Spis treści

Część 2 **Regresja**

Część 5 **Uczenie nienadzorowane**

O autorach

Fred Nwanganga jest profesorem uczelni na wydziale Business Analytics w Mendoza College of Business na uniwersytecie Notre Dame, gdzie prowadzi kursy dla studentów i absolwentów z zakresu zarządzania danymi, uczenia maszynowego i analizy danych nieustrukturyzowanych. Ma ponad 15-letnie doświadczenie w pełnieniu roli lidera technicznego zarówno w sektorze prywatnym, jak i akademickim. Fred uzyskał stopień doktora w zakresie informatyki na uniwersytecie Notre Dame.

Mike Chapple jest profesorem uczelni na wydziale Technology, Analytics, and Operations w Mendoza College of Business na uniwersytecie Notre Dame. Mike ma ponad 20-letnie doświadczenie w pracy na stanowiskach technicznych w sektorze publicznym i prywatnym. Pełni funkcję dyrektora naukowego programu studiów magisterskich z analizy biznesowej i jest autorem ponad 25 książek. Mike zdobył tytuł doktora w zakresie informatyki na uniwersytecie Notre Dame.

O redaktorach technicznych

Everaldo Aguiar uzyskał stopień doktora na uniwersytecie Notre Dame, gdzie współpracował z Interdisciplinary Center for Network Science and Applications. Uczestniczył w projekcie Data Science for Social Good, a teraz zajmuje stanowisko Principal data science manager w SAP Concur, gdzie kieruje zespołem analityków danych, którzy rozwijają, instalują, utrzymują i oceniają rozwiązania uczenia maszynowego wbudowane w wykorzystywane przez klientów produkty.

Seth Berry jest profesorem uczelni na wydziale Information Technology, Analytics, and Operations uniwersytetu Notre Dame. Jest zapalonym użytkownikiem języka R (od tylu lat, że pamięta jeszcze, gdy stosowanie Tinn-R było dobrym pomysłem) i z radością realizuje prawie wszystkie napotkane statystyczne zadania programistyczne. Szczególnie interesuje się wszelkiego rodzaju analizą tekstu oraz tym, w jaki sposób działania użytkowników w Internecie pozwalają przewidzieć decyzje podejmowane w rzeczywistości.

Podziękowania

Opracowanie książki wymaga zaangażowania wielu osób i jesteśmy wdzięczni tym, z którymi mieliśmy przyjemność współpracować w tym projekcie.

Przede wszystkim dziękujemy naszym rodzinom, które ponownie dzielnie znosiły sytuację, gdy przygotowywaliśmy książkę do druku. Chcielibyśmy także podziękować kolegom z wydziału Information Technology, Analytics, and Operations w Mendoza College of Business na uniwersytecie Notre Dame. Duża część tej książki została zapoczątkowana przez rozmowy na uniwersyteckich korytarzach i cieszymy się, że jesteście w naszym życiu.

Jim Minatel, redaktor nabywający w wydawnictwie Wiley, miał kluczowe znaczenie dla powstania tej książki. Mike od wielu lat współpracuje z Jimem i jest mu wdzięczny za nieustające wsparcie. Fred po raz pierwszy miał okazję współpracować z wydawnictwem Wiley i było to naprawdę niezwykłe, satysfakcjonujące przeżycie.

Nasza agentka Carole Jelen z Waterside Productions jest nadal cennym partnerem, pomagając w otwieraniu nowych możliwości, takich jak ta.

Nasi redaktorzy techniczni Seth Berry oraz Everaldo Aguiar dzielili się swoimi cennymi uwagami, gdy pracowaliśmy nad tą książką. Dziękujemy za znaczący wkład w ten projekt.

Nasi asystenci badawczy Nicholas Schmit i Yun "Jessica" Yan spisali się znakomicie, dokonując przeglądu literatury i zbierając dodatkowe materiały do tej książki.

Chcielibyśmy podziękować również zespołowi wsparcia w wydawnictwie Wiley, a w szczególności redaktorce projektu Kezii Endsley oraz redaktorowi produkcji Vasanth Koilraj. Byliście spoiwem, dzięki któremu projekt był realizowany zgodnie z harmonogramem.

– Fred i Mike

Wprowadzenie

Uczenie maszynowe zmienia świat. Wszystkie organizacje, zarówno duże, jak i małe, chcą czerpać wiedzę z ogromnej ilości informacji, jakie zachowują i przetwarzają na co dzień. Pragnienie, by przewidywać przyszłość, napędza do działania analityków biznesowych i specjalistów data science w różnych branżach, od marketingu po opiekę zdrowotną. Napisaliśmy tę książkę po to, by pomóc szerszemu gronu odbiorców poznać narzędzia analizy.

Język programowania R jest językiem zaprojektowanym w konkretnym celu, aby wspierać analizę statystyczną i uczenie maszynowe. Wybraliśmy go w tej książce nie tylko ze względu na dużą popularność w branży, ale również na intuicyjność, w szczególności dla osób, dla których jest on pierwszym językiem programowania.

Dostępnych jest wiele książek omawiających praktyczne zastosowania uczenia maszynowego, napisanych z myślą o ludziach biznesu i osób postronnych. Analogicznie istnieje wiele bardzo technicznych publikacji, które zagłębiają się w matematyczne i informatyczne aspekty uczenia maszynowego. W tej książce próbujemy zbudować pomost między tymi dwoma światami. Przedstawiamy czytelnikom intuicyjne wprowadzenie do uczenia maszynowego z myślą o praktycznych zastosowaniach uczenia maszynowego w dzisiejszym świecie. Równocześnie nie stronimy od kodu. Podobnie jak w prowadzonych przez nas kursach magisterskich i podyplomowych, staramy się, by język programowania R stał się przystępny dla każdego. Mamy nadzieję, że będziecie czytać tę książkę z otwartym laptopem pod ręką, podążając za naszymi przykładami i próbując zrealizować ćwiczenia.

Powodzenia w początkach przygody z uczeniem maszynowym!

Co zawiera ta książka?

Niniejsza książka stanowi wprowadzenie do uczenia maszynowego z wykorzystaniem języka programowania R.

Rozdział 1: Czym jest uczenie maszynowe? Ten rozdział wprowadza czytelników w świat uczenia maszynowego i opisuje, w jaki sposób uczenie maszynowe umożliwia odkrywanie wiedzy ukrytej w danych. W tym rozdziale wyjaśniamy różnice między uczeniem nadzorowanym, uczeniem nienadzorowanym i uczeniem przez wzmacnianie. Opisujemy różnice między problemami klasyfikacji i regresji oraz wytłumaczymy, jak mierzyć wydajność algorytmów uczenia maszynowego.

Rozdział 2: Wprowadzenie do języka R i RStudio W tym rozdziale zapoznamy czytelników z językiem programowania R i zestawem narzędzi, które będziemy wykorzystywać w pozostałej części książki. Przedstawimy język R z perspektywy początkującego programisty, wyjaśnimy, jak korzystać ze zintegrowanego środowiska deweloperskiego RStudio i przeprowadzimy czytelników przez proces tworzenia i wykonywania pierwszych skryptów R. Wyjaśnimy także, jak używać pakietów do redystrybucji kodu R i korzystać z różnych typów danych w języku R.

Rozdział 3: Zarządzanie danymi Ten rozdział przedstawia koncepcję zarządzania danymi i wykorzystywania języka R do zbierania danych i zarządzania nimi. Wprowadzimy pakiet tidyverse, czyli kolekcję pakietów R wspierających procesy analityczne oraz omówimy różne sposoby opisywania i wizualizowania danych w języku R. Pokażemy również, jak czyścić, przekształcać i ograniczać dane, by przygotowywać je na potrzeby uczenia maszynowego.

Rozdział 4: Regresja liniowa W tym rozdziale wkroczymy w świat nadzorowanego uczenia maszynowego, poznając regresję liniową. Wyjaśnimy podstawowe reguły statystyczne, na których opiera się regresja liniowa i zademonstrujemy, jak dopasowywać modele prostej i wielokrotnej regresji w języku R. Wyjaśnimy również, jak oceniać, interpretować i wykorzystywać wyniki modeli regresji.

Rozdział 5: Regresja logistyczna Regresja liniowa pomaga w rozwiązywaniu problemów, które wymagają przewidywania wartości liczbowych, jednak słabo radzi sobie z problemem predykcji kategorialnych. W tym rozdziale opiszemy regresję logistyczną, czyli technikę predykcji kategorialnych. Omówimy zastosowania uogólnionych modeli liniowych i pokażemy, jak budować modele regresji logistycznej w języku R. Wyjaśnimy również, jak oceniać, interpretować i ulepszać wyniki modeli regresji logistycznej.

Rozdział 6: k najbliższych sąsiadów Technika k najbliższych sąsiadów służy do przewidywania klasyfikacji punktów danych na podstawie klasyfikacji innych, podobnych punktów danych. W tym rozdziale opisujemy, jak działa algorytm *k*-NN i zademonstrujemy, jak budować model *k* -NN w języku R. Pokazujemy również, jak stosować ten model, dokonując predykcji, do jakich klas przynależą nowe punkty danych.

Rozdział 7: Naiwny klasyfikator Bayesa Algorytm naiwnego klasyfikatora Bayesa wykorzystuje tabelę prawdopodobieństw do przewidywania szansy, że wystąpienie należy do określonej klasy. W tym rozdziale omówimy koncepcje prawdopodobieństwa łącznego i warunkowego oraz opiszemy, w jaki sposób działa naiwny klasyfikator Bayesa. Zademonstrujemy, jak zbudować naiwny klasyfikator Bayesa w języku R i użyć go do dokonywania predykcji dotyczących niewidzianych wcześniej danych.

Rozdział 8: Drzewa decyzyjne Drzewa decyzyjne są popularną techniką modelowania, ponieważ dają intuicyjne rezultaty. W tym rozdziale opiszemy proces tworzenia i interpretowania modeli drzew decyzyjnych. Wyjaśnimy również proces rozwijania drzewa w języku R i przycinania go w celu zwiększenia potencjału uogólniania modelu.

