oznacza, że y zależy tylko od $x1$ i $x2$, a nie od $x1$ lub $x2$ podniesionych do jakiejś potęgi lub od iloczynu $x1$ i $x2$. Warto też zaznaczyć, że używam tu zamiennie słów „równanie" i „zależność".

Zależność między y, $x1$ i $x2$ jest określona przez stałe $w1$ i $w2$. Te stałe nazywane są współczynnikami lub wagami równania liniowego łączącego y z $x1$ i $x2$. W tym prostym przykładzie, przy założeniu występowania takiej zależności liniowej, wartości $w1$ i $w2$ zostały ustalone na podstawie analizy danych. Jednak często relacja między y a ($x1$, $x2$, …) nie jest tak oczywista, zwłaszcza gdy po prawej stronie równania pojawia się więcej zmiennych. W takich sytuacjach określenie odpowiednich wag staje się bardziej skomplikowane.

Rozważ następujący przykład:

$$y = w1x1+w2x2+w3x3+ \ldots +w9x9$$

W ogólnym ujęciu dla zbioru n wag można za pomocą formalnego zapisu matematycznego przedstawić to tak:

$$y = w1x1 + w2x2 + w3x3 + \cdots + wnxn = \sum_{i=1}^{n} wixi$$

Wyrażenie po prawej stronie, przedstawione za pomocą znaku sigma, jest skróconym zapisem sumy wszystkich iloczynów $wixi$, gdzie i przyjmuje wartości od 1 do n.

Dla dziewięciu wejść trudno ustalić wartości wag od $w1$ do $w9$ na podstawie samej wzrokowej analizy danych i obliczeń przeprowadzonych w pamięci. Tu właśnie wkracza uczenie maszynowe. Jeśli istnieje sposób na algorytmiczne ustalenie wag, można powiedzieć, że algorytm „uczy się" tych wag. Ale jaki jest cel tego procesu?

No cóż, jeśli zostały ustalone wagi — powiedzmy $w1$ i $w2$ w prostym, przykładowym modelu — to po napotkaniu jakichś wartości $x1$ i $x2$, które nie występowały w pierwotnym zbiorze danych, można obliczyć wartość y. Załóżmy, że $x1 = 5$, a $x2 = 2$. Gdy podstawisz te wartości do równania $y = x1+2x2$, otrzymasz $y = 9$.

Jakie ma to odniesienie do rzeczywistego świata? Oto bardzo prosty, praktyczny i, zdaniem niektórych, nudny problem. Załóżmy, że $x1$ oznacza liczbę sypialni w domu, $x2$ całkowitą powierzchnię, a y cenę domu. Przyjmijmy, że istnieje liniowa zależność między $(x1, x2)$ a y. Wtedy po ustaleniu wag równania liniowego na podstawie dostępnych danych o domach i ich cenach uzyskasz bardzo prosty model do przewidywania ceny domu na podstawie liczby sypialni i powierzchni.

Ten przykład to początek uczenia maszynowego, a w zasadzie maleńki pierwszy krok. Opisany proces jest uproszczoną formą tak zwanego uczenia nadzorowanego. Dostępne są próbki danych, w których ukryta jest korelacja między zbiorem danych wejściowych a zbiorem danych wyjściowych. Takie dane nazywamy oznakowanymi lub etykietowanymi. Określa się je również mianem danych treningowych. Do każdego zestawu danych wejściowych $(x1, x2, \ldots, xn)$ przypisana jest etykieta y. Tak więc w przedstawionej wcześniejszej tabeli para liczb (4, 2) jest oznaczona etykietą $y = 8$, para (1, 2) etykietą 5 i tak dalej. Udało się odkryć korelację między wartościami. Po jej ustaleniu można wykorzystać ją do przewidywania wyników dla nowych danych wejściowych, które nie były częścią zbioru treningowego.

Ponadto omówiłem szczególny sposób rozwiązywania problemów zwany regresją, gdzie na podstawie zmiennych niezależnych $(x1, x2)$ tworzony jest model (lub równanie) do przewidywania wartości zmiennej zależnej (y). Istnieje też wiele innych odmian modeli, które można zbudować. Omawiam je dalej.

W przykładzie korelacja (czyli wzorzec) była tak prosta, że wystarczyła niewielka ilość oznaczonych danych. Jednak współczesne uczenie maszynowe wymaga o wiele większych ilości danych, a ich dostępność była jednym z czynników napędzających rewolucję sztucznej inteligencji. Kaczątka prawdopodobnie korzystają z bardziej zaawansowanej formy uczenia się. Żadna kacza matka nie siedzi i nie dodaje etykiet do danych dla piskląt, a mimo to maluchy się uczą. Jak to robią? Uwaga, spoiler: tego nie wiemy, ale być może zrozumienie, *dlaczego* uczą się maszyny, pozwoli kiedyś w pełni pojąć, jak uczą się kaczątka, a nawet ludzie.

