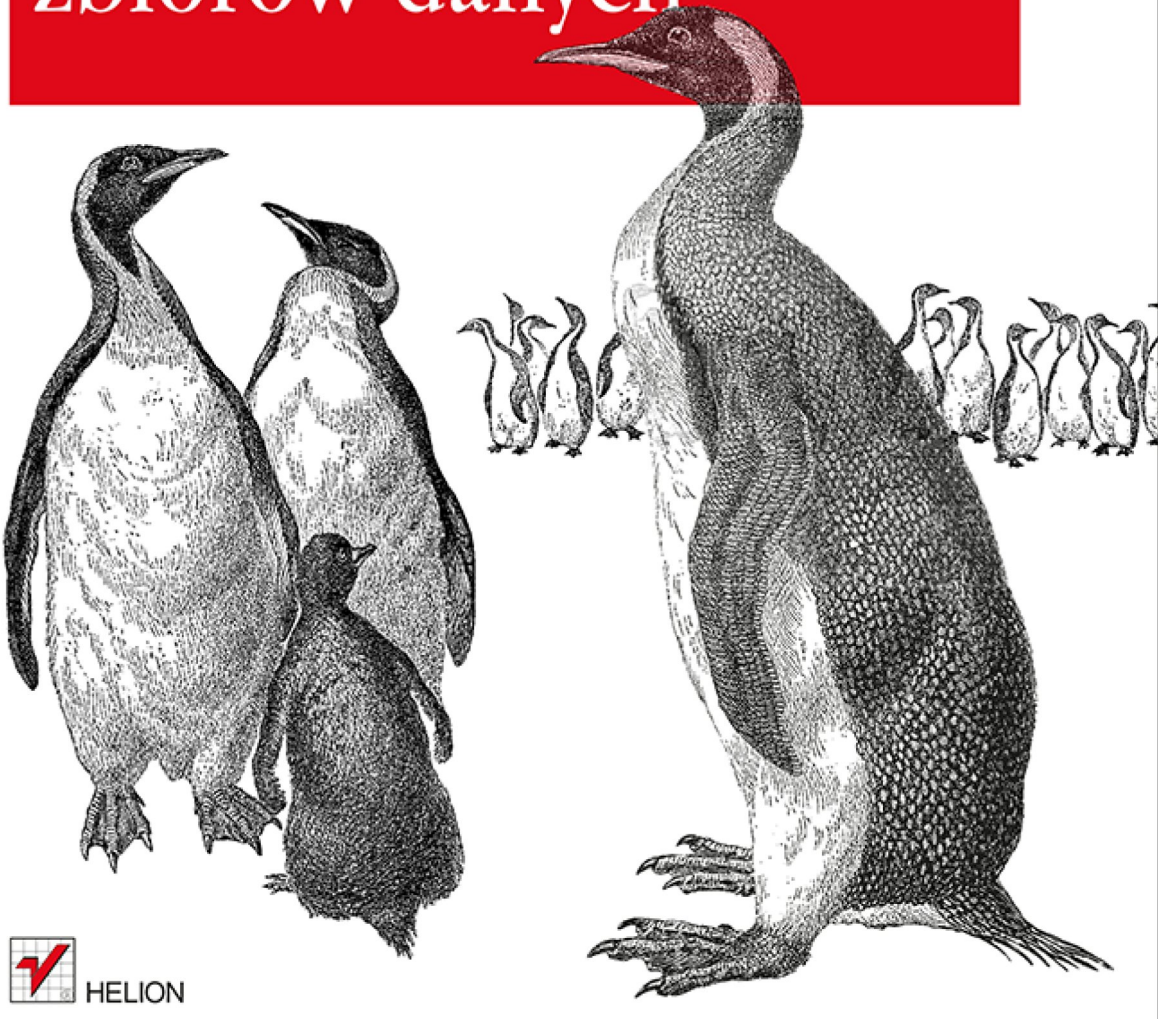
Wykorzystaj dane z sieci do własnych potrzeb!

Nowe usługi 2.0

Przewodnik po analizie zbiorów danych



HELION

O'REILLY®

Toby Segaran

Pochwały dotyczące książki

Każdego roku recenzuję kilka książek. Oczywiście w trakcie pracy czytam dość sporą ich liczbę. Muszę przyznać, że do tej pory lektura żadnej publikacji nie sprawiła mi tyle przyjemności co robocza wersja tej książki. Brawo! Nic lepszego nie przychodzi mi na myśl niż ta książka w przypadku programisty, który dopiero zaczyna przygodę z opisanymi w niej algorytmami i metodami. Sam (jako stary »wyjadacz« od sztucznej inteligencji) sięgnąłbym po nią w pierwszej kolejności, żeby odświeżyć znajomość szczegółów.