Rozdział 9: Ocenianie wydajności Nie ma idealnej techniki modelowania. Każda ma jakieś mocne i słabe strony oraz wprowadza nowe możliwości predykcji dla różnego typu problemów. W tym rozdziale omawiamy proces oceniania wydajności modelu. Wprowadzamy techniki ponownego próbkowania i wyjaśniamy, jak można wykorzystywać je do szacowania przyszłej wydajności modelu. Zademonstrujemy również, jak wizualizować i oceniać wydajność modelu w języku R.

Rozdział 10: Ulepszanie wydajności Gdy mamy już narzędzia do oceniania wydajności modelu, możemy zacząć używać ich do ulepszania tej wydajności. W tym rozdziale przyglądamy się technikom dostrajania modeli uczenia maszynowego. Demonstrujemy również, jak można zwiększyć wartość predykcyjną, wykorzystując możliwości predykcyjne różnych modeli.

Rozdział 11: Odkrywanie wzorców za pomocą reguł asocjacyjnych Reguły asocjacyjne pomagają w odkrywaniu wzorców ukrytych w zbiorze danych. W tym rozdziale opisujemy tę metodę i demonstrujemy, jak w języku R generować reguły asocjacyjne dla zbioru danych. Wyjaśnimy również metody oceniania i kwantyfikowania siły reguł asocjacyjnych.

Rozdział 12: Grupowanie danych poprzez klasteryzację Klasteryzacja jest techniką nienadzorowanego uczenia maszynowego, która grupuje elementy w oparciu o ich podobieństwo. W tym rozdziale wyjaśnimy, w jaki sposób algorytm klasteryzacji metodą *k*-średnich segmentuje dane oraz zademonstrujemy, jak użyć klasteryzacji metodą *k*-średnich w języku R.

Materiały pomocnicze dla czytelników książki

By czerpać jak największe korzyści z lektury, warto sięgnąć po materiały pomocnicze dla studentów i instruktorów dostępne na stronie tej książki. Zachęcamy również do podzielenia się z nami konstruktywnymi opiniami, w jaki sposób moglibyśmy ulepszyć tę książkę.

Pliki pomocnicze do pobrania

Analizując omawiane w tej książce przykłady, można albo wpisywać kod własnoręcznie, albo użyć plików kodu źródłowego znajdujących się w materiałach pomocniczych. Implementując przykłady, warto skorzystać również z tych samych zbiorów danych,

jakich użyliśmy w książce. Cały kod źródłowy i wszystkie zbiory danych wykorzystywane w tej książce są dostępne do pobrania ze strony *www.wiley.com/go/pmlr*.

Jak skontaktować się z wydawcą

Jeśli uważasz, że znalazłeś błąd w tej książce, proszę daj nam znać. W wydawnictwie John Wiley & Sons rozumiemy, jak ważne jest dostarczanie naszym klientom prawidłowych treści, ale mimo największych starań błędy mogą się przydarzyć.

Aby zgłosić potencjalną erratę, wyślij proszę email do naszego zespołu obsługi klientów na adres *wileysupport@wiley.com* z „Possible Book Errata Submission" w temacie.

Część I

Zaczynamy

ROZDZIAŁ

1

Czym jest uczenie maszynowe?

Witamy w świecie *uczenia maszynowego*! Rozpoczynamy ekscytującą przygodę, odkrywając, w jaki sposób specjaliści data science wykorzystują algorytmy do ujawniania wiedzy ukrytej w pokładach danych, jakie są generowane każdego dnia przez firmy, organizacje i osoby prywatne.

Często napotykamy sytuację, gdy mamy do czynienia z ogromnym zbiorem danych, który w naszym przekonaniu zawiera ważne informacje, ale nie wiemy, jak znaleźć tę igłę wiedzy w przysłowiowym stogu siana. Tu właśnie uczenie maszynowe może okazać się pomocne. Ta książka ma na celu przekazanie wiedzy i umiejętności, które pozwolą wykorzystać potencjał algorytmów uczenia maszynowego. Opisuje różne rodzaje problemów, które stanowią doskonałą okazję do zastosowania rozwiązań uczenia maszynowego, a także różne techniki uczenia maszynowego, które najlepiej sprawdzą się w rozwiązywaniu różnego typu problemów.

Co więcej, prezentuje pragmatyczne podejście do tego złożonego, technicznego zagadnienia. W tej książce nie będziemy rozwodzić się nad zawiłymi matematycznymi aspektami tych algorytmów. Zamiast tego skoncentrujemy się na tym, jak można od razu zacząć wykorzystywać omawiane algorytmy do własnych celów. Przedstawimy także język programowania R, który naszym zdaniem doskonale nadaje się do rozwiązywania problemów uczenia maszynowego z praktycznego punktu widzenia. Ale jeszcze nie pora na programowanie czy język R, zajmiemy się nimi w rozdziale 2. Na razie czas rozpocząć pracę i pomóc w lepszym zrozumieniu, jak działa uczenie maszynowe.

W tym rozdziale będzie można się dowiedzieć:

- Jak sposób uczenie maszynowe umożliwia odkrywanie wiedzy w danych
- Czym różnią się od siebie techniki uczenia nienadzorowanego, uczenia nadzorowanego i uczenia przez wzmacnianie
- Jakie są różnice między problemami klasyfikacji i regresji
- Jak zmierzyć wydajność algorytmów uczenia maszynowego
- W jaki sposób walidacja krzyżowa podnosi dokładność modeli uczenia maszynowego

Odkrywanie wiedzy w danych

Świat uczenia maszynowego koncentruje się na wykorzystywaniu algorytmów do odkrywania w zbiorach danych wiedzy, która może pomóc w podejmowaniu świadomych decyzji w przyszłości. I to niezależnie od specjalistycznej dziedziny, w jakiej działamy, ponieważ uczenie maszynowe ma zastosowanie w wielu różnych dziedzinach. Oto przykładowe sytuacje, w których uczenie maszynowe może się przydać:

- Segmentacja klientów i ustalanie, jakie komunikaty marketingowe mogą najbardziej przemawiać do różnych grup klientów.
- Wykrywanie w dziennikach systemu i aplikacji anomalii, które mogą sygnalizować zagrożenie dla cyberbezpieczeństwa.
- Prognozowanie sprzedaży produktów w oparciu o warunki rynkowe i środowiskowe.
- Polecanie kolejnego filmu, który klient może chcieć obejrzeć, w oparciu o jego poprzednie aktywności i preferencje podobnych klientów.
- Ustalanie z dużym wyprzedzeniem cen pokoi hotelowych na podstawie prognozowanego popytu.

Oczywiście to tylko kilka wybranych przykładów. Uczenie maszynowe warto stosować w praktycznie każdej dziedzinie, w której odkrywanie nowej wiedzy może przynosić korzyści. A trudno znaleźć dziedzinę, w której nie można uczynić pożytku z dodatkowej wiedzy!

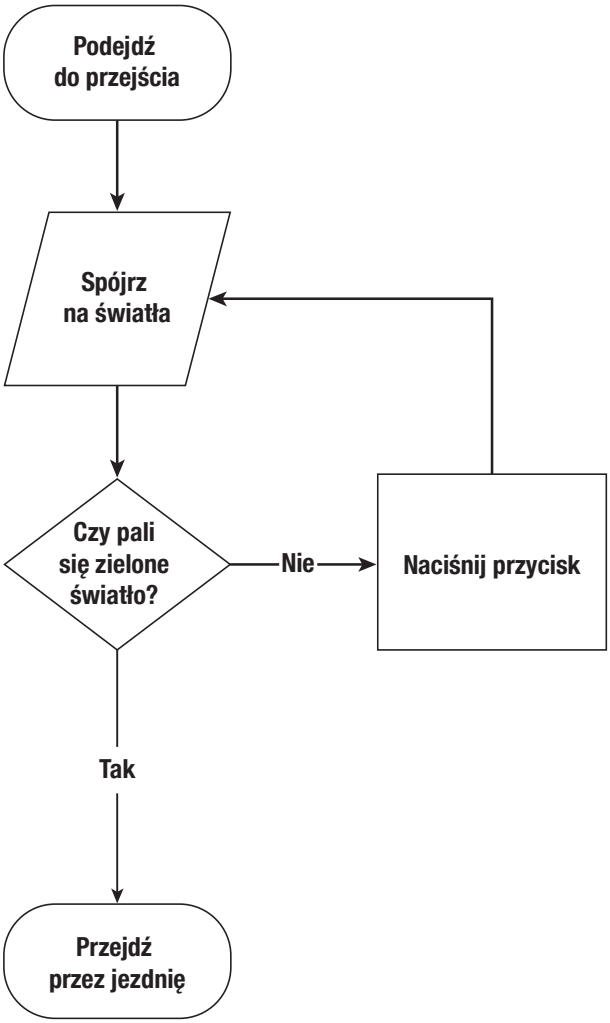
Wprowadzenie do algorytmów

W dalszej części książki techniki uczenia maszynowego będziemy często nazywać *algorytmami*. Ten termin, wywodzący się ze świata informatyki, wielokrotnie pojawia się także w świecie data science, dlatego trzeba go rozumieć. Choć termin ten może sygnalizować techniczną złożoność, koncepcja algorytmu jest tak naprawdę prosta. Można zaryzykować stwierdzenie, że czytelnicy wykorzystują jakąś postać algorytmu prawie każdego dnia.

Algorytm to po prostu szereg kroków podejmowanych w ramach realizacji procesu. Termin ten jest najczęściej stosowany w odwołaniu do kroków realizowanych przez komputer, gdy przeprowadza on zadanie obliczeniowe, ale wiele codziennych działań również można traktować jak algorytmy. Na przykład, gdy spacerujemy po dużym mieście i dochodzimy do skrzyżowania, realizujemy algorytm przechodzenia przez ulicę. Rysunek 1.1 ilustruje, jak przykładowo może wyglądać ten proces.