Może się to wydawać nieprawdopodobne, ale ten pierwszy krok w postaci niezwykle prostego przykładu uczenia nadzorowanego wprowadza nas na ścieżkę do zrozumienia nowoczesnych głębokich sieci neuronowych. Oczywiście będziesz poruszać się po niej małymi kroczkami, a po drodze natrafisz na niewielkie, a czasem trochę bardziej rozbudowane dawki wektorów, macierzy, algebry liniowej, rachunku różniczkowego, rachunku prawdopodobieństwa, statystyki i teorii optymalizacji.

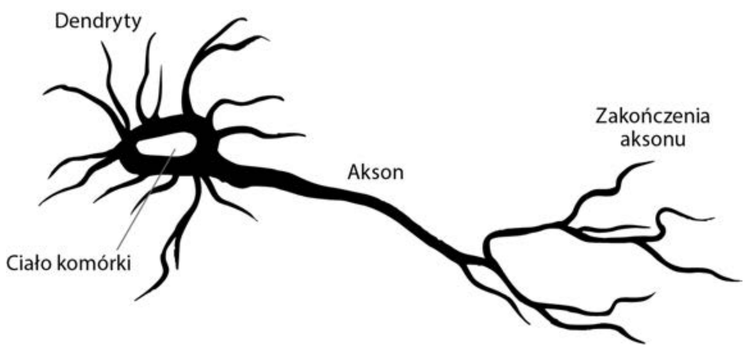
Perceptron Rosenblatta, o którym wspomniałem we wstępie, był zadziwiającym (jak na swoje czasy) przykładem uczącego się algorytmu. A ponieważ inspiracją do jego powstania było to, jak neurobiolodzy wyobrażali sobie działanie ludzkich neuronów, otaczała go aura tajemniczości i był symbolem nadziei na to, że pewnego dnia perceptrony rzeczywiście spełnią marzenia o sztucznej inteligencji.

PIERWSZY SZTUCZNY NEURON

Korzenie perceptronu sięgają artykułu napisanego w 1943 roku przez niezwykłą parę: filozoficznie nastawionego neuronaukowca w średnim wieku i bezdomnego nastolatka. Warren McCulloch był amerykańskim neurofizjologiem z wykształceniem w dziedzinach filozofii, psychologii i medycyny[6]. W latach 30. XX wieku zajmował się neuroanatomią. Tworzył wtedy mapy połączeń w mózgach małp i fascynowała go „logika mózgu”[7]. W tym czasie prace matematyków i filozofów, takich jak Alan Turing, Alfred North Whitehead i Bertrand Russell, sugerowały głęboki związek między obliczeniami a logiką. Zdanie „Jeśli P jest prawdziwe I Q jest prawdziwe, to S jest prawdziwe” to przykładowe twierdzenie logiczne. Zakładano, że wszystkie obliczenia można sprowadzić do tego rodzaju logiki[8]. W kontekście takiego myślenia o obliczeniach McCullocha nurtowało pytanie: Jeśli mózg jest, jak uważają niektórzy, urządzeniem obliczeniowym, to w jaki sposób realizuje taką logikę?

Z tymi pytaniami McCulloch przeniósł się w 1941 roku z Uniwersytetu Yale na Uniwersytet Illinois, gdzie spotkał niezwykle utalentowanego nastolatka, Waltera Pittsa. Ten młody człowiek, już wtedy uznany logik (i „protegowany wybitnego logika matematycznego Rudolfa Carnapa”[9]), uczestniczył w seminariach prowadzonych przez ukraińskiego fizyka matematycznego Nicolasa Rashevsky’ego w Chicago[10]. Pitts był jednak „zagubionym młodym człowiekiem uciekającym od rodziny, która nie potrafiła docenić jego geniuszu”[11]. McCulloch i jego żona Rook przyjęli Waltera pod swój dach. „Zaowocowało to niekończącymi się wieczorami przy kuchennym stole McCullochów spędzanymi na próbach rozwikłania, jak działa mózg. W tym czasie córka McCullochów, Taffy, szkicowała małe rysunki”[12] — napisał informatyk Michael Arbib. Szkice Taffy posłużyły później do zilustrowania słynnego artykułu McCullocha i Pittsa z 1943 roku. Artykuł był zatytułowany *A Logical Calculus of the Ideas Immanent in Nervous Activity*.

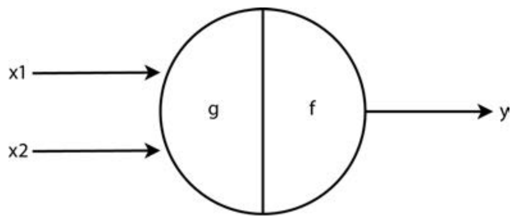
W tej pracy McCulloch i Pitts zaproponowali prosty model biologicznego neuronu[13]. Na rysunku 1.1 przedstawiona jest ilustracja typowego neuronu biologicznego.