— Dan Russell, główny specjalista ds. technologii, firma *Google*

W książce Toby'ego w znakomity sposób dokonano rozbicia złożonego zagadnienia dotyczącego algorytmów uczenia maszynowego na praktyczne i łatwe do zrozumienia przykłady, które mogą być bezpośrednio używane do analizowania interakcji społecznościowej w obecnym Internecie. Jeśli ta książka trafiłaby w moje ręce dwa lata wcześniej, zaoszczędziłbym cenny czas, gdy podążałem okrężnymi ścieżkami.

— Tim Wolters, szef ds. technologii, firma Collective Intellect

Książka stanowi wybitne osiągnięcie pod względem udostępniania obszernej kolekcji komputerowych metod służących do kojarzenia ogromnych ilości danych. Dokładniej rzecz biorąc: techniki te są w książce prezentowane w kontekście Internetu, powodując znajdowanie wartości w przypadku wysp danych, które w innym razie pozostałyby odizolowane. Jeśli tworzysz kod do zastosowań internetowych, ta książka jest dla Ciebie nieodzowna.

— Paul Tyma, starszy inżynier oprogramowania, firma *Google*

Nowe usługi 2.0. Przewodnik po analizie zbiorów danych

Toby Segaran

O'REILLY™

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

Tytuł oryginału: Programming Collective Intelligence: Building Smart Web 2.0 Applications

Tłumaczenie: Piotr Pilch

ISBN: 978-83-246-9298-9

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/noweus.zip

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

Printed in Poland.

Spis treści

Słowo wstępne

Gdy w czasopiśmie „Time" tytuł osobowości roku 2006 nadano „Użytkownikom", potwierdziła się idea, że technologia Web 2.0 ma związek z „treścią generowaną przez użytkowników", a ponadto, że serwisy *Wikipedia*, *YouTube* i *MySpace* są „sercem" rewolucji związanej z Web 2.0. Prawda jest jednak znacznie bardziej złożona. Treść, jaką użytkownicy mogą umieszczać wyłącznie w serwisach Web 2.0, stanowi niewielką część tego, co jest ogólnie widoczne. Osiemdziesiąt procent tego, co ma znaczenie, znajduje się w ukryciu w postaci „ciemnej materii" niejawnie wprowadzanych danych.

W wielu przypadkach początek rewolucji Web 2.0 określa się jako moment pojawienia się algorytmu PageRank wynalezionego przez firmę *Google*. Dzięki niemu uświadomiono sobie, że każdy odnośnik na stronie WWW ma ukryte znaczenie, a mianowicie stanowi on głos dotyczący ważności witryny. Zrozumienie tych głosów oraz relatywnego znaczenia witryn, których one dotyczyły, pozwoliło uzyskać lepsze wyniki wyszukiwania niż w przypadku analizowania wyłącznie samych stron internetowych. Okazało się to przełomem, który umożliwił firmie *Google* zostanie najważniejszą spółką technologiczną nowego wieku. Algorytm PageRank jest obecnie jednym z setek niejawnych czynników, które mają wpływ na to, jakie wyniki wyszukiwania zostaną wyróżnione przez wyszukiwarkę *Google*.

Choć nikt nie scharakteryzowałby firmy *Google* jako spółki „treści generowanej przez użytkowników", jest to oczywiste w odniesieniu do sedna technologii Web 2.0. Z tego właśnie powodu wolę używać frazy „wykorzystanie inteligencji zbiorowej" w roli probierza rewolucji. Odnośnik to treść wygenerowana przez użytkowników, ale algorytm PageRank jest techniką służącą do „wydobywania inteligencji" z tej treści. Tak też jest w przypadku algorytmu „zainteresowania" serwisu *Flickr*, funkcji „osoby, które kupiły ten produkt, nabyły również…" serwisu *Amazon*, algorytmów serwisu *Last.fm* zapewniających „radio z podobnymi artystami", systemu reputacji serwisu *eBay* i serwisu *AdSense* firmy *Google*.

Zdefiniowałem technologię Web 2.0 jako „projekt systemów, które wykorzystują efekty sieci w celu spowodowania, że będzie ich używać jeszcze większa liczba osób". Pierwszym krokiem jest nakłonienie użytkowników do uczestnictwa. Drugim krokiem jest zdobycie wiedzy dzięki tym użytkownikom i dostosowanie witryny na podstawie tego, na co zwracają uwagę, a na co nie.