Oczywiście algorytmy w świecie informatyki są bardziej złożone i są implementowane poprzez pisanie oprogramowania, ale można postrzegać je w ten sam sposób. Algorytm to po prostu serie precyzyjnych obserwacji, decyzji i instrukcji, które informują komputer, w jaki sposób ma wykonać działanie. Algorytmy uczenia maszynowego

projektujemy po to, aby odkryć wiedzę ukrytą w danych. W tej książce omówimy wiele różnego typu algorytmów uczenia maszynowego i pokażemy jak, na bardzo różne sposoby osiągają one ten cel.



Rysunek 1.1 Algorytm przechodzenia przez ulicę

Sztuczna inteligencja, uczenie maszynowe i głębokie uczenie

Terminy sztuczna inteligencja, uczenie maszynowe i głębokie uczenie są używane niemal zamiennie do opisywania różnych technik wiążących się z przetwarzaniem danych przy użyciu komputera. Ale skoro wkraczamy w świat data science warto lepiej zrozumieć te terminy.

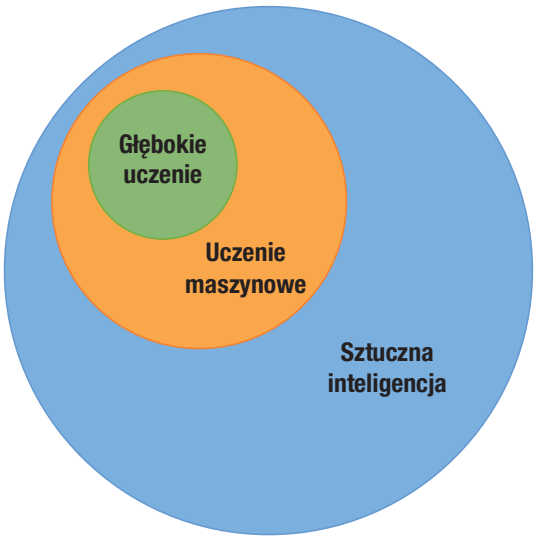
Sztuczna inteligencja (ang. artificial intelligence – AI) obejmuje różnego rodzaju techniki, w których system komputerowy ma imitować ludzkie zachowanie. Jak sama nazwa wskazuje, chcemy, aby systemy komputerowe sztucznie zachowywały się tak, jakby były inteligentne. Oczywiście współczesne komputery nie potrafią

przeprowadzać złożonego wnioskowania, jakie zachodzi w ludzkim umyśle, ale mogą próbować naśladować pewne małe fragmenty ludzkiego zachowania i oceniania.

Uczenie maszynowe (ang. machine learning – ML) jest podzbiorem technik sztucznej inteligencji, które stosują reguły statystyczne na problemach danych, próbując odkryć nową wiedzę poprzez dokonywanie generalizacji na bazie przykładów. Innymi słowy, techniki uczenia maszynowego to techniki sztucznej inteligencji zaprojektowane tak, by się uczyć.

Głębokie uczenie (ang. deep learning) to kolejny podzbiór uczenia maszynowego, w którym zbiór złożonych technik, nazywanych *sieciami neuronowymi*, jest wykorzystywany do odkrywania wiedzy w określony sposób. Jest to wysoce wyspecjalizowana gałąź uczenia maszynowego służąca przede wszystkim do analizy obrazów, wideo i dźwięku.

Rysunek 1.2 przedstawia relację między tymi dziedzinami. Ta książka koncentruje się na technikach uczenia maszynowego, a w szczególności na tych kategoriach uczenia maszynowego, które *nie* podpadają pod definicję głębokiego uczenia.



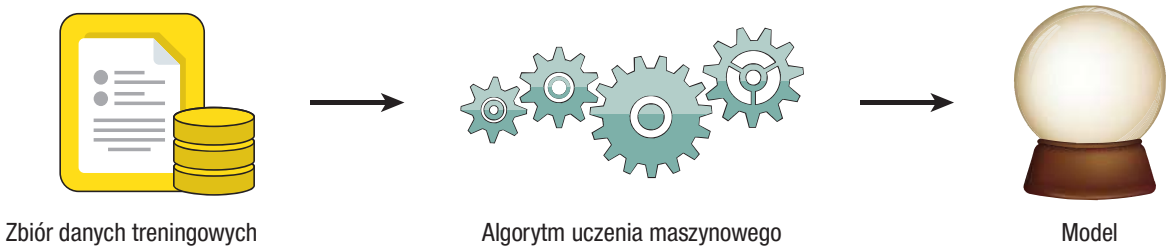
Rysunek 1.2 Zależność między sztuczną inteligencją, uczeniem maszynowym a głębokim uczeniem

Techniki uczenia maszynowego

Omawiane w tej książce techniki uczenia maszynowego można podzielić na dwie główne kategorie. Algorytmy uczenia nadzorowanego uczą się wzorców w oparciu o oznaczone (opatrzone etykietami) przykłady danych historycznych. Algorytmy uczenia nienadzorowanego próbują odkrywać wzorce bez użycia oznaczonych danych. Przyjrzyjmy się dokładniej powyższym technikom.

Uczenie nadzorowane

Techniki *uczenia nadzorowanego* są prawdopodobnie najczęściej stosowaną kategorią algorytmów uczenia maszynowego. Te techniki mają na celu wykorzystanie istniejącego zbioru danych do wygenerowania modelu, który pomaga nam w opracowywaniu predykcji dotyczących przyszłych nieoznaczonych danych. A bardziej formalnie, do algorytmu nadzorowanego uczenia maszynowego przekazujemy dane wejściowe w postaci *zbioru danych treningowych*. Następnie algorytm wykorzystuje te dane treningowe do przygotowania danych wyjściowych, czyli *modelu*, jak pokazano na rysunku 1.3.



Rysunek 1.3 Ogólny model uczenia nadzorowanego

Model wygenerowany przez algorytm nadzorowanego uczenia maszynowego jest jak magiczna kula – gdy ją mamy, możemy formułować predykcje dotyczące danych. Rysunek 1.4 pokazuje, jak to działa. Gdy już mamy model, możemy wykorzystać go do dokonywania predykcji dotyczących nowego elementu w oparciu o wiedzę pozyskaną ze zbioru danych treningowych.

Rysunek 1.4 Sporządzanie prognoz przy użyciu modelu uczenia nadzorowanego

Te techniki opisujemy terminem *nadzorowane*, ponieważ wykorzystujemy zbiór danych treningowych do nadzorowania procesu tworzenia modelu. Ten zbiór danych treningowych zawiera etykiety, które pomagają w przewidywaniu.

Przedstawmy konkretny przykład, aby utrwalić tę wiedzę. Załóżmy, że jesteśmy odpowiedzialni za udzielanie kredytów w salonie samochodowym, jak na rysunku 1.5. Sprzedawcy w salonie oferują auta klientom indywidualnym. Klienci często nie dysponują oszczędnościami, które pozwoliłyby im kupić auto za gotówkę, dlatego potrzebują

finansowania. Naszym zadaniem jest dopasowanie klientów do odpowiedniego produktu kredytowego. Mamy do dyspozycji trzy opcje:

- Kredyty wysokiego ryzyka są najwyżej oprocentowane i oferowane klientom, którzy z dużym prawdopodobieństwem nie będą spłacać rat na czas albo okażą się niewypłacalni.
- Najlepsze pożyczki są najniżej oprocentowane i oferowane klientom, którzy raczej nie powinni mieć problemów z terminowym opłacaniem rat i z bardzo wysokim prawdopodobieństwem spłacą kredyt.
- Standardowe pożyczki są oferowane klientom, którzy mieszczą się między tymi dwoma grupami i ich oprocentowanie również leży gdzieś pośrodku.

Rysunek 1.5 Wykorzystanie uczenia maszynowego do klasyfikowania klientów salonu samochodowego

Otrzymujemy od sprzedawców wnioski kredytowe i musimy od razu podjąć decyzję. Jeśli będziemy zwlekać, klient może zrezygnować i udać się do innego salonu. Jeśli

zaproponujemy klientowi pożyczkę o wyższym stopniu ryzyka niż dostaje zazwyczaj, klient może wybrać inny salon oferujący niższe oprocentowanie. Z drugiej strony, jeśli zaoferujemy klientowi niższe oprocentowanie niż na to zasługuje, możemy stracić na transakcji, jeśli nie spłaci on kredytu.

Do tej pory proces biznesowy polegał na przeglądaniu historii kredytowej klienta i podejmowaniu decyzji o kategorii pożyczki w oparciu o wieloletnie doświadczenie zawodowe. Zgodnie z podejściem: „Wiele już widzieliśmy i podejmując ważne decyzje biznesowe, możemy kierować się instynktem". Jednak, jako specjaliści data science, mamy świadomość, że lepszym sposobem rozwiązywania tego problemu może być zastosowanie uczenia maszynowego.

Salon samochodowy może użyć nadzorowanego uczenia maszynowego jako pomocy w realizacji tego zadania. Przede wszystkim potrzebujemy zbioru danych treningowych zawierającego informację o dotychczasowych klientach i o tym, w jaki sposób spłacali oni pożyczki. Im więcej danych można dołączyć do zbioru danych treningowych, tym lepiej. Dostępność danych z kilku lat pomogłaby w opracowaniu wysokiej jakości modelu.

Zbiór danych mógłby zawierać różne informacje o każdym z klientów m.in.: przybliżony wiek, ocenę ryzyka kredytowego, posiadane nieruchomości i typ samochodu. Każdy z tych punktów danych nazywany jest *cechą* (ang. feature) tego klienta i składa się na dane wejściowe modelu uczenia maszynowego tworzonego przez algorytm. Zbiór danych musi zawierać również *etykietę* (ang. label) dla każdego z klientów w zbiorze danych treningowych. Etykiety są wartościami, do których przewidywania ma służyć model. W tym przypadku dostępne są dwie etykiety: nie spłacił (Default) i spłacił (Repaid). Każdy klient w zbiorze danych treningowych jest oznaczony etykietą opisującą status jego pożyczki. Jeśli spłacił swoją pożyczkę, jest opatrzony etykietą „Repaid", a jeśli zalega, etykietą „Default".