Rysunek 1.1. Model neuronu biologicznego

Ciało komórki neuronu odbiera sygnały przez swoje rozgałęzione wypustki zwane dendrytami. Następnie przetwarza te sygnały wejściowe. W zależności od wyniku tego procesu może wysłać impuls elektryczny wzdłuż dłuższej wypustki zwanej aksonem. Sygnał ten przemieszcza się aksonem i dociera do jego rozgałęzionych zakończeń, gdzie jest przekazywany do dendrytów sąsiednich neuronów, po czym cały proces się powtarza. Neurony połączone w ten sposób tworzą biologiczną sieć neuronową.

McCulloch i Pitts przekształcili tę koncepcję w prosty model obliczeniowy w formie sztucznego neuronu[14]. Wykazali, że za pomocą jednego takiego sztucznego neuronu można zaimplementować podstawowe operacje logiki boolowskiej, takie jak I, LUB, NIE i inne, które są fundamentem obliczeń cyfrowych. Dla niektórych operacji logicznych, na przykład dla alternatywy rozłącznej (XOR), potrzeba więcej niż jednego sztucznego neuronu, ale o tym później. Na rysunku 1.2 znajduje się ilustracja pojedynczej neurody. Na razie zignoruj oznaczenia „g" i „f" wewnątrz neuronu; wrócę do nich wkrótce.



Rysunek 1.2. Jeden sztuczny neuron

W tej uproszczonej wersji modelu McCullocha–Pittsa $x1$ i $x2$ mogą przyjmować wartości 0 lub 1. Formalnie można to zapisać w następujący sposób:

$$x1, x2 \in \{0,1\}$$

Należy to interpretować w następujący sposób: $x1$ i $x2$ są elementami ze zbioru {0, 1}. Oznacza to, że zarówno $x1$, jak i $x2$ mogą przyjmować wyłącznie wartości 0 lub 1. Wyjście neuronu y jest obliczane w dwóch krokach. Najpierw należy zsumować wartości wejściowe, a następnie sprawdzić, czy otrzymana suma jest większa lub równa względem określonej wartości progowej, oznaczanej jako *theta* (θ). Jeśli ten warunek jest spełniony, y przyjmuje wartość 1. W przeciwnym razie y przyjmuje wartość 0.

$$suma = x1+x2$$

$$Jeśli\ suma \geq \theta: y = 1$$

$$W\ przeciwnym\ razie: y = 0$$

Po uogólnieniu tych reguł na dowolny ciąg danych wejściowych $x1, x2, x3, \ldots, xn$ można uzyskać formalny matematyczny opis prostego neuronu. Najpierw należy zdefiniować funkcję $g(x)$, czytaną „g od x", gdzie x to zbiór wejść $(x1, x2, x3, \ldots, xn)$. Ta funkcja sumuje otrzymane wejścia. Następnie należy zdefiniować funkcję $f(g(x))$, czytaną jako „f od g od x”, która na podstawie tej sumy sprawdza wartość progową i generuje wyjście y. Wynosi ono zero, jeśli $g(x)$ jest mniejsze od określonej wartości θ, oraz 1, jeśli $g(x)$ jest większe lub równe względem θ.

$$g(x) = x1 + x2 + x3 + \cdots + xn = \sum_{i=1}^{n} xi$$

$$f(z) = \begin{cases} 0, & z < \theta \\ 1, & z \geq \theta \end{cases}$$

$$y = f(g(x)) = \begin{cases} 0, & g(x) < \theta \\ 1, & g(x) \geq \theta \end{cases}$$

Za pomocą jednego sztucznego neuronu, jak opisałem wcześniej, można zaprojektować niektóre podstawowe bramki logiczne (na przykład I i LUB). W bramce logicznej I wyjście y powinno mieć wartość 1 tylko wtedy, gdy zarówno $x1$, jak i $x2$ są równe 1. W przeciwnym razie wyjście powinno wynosić 0. W tym scenariuszu wystarczy użyć ustawienia $\theta = 2$. Wtedy wyjście y będzie miało wartość 1 tylko wtedy, gdy $x1$ i $x2$ są równe 1 (tylko w tej sytuacji $x1+x2$ jest większe lub równe 2). Możesz eksperymentować z wartością θ, aby zaprojektować inne bramki logiczne. Na przykład w bramce LUB wyjście powinno mieć wartość 1, jeśli $x1$ lub $x2$ jest równe 1. W przeciwnym razie wyjście powinno być 0. Jaką wartość powinno mieć θ w tym scenariuszu?