Mały fragment otrzymanego zbioru danych został przedstawiony na rysunku 1.6. Warto zwrócić uwagę na dwie rzeczy. Po pierwsze, każdy wiersz w zbiorze danych odpowiada jednemu klientowi i wszyscy oni są *dawnymi* klientami, dla których termin zwrotu pożyczki już upłynął. Znamy konsekwencje udzielenia pożyczek tym klientom, dzięki czemu mamy etykiety, których potrzebujemy do trenowania modelu uczenia nadzorowanego. Po drugie, każda z cech zawartych w modelu jest właściwością, którą osoba udzielająca pożyczki znała w momencie podejmowania decyzji. Jest to niezbędne do utworzenia modelu, który będzie pomocny w rozwiązywaniu danego problemu. Gdyby model zawierał cechę wskazującą, czy klient stracił pracę w okresie kredytowania, prawdopodobnie dostarczałby trafne wyniki, ale analityk kredytowy nie mógłby *użyć* tego modelu, ponieważ w momencie podejmowania decyzji o pożyczce, nie wiedziałby, jak ustalić tę cechę. Skąd może wiedzieć, czy klient straci pracę w okresie kredytowania, który jeszcze się nie rozpoczął?

Numer klienta	Wiek	Ocena ryzyka	Własność domu	Typ pojazdu	Rezultat
1	52	420	Własność	Sedan	Niespłacony
2	52	460	Własność	Sedan	Niespłacony
3	64	480	Najem	Sportowy	Spłacony
4	31	580	Najem	Sedan	Niespłacony
5	36	620	Własność	Sportowy	Spłacony
6	29	690	Najem	Pickup	Spłacony
7	23	730	Najem	Sedan	Spłacony
8	27	760	Najem	Pickup	Spłacony
9	43	790	Własność	Pickup	Spłacony

Rysunek 1.6 Zbiór danych spłaty kredytów przez dawnych klientów

Jeśli użyjemy algorytmu uczenia maszynowego do wygenerowania modelu w oparciu o dane, może on wychwycić kilka własności zbioru danych, które można zauważyć także samodzielnie. Po pierwsze, większość osób z oceną ryzyka kredytowego poniżej 600, którym salon udzielił w przeszłości kredytu na zakup samochodu, nie spłaciła go. Gdybyśmy użyli tylko tej własności do podejmowania decyzji, najprawdopodobniej osiągnęlibyśmy przyzwoite rezultaty. Jednak jeśli dokładnie przyjrzymy się danym, możemy zauważyć, że moglibyśmy osiągnąć jeszcze lepszy wynik, decydując, że wszystkie osoby, które otrzymały ocenę ryzyka kredytowego niższą niż 600 *i* kupują samochód typu sedan, prawdopodobnie nie spłacą kredytu. Tego typu wiedza, gdy jest generowana przez algorytm, jest nazywana modelem uczenia maszynowego!

Analityk kredytowy mógłby następnie wdrożyć ten model uczenia maszynowego, po prostu wykorzystując te reguły do przewidywania spłaty za każdym razem, gdy ktoś ubiega się o kredyt. Jeśli kolejnym klientem okazałaby się osoba z oceną ryzyka kredytowego 780, która chce kupić samochód sportowy, jak na rysunku 1.7, należy zaoferować jej najlepszy kredyt, ponieważ ryzyko niespłacenia pożyczki jest niewielkie. Natomiast jeśli klient ma ocenę ryzyka kredytowego 410 i kupuje sedana, z pewnością lepiej przydzielić mu kredyt wysokiego ryzyka. Natomiast dla klientów, którzy mieszczą się między tymi dwoma ekstremami, najlepszym rozwiązaniem byłaby standardowa pożyczka.

Był to jednak uproszczony przykład. Wszystkich klientów w tym przykładzie można z łatwością zaliczyć do przedstawionych kategorii. Oczywiście nie jest to realistyczna sytuacja. W rzeczywistości algorytmy uczenia maszynowego zawierają niedoskonałe dane, których nie da się tak łatwo i jednoznacznie podzielić na grupy. Zbiory danych mogą zawierać dużo więcej obserwacji i algorytmy czasami prowadzą do podjęcia błędnych decyzji. Może się zdarzyć, że kolejny młody klient o przychylnej ocenie ryzyka kredytowego, który kupi w salonie sportowy samochód, straci później pracę i nie spłaci pożyczki. Przedstawiony algorytm skutkowałby nieprawidłową prognozą. Typy błędów popełnianych przez algorytmy omówione zostaną w dalszej części rozdziału.

Rysunek 1.7 Stosowanie modelu uczenia maszynowego

Uczenie nienadzorowane

Techniki *uczenia nienadzorowanego* działają nieco inaczej. Podczas gdy techniki nadzorowane trenują na danych opatrzonych etykietami, techniki nienadzorowane budują modele w oparciu o zbiory danych treningowych bez etykiet (nieoznaczonych). To wpływa na naturę obsługiwanych zbiorów danych i generowanych modułów. Zamiast przypisywać etykiety do danych wejściowych w oparciu o dane historyczne, techniki nienadzorowane umożliwiają odkrywanie wzorców ukrytych w danych.

Jedną z różnic między algorytmami nadzorowanymi a nienadzorowanymi jest to, że algorytmy nadzorowane pomagają w przypisywaniu znanych etykiet do nowych obserwacji, natomiast algorytmy nienadzorowane pomagają w odkrywaniu w zbiorze danych nowych etykiet, czyli sposobów grupowania obserwacji.

Wróćmy do przykładu salonu samochodowego i wyobraźmy sobie, że mamy do dyspozycji zbiór danych klientów i chcemy przygotować kampanię reklamową dla naszego działu usług. Podejrzewamy, że klientów w naszej bazie danych łączą pewne

podobieństwa, które nie są tak oczywiste jak typ kupowanego samochodu i chcielibyśmy odkryć potencjalne sposoby grupowania klientów, aby skierować do nich różne przekazy marketingowe.

Algorytmy uczenia nienadzorowanego dobrze radzą sobie z tego typu otwartymi zadaniami odkrywczymi. Problem opisany na przykładzie salonu samochodowego jest ogólnie nazywany problemem *segmentacji rynku* i dostępnych jest wiele technik uczenia nienadzorowanego zaprojektowanych z myślą o tego typu analizie. W rozdziale 12 pokażemy, w jaki sposób organizacje wykorzystują algorytmy nienadzorowanej *klasteryzacji* do przeprowadzania segmentacji rynku.

Rozważmy kolejny przykład. Wyobraźmy sobie, że zarządzamy sklepem spożywczym i próbujemy ustalić najbardziej optymalne rozmieszczenie towarów na półkach. Wiemy, że klienci często wchodzą do sklepu po to, by kupić typowe podstawowe produkty, takie jak mleko, chleb, wędliny itp. Chcemy zaprojektować sklep tak, aby rozmieścić koło nich asortyment często kupowany pod wpływem impulsu. Jak widać na rysunku 1.8, chcemy umieścić ciastka koło mleka, aby klient, który przyszedł po mleko, zobaczył je i pomyślał „Te ciastka doskonale smakowałyby ze szklanką mleka!”.

Rysunek 1.8 Strategiczne rozmieszczanie produktów w sklepie spożywczym w oparciu o uczenie nienadzorowane

Ustalanie, jakie produkty klienci często kupują razem, to problem dobrze znany w świecie uczenia maszynowego i nazywany problemem *koszyka zakupów*. W rozdziale 11 wyjaśnimy, w jaki sposób analitycy danych rozwiązują problem koszyka zakupów, używając *reguł asocjacyjnych*.

Uwaga Czytelnicy być może słyszeli o trzecim typie algorytmów uczenia maszynowego nazywanych *uczeniem przez wzmacnianie*. Te algorytmy uczą się metodą prób i błędów, podobnie jak dziecko, które poprzez karanie i nagradzanie poznaje panujące w domu reguły. Uczenie poprzez wzmacnianie to interesująca technika, jednak wykracza poza zakres tej książki.

Wybór modelu

W poprzedniej części rozdziału opisane zostały sposoby grupowania algorytmów w oparciu o typy danych, które są wykorzystywane w procesie uczenia. Algorytmy wykorzystujące zbiory danych treningowych z etykietami są nazywane *algorytmami nadzorowanymi*, ponieważ proces uczenia się jest „nadzorowany" przez etykiety, natomiast te wykorzystujące zbiory danych treningowych bez etykiet są nazywane *algorytmami nienadzorowanymi*, ponieważ mogą nauczyć się dowolnych wzorców, które zdołają odkryć, bez „nadzoru". Ten system kategoryzacji opisuje, *w jaki sposób* uczą się algorytmy uczenia maszynowego.

Można również skategoryzować algorytmy w oparciu o to, *czego* się uczą. W tej książce będziemy omawiać trzy główne typy wiedzy, jaką można zdobyć na podstawie danych. Techniki *klasyfikacji* trenują modele, które umożliwiają przewidywanie przynależności do kategorii. Techniki *regresji* umożliwiają przewidywanie wyniku liczbowego. Techniki *uczenia się relacji podobieństwa* pomagają w odkrywaniu podobieństw i różnic między obserwacjami w zbiorze danych.

Techniki klasyfikacji

Techniki klasyfikacji wykorzystują nadzorowane uczenie maszynowe, aby pomóc w przewidywaniu *odpowiedzi kategorialnej*. To oznacza, że wynikiem modelu jest nienumeryczna etykieta lub bardziej formalnie zmienna kategorialna. To oznacza po prostu, że zmienna przyjmuje dyskretne wartości nienumeryczne, zamiast wartości liczbowych. Oto kilka przykładowych zmiennych kategorialnych z pewnymi wartościami, jakie mogą one przyjmować:

- stopień naukowy (brak, licencjat, magister, doktor),
- obywatelstwo (Stany Zjednoczone, Irlandia, Nigeria, Chiny, Australia, Południowa Korea),
- grupa krwi (A+, A-, B+, B-, AB+, AB-, O+, O-),

- członkostwo w partiach politycznych (liberał, republikanin, niezrzeszony),
- status klienta (obecny klient, dawny klient, potencjalny klient).

Na przykład, we wcześniejszej części tego rozdziału opisana została sytuacja, gdy menedżerowie salonu samochodowego chcą przewidywać spłatę kredytu. Jest to przykład problemu klasyfikacji, ponieważ próbujemy przypisać każdego klienta do jednej z dwóch kategorii: kredyt spłacony i kredyt niespłacony.

W realnym świecie napotykamy wiele różnych typów problemów klasyfikacji. Na przykład, gdy próbujemy ustalić, która z trzech ofert promocyjnych będzie najbardziej atrakcyjna dla potencjalnego klienta. Jest to problem klasyfikacji, w którym kategoriami są trzy różne oferty.

Inny przykład to sytuacja, gdy obserwujemy próby logowania w systemie informatycznym i usiłujemy przewidzieć, czy inicjują je osoby będące uprawnionymi użytkownikami czy hakerami, którzy próbują przełamać zabezpieczenia systemu. To również jest problem klasyfikacji, w którym próbujemy przypisać każdą próbę logowania do kategorii „uprawniony użytkownik" lub „haker".

Techniki regresji

Techniki regresji używają technik nadzorowanego uczenia maszynowego do przewidywania *odpowiedzi ciągłych*. W uproszczeniu oznacza to, że danymi wyjściowymi modelu są wartości liczbowe. Zamiast przewidywać przynależność do dyskretnego zbioru kategorii, przewidywana jest wartość zmiennej liczbowej.

Na przykład, doradcy finansowemu szukającemu nowych klientów mogłaby przydać się możliwość wybierania klientów na podstawie ich dochodu. Jeśli doradca dysponuje listą potencjalnych klientów, która niestety nie zawiera jawnie określonego dochodu, może użyć bazy danych dotychczasowych klientów o znanych dochodach do trenowania modelu regresji, który przewiduje dochód potencjalnych klientów. Ten model mógłby wyglądać następująco:

$$Dochód = 5000 + 1000 \cdot wiek + 3000 \cdot lataEdukacjiWyższej$$

Następnie, gdy doradca finansowy znajdzie nowego potencjalnego klienta, może użyć tego wzoru do przewidywania jego dochodu w oparciu o wiek i lata edukacji. Na każdy dodatkowy rok w wieku potencjalnego klienta przewidywany byłby dodatkowy 1000$ dochodu. Natomiast każdy rok edukacji wyższej zwiększałby przewidywany dochód o 3000$.

Modele regresji są dość elastyczne. Można wstawić dowolne wartości wieku oraz edukacji, aby otrzymać przewidywania dotyczące dochodu potencjalnego klienta. Oczywiście, jeśli brakowało dobrych danych treningowych, predykcje mogą być niedokładne. Może się również okazać, że relacji między zmiennymi nie można odzwierciedlić za pomocą prostej techniki liniowej. Na przykład dochód prawdopodobnie

rośnie z wiekiem, ale tylko do pewnego pułapu. Bardziej zaawansowane techniki regresji pozwoliłyby zbudować bardziej złożone modele, które uwzględniają te czynniki. Omówimy je w rozdziale 4.

Techniki uczenia się relacji podobieństwa

Techniki uczenia się relacji podobieństwa wykorzystują algorytmy uczenia maszynowego, by pomóc w identyfikowaniu wspólnych wzorców w danych. Czasami nie wiemy, co dokładnie chcemy odkryć, więc pozwalamy algorytmowi na zbadanie zbioru danych w poszukiwaniu podobieństw, których nie braliśmy jeszcze pod uwagę.

W tym rozdziale zostały już nadmienione dwa sposoby uczenia się relacji podobieństwa. Techniki reguł asocjacyjnych, które omówimy w sposób bardziej szczegółowy w rozdziale 11, umożliwiają rozwiązywanie problemów zbliżonych do problemu koszyka zakupów, czyli ustalania, jakie produkty są często kupowane razem. Techniki klastrowania, które omówimy w sposób bardziej szczegółowy w rozdziale 12, umożliwiają grupowanie obserwacji w klastry w oparciu o ich podobne cechy.

Reguły asocjacyjne i klastrowanie to przykłady nienadzorowanego uczenia się relacji podobieństwa. Można również uczyć relacji podobieństwa w sposób nadzorowany. Na przykład, algorytmy najbliższego sąsiedztwa służą do przypisywania obserwacjom etykiet na podstawie etykiet najbardziej podobnych obserwacji w zbiorze danych treningowych. Dodatkowe informacje przedstawimy w rozdziale 6.

Ocenianie modelu

Przed przejściem do omówienia konkretnych algorytmów uczenia maszynowego warto mieć pewne pojęcie, w jaki sposób oceniana będzie wydajność algorytmów. Ten temat omówimy dużo bardziej szczegółowo w dalszej części książki, teraz jedynie zarysujemy ogólną koncepcję. Analizując poszczególne techniki uczenia maszynowego, będziemy omawiać ocenianie jego wydajności na zbiorze danych. Ponadto rozdział 9 zawiera kompleksowe omówienie oceny wydajności modelu.

Na razie wystarczy mieć świadomość, że różne algorytmy różnie radzą sobie z rozwiązywaniem pewnych problemów. Wybór odpowiedniej techniki zależy od natury zbioru danych i natury algorytmu.

W świecie uczenia nadzorowanego możemy ocenić wydajność algorytmu w oparciu o liczbę i/lub wielkość popełnianych przez niego błędów. W przypadku problemów klasyfikacji często sprawdzamy, jak często procentowo algorytm nieprawidłowo przewiduje kategorię, czyli tzw. *współczynnik błędnej klasyfikacji*. Analogicznie możemy sprawdzać procent predykcji, które były prawidłowe, czyli tzw. *dokładność* algorytmu. W przypadku problemów regresji często sprawdzamy różnicę między wartościami przewidzianymi przez algorytm a rzeczywistymi.

Uwaga Mówienie o tego typu ocenie ma sens tylko w kontekście technik uczenia nadzorowanego, gdzie istnieje prawidłowa odpowiedź. W przypadku uczenia nienadzorowanego wykrywamy wzorce bez jakichkolwiek obiektywnych wytycznych, dlatego nie ma z góry ustalonej „prawidłowej" lub „nieprawidłowej" odpowiedzi, na podstawie której można byłoby mierzyć wydajność. W związku z tym wydajność algorytmu uczenia nienadzorowanego zależy od przydatności dostarczanej przez niego wiedzy.

Błędy klasyfikacji

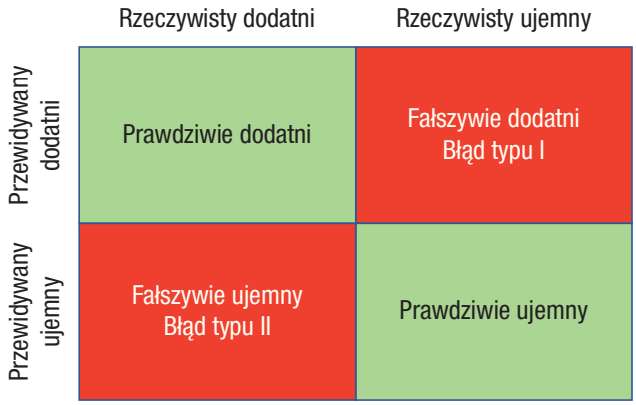
Wiele problemów klasyfikacji wiąże się z przewidywaniem wartości binarnej określającej, czy obserwacja należy do danej klasy. Przypadki, gdy obserwacja należy do klasy, nazywamy przypadkami *dodatnimi*, a te, gdy obserwacja nie należy do klasy, przypadkami *ujemnymi*.

Wyobraźmy sobie na przykład, że budujemy model zaprojektowany z myślą o przewidywaniu, czy dana osoba nie toleruje laktozy, co utrudnia jej trawienie produktów mlecznych. Ten model mógłby zawierać czynniki demograficzne, genetyczne i środowiskowe, które mogą przyczyniać się do nietolerowania laktozy. Następnie w zależności od tych atrybutów model przewiduje, czy poszczególne osoby tolerują czy nie tolerują laktozy. Osoby, dla których przewidywana jest nietolerancja laktozy, są przewidywanymi przypadkami dodatnimi, a te, dla których przewiduje się brak nietolerancji laktozy (czyli w uproszczeniu przewiduje się tolerancję laktozy) są przewidywanymi przypadkami ujemnymi. Te przewidywane wartości pochodzą z modelu uczenia maszynowego.

Jednak w rzeczywistości istnieje jakaś prawda. Niezależnie od przewidywań modelu, każda osoba albo toleruje laktozę, albo nie. Te dane rzeczywiste określają, czy osoba reprezentuje rzeczywisty przypadek dodatni czy ujemny. Gdy przewidywana wartość obserwacji różni się od wartości rzeczywistej dla tej samej obserwacji, pojawia się *błąd*. W problemie klasyfikacji mogą występować dwa różne typy błędów.

- *Wyniki fałszywie dodatnie* pojawiają się, gdy model przewiduje dla obserwacji wynik dodatni, podczas gdy w rzeczywistości jest on ujemny. Na przykład, gdy model identyfikuje kogoś jako najprawdopodobniej nietolerującego laktozy, mimo iż w rzeczywistości toleruje on laktozę, jest to wynik fałszywie dodatni. Wyniki fałszywie dodatnie są również nazywane *błędami typu I*.
- *Wyniki fałszywie ujemne* pojawiają się, gdy model przewiduje dla obserwacji wynik ujemny, podczas gdy w rzeczywistości jest on dodatni. W naszym modelu nietolerancji laktozy, jeśli model przewiduje, że pewna osoba toleruje laktozę, podczas gdy w rzeczywistości nie toleruje ona laktozy, jest to wynik fałszywie ujemny. Wyniki fałszywie ujemne są również nazywane *błędami typu II*.

Analogicznie można określić prawidłowo przewidziane obserwacje jako wyniki *prawdziwie dodatnie* lub *prawdziwie ujemne*, w zależności od ich etykiety. Rysunek 1.9 przedstawia typy błędów w postaci tabeli.