Model McCullocha–Pittsa (MCP) można rozbudować na różne sposoby. Można na przykład zwiększyć liczbę wejść lub wprowadzić wejścia „hamujące”, w których wartości $x1$ lub $x2$ są mnożone przez –1. Jeśli jedno z wejść neuronu jest hamujące, to przy odpowiednio ustawionym progu neuron niezależnie od wartości pozostałych wejść zawsze będzie zwracał 0. Pozwala to na tworzenie bardziej złożonych układów logicznych. Podobny efekt można uzyskać dzięki połączeniu ze sobą wielu neuronów, tak aby wyjście jednego neuronu było wejściem dla innego.

Było to odkrywcze rozwiązanie, ale dawało ograniczone możliwości. Neuron w modelu MCP jest jednostką obliczeniową, a kombinacje takich neuronów pozwalają wykonywać dowolne operacje logiki boolowskiej. Ponieważ wszystkie obliczenia cyfrowe można sprowadzić do sekwencji takich operacji logicznych, to przez odpowiednie łączenie neuronów MCP można wykonywać dowolne obliczenia. Było to niezwykłe stwierdzenie jak na rok 1943. Matematyczne korzenie pracy McCullocha i Pittsa były oczywiste. Autorzy podali tylko trzy źródła: *The Logical Syntax of Language* Carnapa, *Foundations of Theoretical Logic* Davida Hilberta I Wilhelma Ackermanna oraz *Principia Mathematica* Whiteheada i Russella. Żaden z tych tekstów nie dotyczył biologii. Nie można było podważyć wyników przedstawionych w pracy McCullocha i Pittsa. Jednak efektem końcowym była maszyna, która potrafiła przeprowadzać obliczenia, ale nie umiała się uczyć. Przede wszystkim trzeba było ręcznie zaprogramować wartość θ. Neuron nie potrafił samodzielnie analizować danych i określać θ.

Nic dziwnego, że perceptron Rosenblatta zrobił takie wrażenie. Potrafił uczyć się wag na podstawie danych. Wagi te kodowały wiedzę, choćby minimalną, o wzorcach w danych i w pewnym sensie pozwalały ją zapamiętać.

NAUKA NA BŁĘDACH

Erudycja Rosenblatta często zdumiewała jego studentów. George Nagy, który trafił na Uniwersytet Cornella w Ithace w stanie Nowy Jork w 1960 roku, aby zrobić doktorat pod kierunkiem Rosenblatta, wspominał spacer, podczas którego rozmawiali o widzeniu stereoskopowym. Rosenblatt zaimponował Nagyowi dogłębną znajomością tematu. „Trudno było nie czuć się onieśmielonym w trakcie rozmowy z nim"[15] — powiedział Nagy, obecnie profesor emerytowany w Rensselaer Polytechnic Institute w Troy w stanie Nowy Jork. Imponująca wiedza Rosenblatta była tym bardziej zaskakująca ze względu na jego młody wiek — był zaledwie dziesięć lat starszy od Nagya.

Młody wygląd Rosenblatta o mało nie wpędził ich w kłopoty podczas podróży samochodem z Ithaki do Chicago na konferencję. Rosenblatt nie napisał jeszcze referatu, który miał przedstawić, więc poprosił Nagya o prowadzenie, a sam miał zamiar pracować. Nagy nigdy nie miał samochodu i ledwo umiał prowadzić, ale się zgodził. „Niestety, zjechałem ze swojego pasa i zatrzymał nas policjant" — wspominał Nagy. Rosenblatt powiedział funkcjonariuszowi, że jest profesorem i poprosił swojego studenta o prowadzenie. „Policjant się roześmiał i stwierdził: »Pan nie jest profesorem, tylko studentem«". Na szczęście Rosenblatt miał przy sobie wystarczająco dużo dokumentów, by przekonać policjanta o swoich kwalifikacjach, i ten ich puścił. Resztę drogi do Chicago prowadził już Rosenblatt. Na miejscu nie spał całą noc. Spędził ją na pisaniu referatu, który przedstawił następnego dnia. „Potrafił zrobić coś takiego" — powiedział mi Nagy.

Gdy Nagy przybył na Uniwersytet Cornella, Rosenblatt zbudował już system Mark I Perceptron. Jak wspomniałem we wstępie, Rosenblatt dokonał tego w 1958 roku, co zostało opisane w „The New York Timesie". Nagy rozpoczął pracę nad kolejną maszyną, o nazwie Tobermory (nazwaną tak na cześć mówiącego kota wymyślonego przez H.H. Munro, piszącego pod pseudonimem Saki). Tobermory był sprzętowym modelem sieci neuronowej zaprojektowanym do rozpoznawania mowy. W tym czasie Mark I Perceptron i koncepcje Rosenblatta zdążyły już wzbudzić spore zainteresowanie.