Rysunek 1.9 Typy błędów

Oczywiście całkowita liczba wyników fałszywie dodatnich i fałszywie ujemnych zależy od liczby dokonywanych predykcji. Zamiast opierać się na pomiarach bazujących na liczebności, mierzymy, jak często procentowo pojawiają się tego typu błędy. Na przykład *współczynnik fałszywie dodatnich* (FPR, false positive rate) to odsetek ujemnych przypadków, które zostały nieprawidłowo zidentyfikowane jako dodatnie. Możemy obliczyć ten odsetek, dzieląc liczbę wyników fałszywie dodatnich (FP, false positive) przez sumę liczby wyników fałszywie dodatnich i liczby wyników prawdziwie ujemnych (TN, true negative), jak w poniższym wzorze:

$$FPR = \frac{FP}{FP + TN}$$

Analogicznie możemy obliczyć *współczynnik fałszywie ujemnych* (FNR, false negative rate) w następujący sposób:

$$FNR = \frac{FN}{FN + TP}$$

Nie ma jasnej reguły wskazującej, czy jeden typ błędu jest lepszy, czy gorszy od drugiego. Zależy to przede wszystkim od typu rozwiązywanego problemu.

Wyobraźmy sobie na przykład, że używamy algorytmu uczenia maszynowego do klasyfikowania długiej listy potencjalnych klientów jako osób, które zakupią nasz produkt (przypadki dodatnie) lub które go nie zakupią (przypadki ujemne). Wydamy pieniądze tylko na wysłanie listów do kontaktów oznaczonych przez algorytm jako dodatnie.

W przypadku wyniku fałszywie dodatniego wysyłamy broszurę do klienta, który nie kupi naszego produktu. Niepotrzebnie wydaliśmy więc pieniądze na drukowanie i przesłanie tej broszury. W przypadku wyniku fałszywie ujemnego nie wysyłamy listu do klienta, który odpowiedziałby na niego. Tracimy okazję, by sprzedać produkt temu klientowi. Która sytuacja jest gorsza? To zależy od kosztów przesyłki, potencjalnego zysku z transakcji sprzedaży i innych czynników.

Z drugiej strony rozważmy użycie modelu uczenia maszynowego do prowadzenia badań przesiewowych pod kątem nowotworu i kierowania pacjentów z wynikiem dodatnim na dalsze bardziej inwazyjne testy. W przypadku wyniku fałszywie ujemnego pacjent, który być może ma raka, nie zostaje skierowany na dodatkowe badania, co może prowadzić do nieleczenia aktywnej choroby. Co jest oczywiście sytuacją wysoce niepożądaną.

Jednak wyniki fałszywie dodatnie również nie są pozbawione negatywnych konsekwencji. Pacjent, u którego fałszywie zidentyfikowano podejrzenie nowotworu, jest niepotrzebnie poddawany potencjalnie bolesnym i kosztownym badaniom, co zużywa zasoby, które można byłoby poświęcić na innego pacjenta. Ponadto jest narażony na szkody emocjonalne, gdy czeka na wynik nowych testów.

Ocena problemów uczenia maszynowego to skomplikowana kwestia i musi być przeprowadzana z uwzględnieniem domeny problemu. Analitycy danych i eksperci domenowi, a w niektórych przypadkach także etycy, muszą wspólnie ocenić modele pod kątem korzyści i kosztu każdego z typów błędów.

Błędy regresji

Błędy, z którymi mamy do czynienia w przypadku problemów regresji, są nieco inne, ponieważ predykcje mają odmienny charakter. Gdy przypisujemy obserwacjom etykiety, nasze predykcje mogą być albo prawidłowe, albo nieprawidłowe. Gdy oznaczymy łagodny nowotwór jako złośliwy, jest to oczywisty błąd. Jednak w przypadku problemów regresji przewidujemy wartość liczbową.

Powróćmy do problemu przewidywania dochodów omawianego we wcześniejszej części rozdziału. Jeśli pewna osoba zarabia 45.000$ rocznie i nasz algorytm trafnie przewiduje dokładnie 45.000$, jest to wyraźnie prawidłowa predykcja. Gdyby algorytm przewidział dochód 0$ lub 10.000.000$, prawie wszyscy zgodziliby się, że te predykcje są obiektywnie nieprawidłowe. Ale co z predykcjami 45.001$, 45.500$, 46.000$ czy 50.000$? Czy wszystkie są nieprawidłowe? A może niektóre z nich lub nawet wszystkie są wystarczającym przybliżeniem?

Bardziej sensowne jest ocenianie algorytmów regresji w oparciu o wielkość błędu predykcji. Określamy go, mierząc odległość między wartością przewidywaną a faktyczną. Weźmy pod uwagę przykładowy zbiór danych przedstawiony na rysunku 1.10.

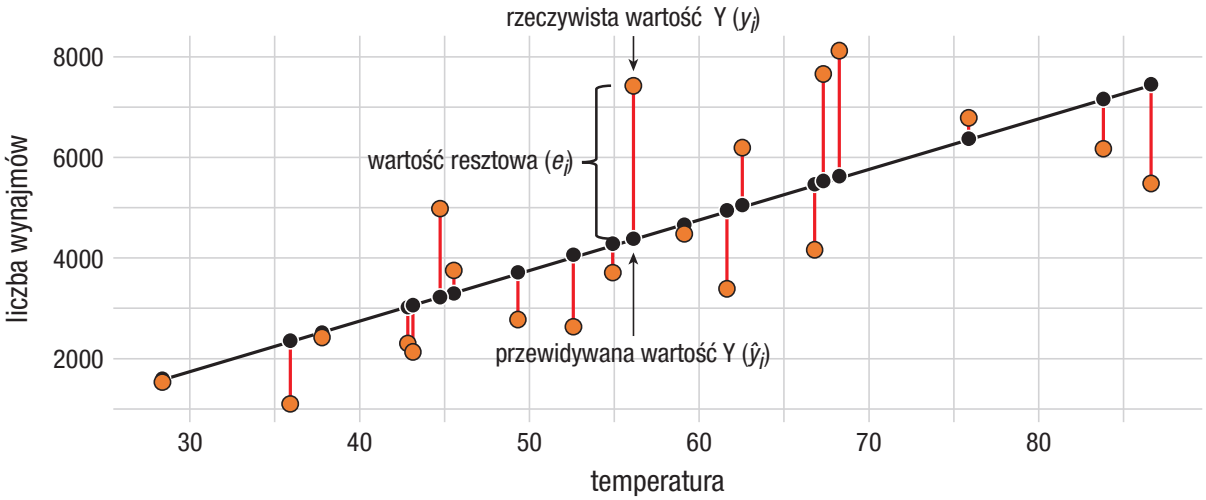
W tym zbiorze danych próbujemy przewidzieć liczbę rowerów, jakie zostaną wynajęte danego dnia w oparciu o średnią temperaturę w tym dniu. Liczba wynajętych rowerów znajduje się na osi y, natomiast temperatura na osi x. Czarna linia to linia regresji, która wskazuje, że spodziewamy się wzrostu liczby wynajmowanych rowerów wraz ze wzrostem temperatury. Ta czarna linia to nasz model, natomiast czarne kropki to przewidywania dla konkretnych temperatur wzdłuż tej linii.

Pomarańczowe kropki reprezentują rzeczywiste dane zebrane przez firmę wynajmującą rowery. Są to „prawidłowe" dane. Czerwone linie między przewidywanymi

i prawdziwymi wartościami to wielkość błędu, którą nazywamy *wartością resztową*. Im dłuższa linia, tym gorzej algorytm sprawdzał się dla tych danych.

Nie można po prostu zsumować wartości resztowych, ponieważ niektóre z nich są ujemne, więc neutralizowałyby wartości dodatnie. Dlatego każda z wartości resztowych jest podnoszona do kwadratu i dopiero później sumowana z pozostałymi, dzięki czemu powstaje miara wydajności o nazwie *resztowa suma kwadratów*.

Do koncepcji błędu resztowego, jak i tego konkretnego zbioru danych wypożyczalni rowerów, wrócimy w rozdziale 4.



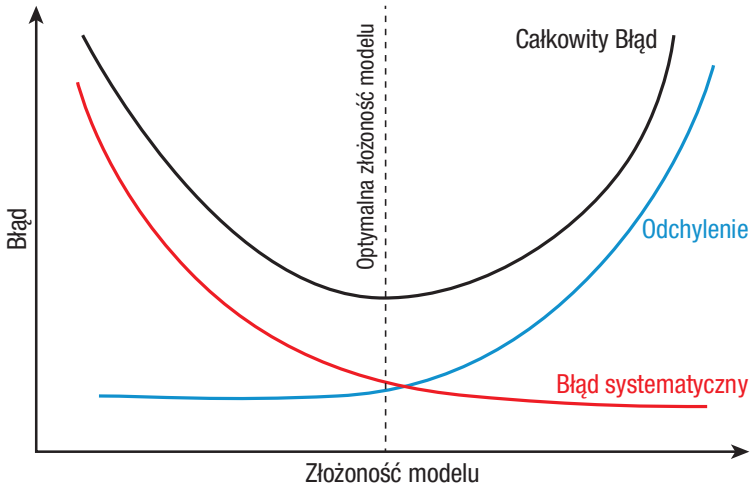
Rysunek 1.10 Błąd resztowy

Typy błędów

Modele uczenia maszynowego budowane dla wszystkich problemów, poza tymi najprostszymi, będą zawierać jakieś błędy predykcji. Można wyróżnić trzy różne rodzaje błędów:

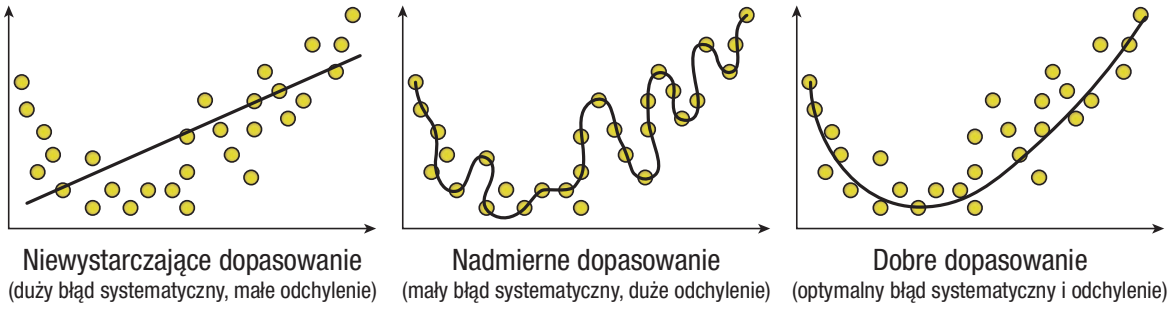
- *Błąd systematyczny (bias)* w świecie uczenia maszynowego to taki typ błędu, który ściśle wiąże się z wyborem modelu uczenia maszynowego. Gdy wybranego modelu nie można odpowiednio dopasować do zbioru danych, powstaje błąd systematyczny.
- *Odchylenie (variance)* to typ błędu, który pojawia się, gdy zbiór danych wykorzystywany do trenowania modelu uczenia maszynowego nie reprezentuje całego spektrum możliwych danych.
- *Nieredukowalny błąd* nazywany też szumem pojawia się niezależnie od zastosowanego algorytmu uczenia maszynowego bądź zbioru danych treningowych. Ten błąd jest ściśle powiązany z problemem, który jest rozwiązywany.

Próbując rozwiązać konkretny problem uczenia maszynowego, niewiele możemy zrobić z błędem nieredukowalnym, dlatego lepiej skoncentrować się na dwóch pozostałych typach błędów: błędzie systematycznym i odchyleniu. Zasadniczo algorytm, który charakteryzuje się dużym odchyleniem, ma niewielki błąd systematyczny, natomiast algorytm o małym odchyleniu ma duży błąd systematyczny, jak widać na rysunku 1.11. Błędy systematyczne i odchylenia są powiązanymi ze sobą cechami charakterystycznymi modelu. Gdy modyfikujemy model, by skorygować jeden z nich, robimy to kosztem drugiego. Celem jest osiągnięcie równowagi między nimi.



Rysunek 1.11 Kompromis między błędem systematycznym a odchyleniem

Gdy mamy do czynienia z dużym błędem systematycznym i małym odchyleniem, mówimy, że model jest niewystarczająco dopasowany do danych (ang. underfitting). Przyjrzyjmy się kilku przykładom, które ilustrują taką sytuację. Rysunek 1.12 przedstawi kilka prób użycia funkcji dwóch zmiennych do przewidzenia wartości trzeciej zmiennej.



Rysunek 1.12 Niewystraczające, nadmierne i optymalne dopasowanie

Wykres po lewej na rysunku 1.12 przedstawia model liniowy, który jest niewystarczająco dopasowany do danych. Nasze dane są rozproszone wzdłuż krzywej, natomiast my wybraliśmy linię prostą (model liniowy), ograniczając możliwość dopasowania modelu do danych. Nie można narysować linii prostej w sposób dobrze dopasowany

do tego zbioru danych. Z tego powodu większość błędów w naszym rozwiązaniu będzie wynikała z wyboru modelu i nasz zbiór danych ujawni duży błąd systematyczny.

Środkowy wykres na rysunku 1.12 ilustruje problem nadmiernego dopasowania (ang. overfitting), który występuje, gdy mamy model o niskim błędzie systematycznym, ale wysokim odchyleniu. W tym przypadku nasz model *zbyt* dobrze pasuje do zbioru danych treningowych. Jest odpowiednikiem przygotowywania się do konkretnego testu (zbioru danych treningowych), zamiast uczenia się ogólnej techniki rozwiązania problemów. Istnieje duże ryzyko, że gdy ten model zostanie zastosowany na innym zbiorze danych, nie będzie działał zbyt dobrze. Zamiast zdobywać ogólną wiedzę, nauczyliśmy się odpowiedzi z poprzednich testów. I gdy zobaczymy nowy test, nie będziemy dysponować wiedzą potrzebną do wywnioskowania odpowiedzi.

Pożądaną równowagą jest model, który optymalizuje zarówno błąd systematyczny, jak i odchylenie, taki jak wykres po prawej stronie na rysunku 1.12. Ten model odzwierciedla krzywiznę rozmieszczenia danych, ale nie podąża za poszczególnymi punktami w zbiorze danych treningowych. Pasuje do zbioru danych dużo bardziej niż model niewystarczająco dopasowany, ale jednocześnie nie śledzi pojedynczych punktów w zbiorze danych treningowych jak model nadmiernie dopasowany.

Partycjonowanie zbiorów danych

Oceniając model uczenia maszynowego, można przeciwdziałać błędom odchylenia, używając techniki walidacji, która zaznajamia model z danymi innymi niż te, które zostały użyte do jego utworzenia. To podejście pomaga radzić sobie z problemem nadmiernego dopasowania. Wróćmy do nadmiernie dopasowanego modelu z rysunku 1.12. Gdybyśmy użyli zbioru danych treningowych do oceny tego modelu, odkrylibyśmy, że osiągał on rewelacyjne rezultaty, ponieważ został zbudowany tak, by dobrze radzić sobie z tym konkretnym zbiorem danych. Jednak gdybyśmy użyli nowego zbioru danych do oceny tego modelu, najprawdopodobniej osiągałby on bardzo słabe rezultaty.

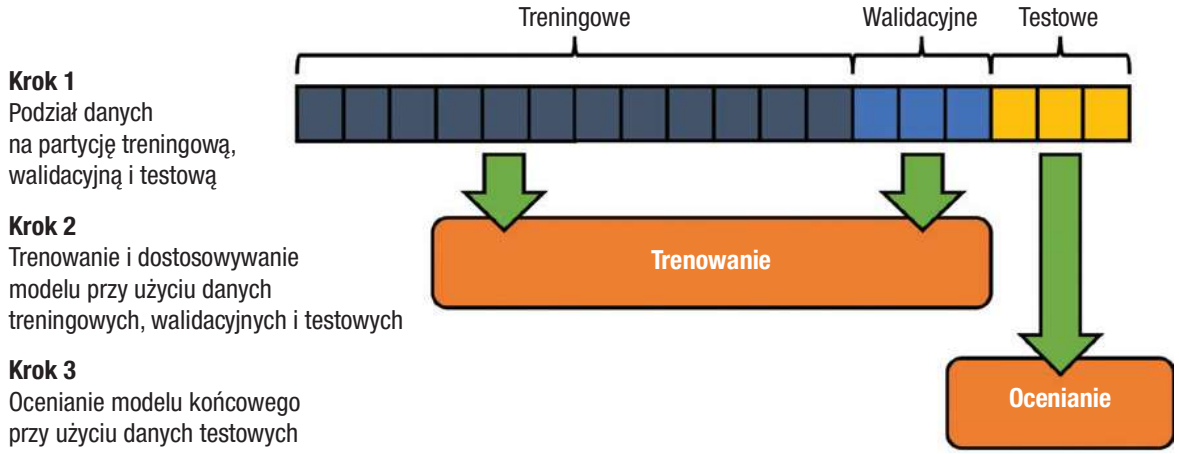
Można zbadać ten problem, używając *zbioru danych testowych* do ocenienia wydajności tego modelu. W początkowej fazie rozwijania modelu rezerwowany jest zbiór danych testowych, który posłuży do oceniania modelu. Nie jest on wykorzystywany w procesie trenowania, dlatego model nie może nadmiernie się do niego dopasować. Jeśli opracujemy ogólny model, który nie jest nadmiernie dopasowany do zbioru danych treningowych, będzie on osiągał dobre rezultaty dla zbioru danych testowych. Natomiast jeśli model jest nadmiernie dopasowany do zbioru danych treningowych, nie będzie dobrze radził sobie ze zbiorem danych testowych.

Czasami potrzebujemy również osobnego zbioru danych stanowiącego dodatkową pomoc w procesie budowania modelu. Tego typu zbiory danych, nazywane *zbiorami danych walidacyjnych*, pomagają w rozwijaniu modelu w sposób iteracyjny, gdzie w każdej kolejnej iteracji parametry modelu są dostrajane aż do znalezienia rozwiązania, które dobrze radzi sobie ze zbiorem danych walidacyjnych. Choć może się pojawić

pokusa, by użyć zbioru danych testowych w roli zbioru danych walidacyjnych, to podejście pociąga za sobą ryzyko nadmiernego dostosowania, dlatego trzeba użyć trzeciego zbioru danych.

Metoda wydzielania

Najprostszym sposobem uzyskiwania testowego i walidacyjnego zbioru danych jest *metoda wydzielania*. To podejście, zilustrowane na rysunku 1.13, polega na wydzieleniu na początku procesu budowania modelu porcji pierwotnego zbioru danych na potrzeby walidacji i testowania. Zbiór danych walidacyjnych pomoże w rozwijaniu modelu, a następnie zbiór danych testowych posłuży do oceniania wydajności końcowego modelu.



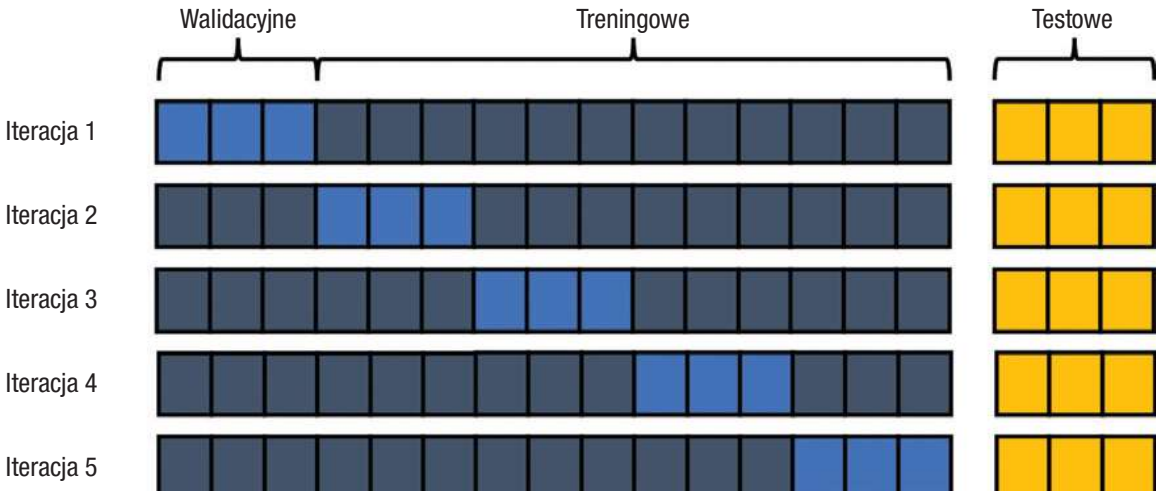
Rysunek 1.13 Metoda podziału

Metody walidacji krzyżowej

Istnieją bardziej zaawansowane metody tworzenia zbiorów danych walidacyjnych, które wielokrotnie pobierają próbki danych w iteracyjnym procesie rozwijania modelu. Te metody, nazywane technikami *walidacji krzyżowej*, są szczególnie przydatne, gdy zbiory danych są niewielkie i niepożądane byłoby rezerwowanie porcji zbioru danych na potrzeby walidacji.