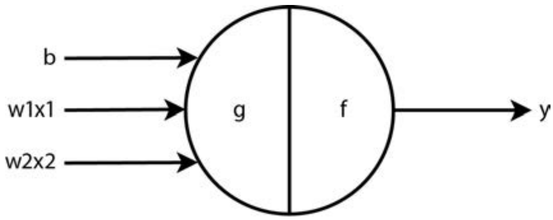
Latem 1958 roku redaktor magazynu „Research Trends", wydawanego przez Cornell Aeronautical Laboratory, poświęcił cały numer pracom Rosenblatta (jak stwierdził: „ze względu na wyjątkowe znaczenie artykułu dr. Rosenblatta"). Artykuł nosił tytuł *Projekt inteligentnego automatu. Poznajcie perceptron — maszynę, która odczuwa, rozpoznaje, zapamiętuje i reaguje podobnie jak ludzki umysł*[16]. Rosenblatt miał później żałować, że użył terminu „perceptron" do opisania efektów swojej pracy. „Jedną z rzeczy, których Rosenblatt najbardziej żałował, było użycie słowa brzmiącego jak nazwa maszyny" — powiedział mi Nagy. W rzeczywistości przez „perceptron" Rosenblatt rozumiał klasę modeli układu nerwowego służących do percepcji i wykonywania funkcji poznawczych.

Odwoływanie się przez Rosenblatta do mózgu nie było niczym zaskakującym. Naukowiec studiował pod okiem Jamesa Gibsona, jednego z wybitnych badaczy w dziedzinie percepcji wzrokowej. Podziwiał także McCullocha i Pittsa oraz Donalda Hebba, kanadyjskiego psychologa, który w 1949 roku przedstawił model uczenia się biologicznych neuronów. Warto zaznaczyć, że „uczenie się" w tym kontekście dotyczy rozpoznawania wzorców w danych, a nie złożonych procesów poznawczych kojarzonych z ludzkim umysłem. „[Rosenblatt] zawsze wypowiadał się o tych badaczach z wielkim uznaniem" — wspominał Nagy.

Choć McCulloch i Pitts opracowali modele neuronu, sieci złożone z tych sztucznych neuronów nie potrafiły się uczyć. W kontekście neuronów biologicznych Hebb zaproponował mechanizm uczenia, który często jest zwięźle, choć nie do końca precyzyjnie ujmowany w następującej formie: „neurony, które są razem aktywowane, łączą się ze sobą"[17]. W bardziej precyzyjnym ujęciu zgodnie z tym sposobem myślenia mózg może się uczyć, ponieważ połączenia między neuronami są wzmacniane, gdy sygnał wyjściowy jednego neuronu konsekwentnie przyczynia się do pobudzenia drugiego, a słabną, gdy tak nie jest. Proces ten nazywany jest uczeniem hebbowskim[18]. To właśnie Rosenblatt wykorzystał prace tych pionierów i połączył je w nową koncepcję: sztuczne neurony, które rekonfigurują się w trakcie uczenia i kodują informacje za pomocą siły połączeń.

Ponieważ Rosenblatt był psychologiem, nie miał dostępu do mocy obliczeniowej potrzebnej do symulacji swoich pomysłów za pomocą sprzętu lub oprogramowania. Dlatego w Laboratorium Aeronautycznym Cornell korzystał z komputera IBM 704, pięciotonowego kolosa wielkości pokoju. Współpraca z laboratorium okazała się owocna[19], gdyż prace Rosenblatta zwróciły uwagę fizyków, co zaowocowało publikacjami w czasopismach psychologicznych oraz w periodykach Amerykańskiego Towarzystwa Fizycznego. Ostatecznie Rosenblatt zbudował urządzenie Mark I Perceptron. Miało ono kamerę rejestrującą obraz o rozdzielczości 20×20 pikseli. Mark I potrafił rozpoznawać litery alfabetu na takich obrazach. Jednak zdaniem Nagya stwierdzenie, że Mark I „rozpoznawał" znaki, nie oddaje istoty rzeczy. W końcu systemy optycznego rozpoznawania znaków oferujące podobne możliwości były już dostępne komercyjnie w połowie lat 50. „Ważne jest to, że Mark I *uczył się* rozpoznawać litery na podstawie sygnału otrzymywanego, gdy popełniał błąd!"[20] — podkreślał Nagy w swoich wykładach.

Czym właściwie jest perceptron i jak się uczy? W najprostszej postaci jest to rozszerzony neuron McCullocha–Pittsa wzbogacony o algorytm uczenia[21]. Na rysunku 1.3 przedstawiam przykładowy perceptron z dwoma wejściami. Zauważ, że każde wejście jest mnożone przez odpowiadającą mu wagę. Istnieje też dodatkowe wejście, *b*, którego znaczenie omawiam dalej.



Rysunek 1.3. Perceptron z dwoma wejściami

Obliczenia wykonywane przez perceptron przebiegają następująco:

$$suma = w1x1 + w2x2 + b$$

$$Jeśli\ suma > 0: y = 1$$

$$W\ przeciwnym\ razie: y = -1$$

Bardziej ogólnie można to zapisać za pomocą notacji matematycznej:

$$g(x) = w1x1 + w2x2 + \cdots + wnxn + \mathrm{b} = \sum_{i=1}^{n} wixi + b$$

$$f(z) = \begin{cases} -1, z \leq 0 \\ 1, z > 0 \end{cases}$$

$$y = f(g(x)) = \begin{cases} -1, g(x) \leq 0 \\ 1, g(x) > 0 \end{cases}$$

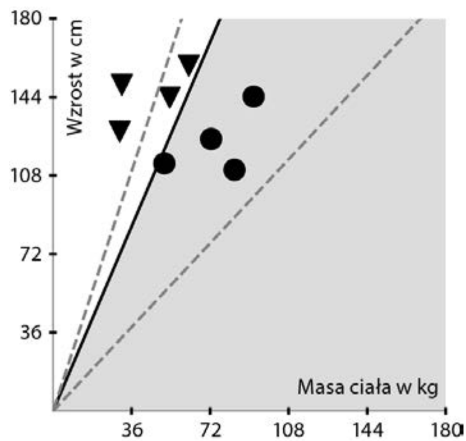
Główna różnica w porównaniu z wcześniej przedstawionym modelem MCP wiąże się z tym, że wejścia perceptronu nie muszą być binarne (0 lub 1), lecz mogą przyjmować dowolne wartości. Ponadto wejścia są mnożone przez odpowiadające im wagi, co daje sumę ważoną. Do tego dodawany jest jeszcze jeden składnik, b. Jest to wyraz wolny. Wyjście y przyjmuje wartość -1 lub $+1$ (zamiast 0 lub 1 w neuronie MCP). Co istotne, w odróżnieniu od neuronu MCP perceptron może nauczyć się odpowiednich wartości wag i wyrazu wolnego, by rozwiązać problem.

Aby zrozumieć, jak to działa, rozważ perceptron, który ma za zadanie sklasyfikować osobę jako otyłą ($y = +1$) lub nieotyłą ($y = -1$). Danymi wejściowymi są masa ciała osoby ($x1$) i jej wzrost ($x2$). Załóżmy, że zbiór danych zawiera sto pozycji, z których każda obejmuje masę ciała i wzrost danej osoby oraz etykietę określającą, czy według lekarza i zgodnie z wytycznymi instytucji National Heart, Lung, and Blood Institute osoba ta jest otyła. Zadaniem perceptronu jest nauczenie się wartości $w1$ i $w2$ oraz wyrazu wolnego b, by poprawnie sklasyfikować każdą osobę w zbiorze danych jako „otyłą” lub „nieotyłą”. Uwaga: analizowane są waga ciała i wzrost osoby, a jednocześnie piszę o wagach w perceptronie ($w1$ i $w2$). Podczas dalszej lektury należy pamiętać o tych dwóch różnych znaczeniach słowa „waga”.

Gdy perceptron nauczy się poprawnych wartości wag $w1$ i $w2$ oraz wyrazu wolnego, jest gotowy do generowania predykcji. Na podstawie wagi i wzrostu nowej osoby, która nie występowała w oryginalnym zbiorze danych (nie wystarczy więc sprawdzić etykiety w tabeli), perceptron może zaklasyfikować ją jako otyłą lub nie. Oczywiście model ten opiera się na pewnych założeniach, z których wiele dotyczy rozkładów prawdopodobieństwa (zagadnienie to omawiam w kolejnych rozdziałach). Perceptron przyjmuje jednak jedno podstawowe założenie, zgodnie z którym istnieje jednoznaczna liniowa granica między kategoriami osób klasyfikowanych jako otyłe i nieotyłe.

Co to oznacza w kontekście tego prostego przykładu? Jeśli naniesiesz na wykres *xy* wagę (na osi *x*) i wzrost (na osi *y*) badanych, by każdy z nich był reprezentowany przez punkt na wykresie, to założenie o „wyraźnym podziale" oznacza, że istnieje linia prosta separująca punkty reprezentujące osoby otyłe od punktów reprezentujących osoby nieotyłe. W takiej sytuacji mówimy, że zbiór danych jest liniowo separowalny.

Rysunek 1.4 przedstawia graficznie proces uczenia się perceptronu. Początkowo istnieją dwa zbiory punktów danych: jeden reprezentowany przez czarne kółka ($y = +1$, otyłość), drugi przez czarne trójkąty ($y = -1$, brak otyłości). Każdy punkt danych jest charakteryzowany przez parę wartości ($x1$, $x2$), gdzie $x1$ to masa ciała określonej osoby w kilogramach na osi *x*, a $x2$ to wzrost w centymetrach na osi *y*.