Rysunek 1.14 przedstawia przykład walidacji krzyżowej. W tej metodzie porcja zbioru danych nadal odkładana jest na potrzeby testów, ale w każdej iteracji procesu rozwijania modelu do walidacji wykorzystywana jest inna porcja zbioru danych treningowych.

Nic nie szkodzi, jeśli wydaje się to skomplikowane. Metody podziału i walidacji krzyżowej zostaną omówione w sposób bardziej szczegółowy w rozdziale 9. Na razie wystarczy ogólne zaznajomienie się z tymi technikami.



Rysunek 1.14 Metoda walidacji krzyżowej

Ćwiczenia

1. Rozważ następujące problemy uczenia maszynowego. Czy dany problem lepiej potraktować jako problem klasyfikacji czy regresji? Przedstaw uzasadnienie dla swojej odpowiedzi.
   a. Przewidywanie liczby ryb złowionych w czasie komercyjnego rejsu połowowego.
   b. Identyfikowanie osób, które mogą być zainteresowane nową technologią.
   c. Przewidywanie cen najmu rowerów na podstawie danych o pogodzie i ludności.
   d. Przewidywanie, jaką kampanię promocyjną najlepiej skierować do danej osoby.
2. Opracowałeś algorytm uczenia maszynowego, który ocenia ryzyko, że dany pacjent będzie miał zawał serca (zdarzenie dodatnie) w oparciu o różne kryteria diagnostyczne. W jaki sposób opisałbyś każde z następujących zdarzeń?
   a. Model identyfikuje pacjenta jako osobę o wysokim ryzyku zawału i pacjent dostaje zawału.
   b. Model identyfikuje pacjenta jako osobę o wysokim ryzyku zawału i pacjent nie dostaje zawału.
   c. Model identyfikuje pacjenta jako osobę o niskim ryzyku zawału i pacjent nie dostaje zawału.
   d. Model identyfikuje pacjenta jako osobę o niskim ryzyku zawału i pacjent dostaje zawału.

ROZDZIAŁ

2

Wprowadzenie do języka R i RStudio

Uczenie maszynowe leży na przecięciu świata statystyki i rozwoju oprogramowania. W niniejszej książce koncentrujemy się na technikach statystycznych pozwalających czerpać korzyści z wiedzy ukrytej w danych. W tym rozdziale opiszemy narzędzia informatyczne, które będą potrzebne do implementowania opisywanych technik. Wybraliśmy do tego celu język programowania R, dlatego przedstawimy wprowadzenie do podstawowych elementów języka R, które będziemy wielokrotnie wykorzystywać w pozostałej części książki.

Z tego rozdziału można dowiedzieć się:

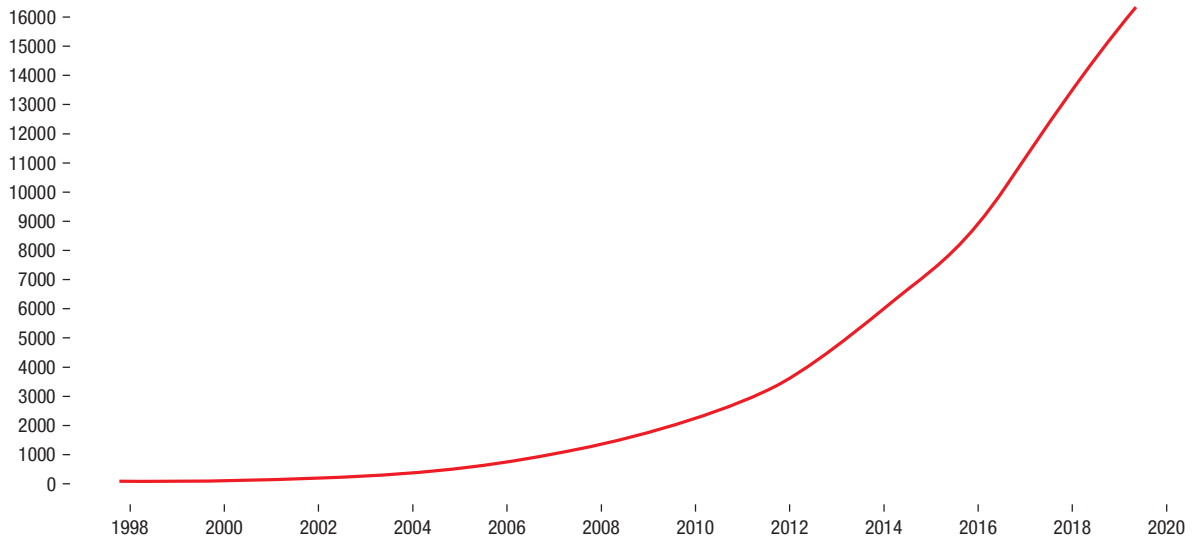
- Jaką rolę język programowania R odgrywa w świecie analizy danych.
- W jaki sposób zintegrowane środowisko deweloperskie (IDE) RStudio ułatwia programowanie w języku R.
- Jak używać pakietów do redystrybucji i wielkokrotnego użycia kodu R.
- Jak pisać, zapisywać i wykonywać własne podstawowe skrypty R.
- Do czego służą różne typy danych R.

Witamy w świecie R

Język programowania R powstał w 1992 roku jako próba utworzenia specjalistycznego języka do celów statystycznych. Ponad dwadzieścia lat później stał się jednym z najpopularniejszych języków wykorzystywanych przez statystyków, analityków danych i analityków biznesowych na całym świecie.

Język programowania R szybko zyskał na popularności z kilku powodów. Po pierwsze, jest dostępnym dla wszystkich darmowym językiem typu open source opracowanym przez społeczność zaangażowanych programistów. To podejście zerwało z tradycją opierania narzędzi analitycznych na zastrzeżonym, komercyjnym oprogramowaniu, które przekraczało możliwości finansowe wielu osób i organizacji.

Popularność języka R rośnie również dlatego, że jest on powszechnie stosowany przez twórców metod uczenia maszynowego. Obecnie prawie każda nowo tworzona technika uczenia maszynowego zostaje szybko udostępniona użytkownikom języka R w postaci *pakietu redystrybucyjnego*, oferowanego w postaci open source w Comprehensive R Archive Network (CRAN), czyli międzynarodowym repozytorium popularnego kodu R. Rysunek 2.1 przedstawia wzrost liczby pakietów dostępnych w repozytorium CRAN na przestrzeni lat. Jak widać, w ostatniej dekadzie liczba ta znacznie wzrosła.



Rysunek 2.1 Wzrost liczby pakietów CRAN na przestrzeni czasu

Warto mieć również świadomość, że R jest *językiem interpretowanym*, a nie kompilowanym. W języku interpretowanym pisany przez programistę kod jest przechowywany w dokumencie nazywanym *skryptem* i ten skrypt jest kodem bezpośrednio wykonywanym przez system przetwarzający kod. W języku kompilowanym kod źródłowy pisany przez programistę jest przetwarzany przez specjalny program nazywany *kompilatorem*, który konwertuje kod źródłowy do wykonalnego języka maszynowego.

Fakt, iż R jest językiem interpretowanym, oznacza również, że można wykonywać polecenia R bezpośrednio, aby od razu zobaczyć ich wynik. Na przykład, można wykonać następujące proste polecenie, by dodać 1 plus 1:

```
> 1+1
[1] 2
```

To spowoduje, że interpreter R natychmiast odpowie, wyświetlając wynik 2.

Komponenty języka R i RStudio

Środowisko robocze tej książki składa się z dwóch głównych komponentów: języka programowania R oraz zintegrowanego środowiska deweloperskiego RStudio. R jest językiem typu open source, natomiast RStudio jest komercyjnym produktem zaprojektowanym po to, by ułatwić korzystanie z języka R.

Język R

Język R jest językiem typu open source dostępnym do pobrania za darmo ze strony projektu R o adresie https://www.r-project.org. W czasie pisania tej książki aktualną wersją języka R była wersja 3.6.0, o nazwie kodowej „Planting of a Tree" (Sadzenie drzewa). R jest zasadniczo napisany tak, by zapewniać zgodność z poprzednimi wersjami, dlatego osoby wykorzystujące nowszą wersję języka R nie powinny mieć problemów z wykonywaniem kodu przedstawianego w tej książce.

Uwaga Poszczególnym wydaniom języka R przypisywane są dość ciekawe nazwy kodowe, takie jak „Great Truth" (Wielka prawda), „Roasted Marshmallows" (Pianki z ogniska), „Wooden Christmas-Tree" (Drewniana choinka) czy „You Stupid Darkness" (Ty głupia ciemności). Wszystkie one odwołują się do serii komiksów Charlesa Schultza zatytułowanej *Peanuts*.

Nadszedł czas, by zainstalować na swoim komputerze najnowszą wersję języka R (o ile nie jest ona już zainstalowana). Wystarczy wejść na stronę główną projektu R, kliknąć łącze CRAN i wybrać najbliżej zlokalizowany serwer CRAN. Następnie wyświetlona zostanie witryna CRAN podobna do tej przedstawionej na rysunku 2.2. Należy wybrać łącze pobierania dla swojego systemu operacyjnego i uruchomić program instalacyjny, gdy pobieranie się zakończy.