Rysunek 1.4. Proces uczenia się perceptronu

Perceptron zaczyna od wag $w1$ i $w2$ oraz wyrazu wolnego równego zero. Wagi i wyraz wolny reprezentują linię na płaszczyźnie *xy*. Następnie perceptron próbuje znaleźć linię separującą, zdefiniowaną przez określony zestaw wartości wag i wyrazu wolnego, która pozwala sklasyfikować punkty. Na początku perceptron klasyfikuje niektóre punkty poprawnie, a inne błędnie. Dwie z niepoprawnych prób są przedstawione za pomocą szarych przerywanych linii. W tym przykładzie widać, że w jednej próbie wszystkie punkty leżą po jednej stronie przerywanej linii, więc trójkąty są klasyfikowane poprawnie, ale kółka nie. W innej próbie kółka są klasyfikowane poprawnie, ale niektóre trójkąty błędnie. Perceptron uczy się na błędach i odpowiednio dostosowuje wagi oraz wyraz wolny. Po wielu przebiegach przez dane perceptron w końcu odkrywa co najmniej jeden zestaw poprawnych wartości wag i wyrazu wolnego. Znajduje linię, która rozdziela skupiska, dzięki czemu kółka i trójkąty leżą po przeciwnych stronach. Na rysunku obrazuje to ciągła czarna linia dzieląca przestrzeń współrzędnych na dwa obszary (jeden z nich jest wyróżniony szarym kolorem). Wagi, których nauczył się perceptron, określają nachylenie linii, a wyraz wolny wyznacza odległość linii od początku układu współrzędnych.

Po nauczeniu się korelacji między cechami fizycznymi (masa ciała i wzrost) a tym, czy dana osoba jest otyła ($y = +1$ lub -1), perceptron może ocenić, czy nowy badany, którego dane nie były używane podczas treningu, powinien zostać sklasyfikowany jako otyły. Oczywiście perceptron generuje wtedy najlepszą możliwą prognozę na podstawie wyuczonych wag i wyrazu wolnego, ale podana predykcja może być błędna. Czy potrafisz się domyślić, dlaczego tak się dzieje? Spróbuj dostrzec problem na podstawie analizy wykresu. (Podpowiedź: ile różnych linii można narysować, aby poprawnie oddzielić kółka od trójkątów?) Jak się przekonasz, istotą uczenia maszynowego jest minimalizacja błędów predykcji.

Powyższy opis dotyczy pojedynczego perceptronu, czyli jednego sztucznego neuronu. Może się to wydawać proste i zapewne się zastanawiasz, o co to całe zamieszanie. Wyobraź sobie jednak, że liczba wejść do perceptronu rośnie powyżej dwóch: ($x1$, $x2$, $x3$, $x4$ i tak dalej), gdzie każde wejście (xi) ma swoją własną oś. Nie da się już wtedy rozwiązać problemu za pomocą prostych obliczeń wykonanych w pamięci. Linia nie wystarcza już do oddzielenia dwóch skupisk danych, które teraz mają znacznie więcej wymiarów niż tylko dwa. Na przykład przy trzech wejściach ($x1$, $x2$, $x3$) dane są trójwymiarowe, dlatego potrzebna jest dwuwymiarowa płaszczyzna, aby rozdzielić punkty danych. W czterech lub więcej wymiarach potrzebujesz hiperpłaszczyzny (której nie sposób zwizualizować w trójwymiarowym umyśle). Taki wielowymiarowy odpowiednik jednowymiarowej linii prostej lub dwuwymiarowej płaszczyzny nazywany jest hiperpłaszczyzną.

Cofnijmy się teraz do roku 1958. Rosenblatt zbudował system Mark I Perceptron składający się z wielu opisanych jednostek. Urządzenie to potrafiło przetwarzać obraz o rozdzielczości 20×20 pikseli, składający się więc łącznie z 400 pikseli, z których każdy miał wartość wejściową x. Tak więc Mark I przyjmował jako dane wejściowe długi ciąg wartości: $x1$, $x2$, $x3$, …, $x400$. Złożony układ sztucznych neuronów, zarówno o stałych losowych wagach, jak i wagach podlegających uczeniu, przekształcał ten wektor 400 wartości w sygnał wyjściowy, który można było wykorzystać do rozpoznawania wzorców w obrazie. To oczywiście bardzo uproszczony opis. Niektóre obliczenia były na tyle skomplikowane, że wymagały użycia komputera IBM 704. Szczegóły tej architektury omawiam w rozdziale 10. Mark I potrafił nauczyć się kategoryzować litery alfabetu zakodowane w wartościach pikseli. Cała opisana logika, przeskalowana do obsługi 400 wejść, była zaimplementowana sprzętowo. Maszyna po zakończeniu procesu uczenia (który opisuję w następnym rozdziale) przechowywała wiedzę w sile (wagach) połączeń. Nic więc dziwnego, że wszyscy dali się ponieść wyobraźni.

Jeśli jednak dokładnie przeanalizować, czego uczy się perceptron, jego ograniczenia — naturalnie z perspektywy czasu — stają się oczywiste. Algorytm pozwala perceptronowi nauczyć się korelacji między wartościami ($x1$, $x2$, …, $x400$) a odpowiadającą im wartością y, o ile takie korelacje występują w danych. To prawda, że uczy się tych korelacji, choć nikt ich mu nie wskazuje, ale nadal są to tylko korelacje. Czy ich identyfikowanie jest tym samym, co myślenie i rozumowanie? Choć Mark I odróżniał literę „B" od „G", robił to na podstawie wzorców i nie przypisywał im żadnego znaczenia, które mogłoby prowadzić do jakiegoś wnioskowania. Pytania z tego obszaru leżą u podstaw współczesnej debaty nad ograniczeniami głębokich sieci neuronowych, zadziwiających potomków perceptronu. Istnieje ścieżka łącząca wczesne perceptrony z technologią dużych modeli językowych czy sztuczną inteligencją rozwijaną na przykład na potrzeby samochodów autonomicznych. Jest to długa i zawiła ścieżka, z wieloma ślepymi zaułkami i błędnymi skrętami. Niemniej jednak jest ona fascynująca i intrygująca, a zaraz zacznę jej omawianie.

Skonstruowanie perceptronu było znaczącym osiągnięciem. Jeszcze większym sukcesem okazał się matematyczny dowód, że pojedyncza warstwa perceptronów zawsze znajdzie liniowo separującą hiperpłaszczyznę, *jeśli* dane są liniowo separowalne. Aby zrozumieć ten dowód, należy zapoznać się z pojęciem wektorów i ich znaczeniem jako podstawowej metody reprezentacji danych w uczeniu maszynowym. Będzie to pierwszy krok w świat matematyki.


PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA
Helion

Po zaledwie kilku minutach lektury Dlaczego maszyny się uczą? poczujesz, jak Twoje połączenia neuronowe zaczynają się aktualizować. Gdy skończysz czytać, odczujesz efekty głębokiego uczenia się na sobie — z ogromną przyjemnością i wnikliwym zrozumieniem tematu.

— Steven Strogatz, autor bestsellerów z list „New York Timesa", w tym *Potęgi nieskończoności*

Sztuczna inteligencja coraz silniej wpływa na nasze życie. Systemy oparte na uczeniu maszynowym decydują o przyznaniu kredytu, wspierają diagnozowanie chorób, odgrywają coraz większą rolę w odkryciach z zakresu biologii, chemii, fizyki czy medycyny. Skokowy rozwój dużych modeli językowych, takich jak ChatGPT, Gemini czy Claude, pozwala używać AI do zadań, które jeszcze niedawno były zarezerwowane wyłącznie dla ludzi.

Jak bardzo złożona musi być matematyka stojąca za tymi potężnymi i zadziwiającymi technologiami? Dzięki lekturze **Dlaczego maszyny się uczą?** przekonasz się, że fundamenty sztucznej inteligencji opierają się na stosunkowo prostych ideach matematycznych, zaczerpniętych z algebry liniowej, rachunku różniczkowego i całkowego, rachunku prawdopodobieństwa, ze statystyki, a także z teorii optymalizacji. Ta błyskotliwa książka pokazuje, jak przez lata kształtowały się współczesne systemy AI i jak w rzeczywistości działają jej kluczowe mechanizmy. Odkryjesz też fascynujące powiązania między sztuczną a naturalną inteligencją — ich podstawy bywają zbliżone bardziej, niż się wydaje.

W matematyce odnajdziesz nie tylko wyjaśnienie działania AI, lecz również klucz do zrozumienia jej możliwości i ograniczeń. A to wiedza niezbędna, by z niej korzystać świadomie, skutecznie — i bezpiecznie.

Anil Ananthaswamy jest uznanym dziennikarzem naukowym i autorem bestsellerowych książek. Regularnie publikuje w „Nature", „Scientific American" i „New Scientist". Specjalizuje się w popularyzacji fizyki, kosmologii i neuronauki. Za swoją działalność w obszarze dziennikarstwa naukowego otrzymał prestiżową nagrodę American Institute of Physics.

Lor
em
ips

Helion

helion.pl

HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej!

ISBN 978-83-289-3002-5
9 788328 930025

Cena: 69,00 zł