W książce Toby Segaran omawia algorytmy i techniki służące do wyodrębniania znaczenia z danych, w tym z danych użytkowników. Książka jest zestawem narzędziowym programisty korzystającego z technologii Web 2.0. Nie wystarczy już tylko wiedzieć, jak zbudować

witrynę internetową z zapleczem w postaci bazy danych. Jeśli chcesz odnieść sukces, musisz dowiedzieć się, jak eksplorować dane dodawane przez użytkowników, zarówno jawnie, jak i jako efekt uboczny ich aktywności w witrynie.

Choć powstało wiele książek o technologii Web 2.0 odkąd po raz pierwszy termin ten pojawił się w 2004 r., pod wieloma względami książka Toby'ego to pierwszy praktyczny przewodnik po programowaniu aplikacji Web 2.0.

— Tim O'Reilly

Przedmowa

Zwiększająca się liczba osób korzystających z Internetu, świadomie lub przypadkowo, spowodowała wygenerowanie ogromnego zbioru danych, który zapewnia nam miliony potencjalnych analiz opinii użytkowników, marketingu, osobistych preferencji i zachowań ludzkich w ogólności. Książka stanowi wprowadzenie do rozwijającej się dziedziny określanej mianem *inteligencji zbiorowej*. Omówiono w niej metody uzyskiwania interesujących zbiorów danych z wielu witryn internetowych, o których prawdopodobnie słyszałeś, sposoby gromadzenia danych od użytkowników Twoich aplikacji oraz wiele różnych metod pozwalających przeanalizować i zrozumieć dane po ich znalezieniu.

Celem książki jest wykroczenie poza proste aplikacje z zapleczem w postaci bazy danych, a także nauczenie pisania bardziej inteligentnych programów, które wykorzystają informacje codziennie gromadzone przez Ciebie i inne osoby.

Wymagania wstępne

Przykłady z kodami wykorzystane w książce napisano w języku Python. Choć znajomość sztuki programowania w tym języku będzie pomocna, zamieściłem objaśnienia wszystkich algorytmów. Dzięki temu programiści, którzy używają innych języków, mogą poradzić sobie z tymi kodami. Kod Python będzie szczególnie łatwy do opanowania przez osoby, które znają takie języki wysokiego poziomu, jak Ruby lub Perl. Książka nie ma pełnić funkcji podręcznika do nauki programowania, dlatego ważna jest umiejętność pisania kodu w stopniu wystarczającym do zrozumienia podstawowych zagadnień. Jeśli odpowiednio opanowałeś rekurencję i podstawy programowania funkcyjnego, materiał zawarty w książce okaże się jeszcze łatwiejszy.

W przypadku tej książki nie przyjęto, że dysponujesz już jakąkolwiek wiedzą z zakresu analizy danych, uczenia maszynowego lub statystyki. Choć podjąłem się próby objaśnienia pojęć matematycznych w jak najprostszy sposób, w zrozumieniu algorytmów pomocna będzie wiedza dotycząca trygonometrii i statystyki w podstawowym zakresie.

Styl przykładów

Przykłady z kodem w każdym podrozdziale zostały napisane w stylu prowadzenia korepetycji. Sprzyja to etapowemu budowaniu aplikacji i dobremu zrozumieniu sposobu działania algorytmów. W większości przypadków po utworzeniu nowej funkcji lub metody użyjesz jej w sesji

interaktywnej, aby zrozumieć, jak działa. Algorytmy są przeważnie prostymi wariantami, które mogą być rozszerzane na wiele sposobów. Wykonując przykłady i testując je w trybie interaktywnym, poznasz metody ulepszenia ich na potrzeby własnych zastosowań.

Dlaczego język Python?

Choć algorytmy są opisywane słownie z uwzględnionymi objaśnieniami formuł, znacznie bardziej przydatny (i prawdopodobnie łatwiejszy do prześledzenia) będzie faktyczny kod algorytmów i przykładowych problemów. Cały kod zawarty w książce napisano w znakomitym języku Python wysokiego poziomu. Zdecydowałem się na ten język z następujących powodów.

Zwięzłość
: Kod napisany za pomocą języków z dynamicznie określanym typem danych, takich jak Python, jest zwykle krótszy od kodu utworzonego za pomocą innych popularnych języków. Oznacza to mniej pisania podczas realizowania przykładów, ale też łatwiejsze dopasowanie kodu do obmyślonego algorytmu i naprawdę dobre zrozumienie tego, co jest wykonywane.

Łatwość czytania
: Zdarzało się, że język Python określano mianem „wykonywalnego pseudokodu". Choć oczywiście jest to przesada, wskazuje na fakt, że większość doświadczonych programistów może przeczytać kod języka Python i na tej podstawie zrozumieć, do czego ma on służyć. Niektóre z mniej oczywistych konstrukcji języka Python objaśniono w zamieszczonym dalej punkcie „Wskazówki związane z językiem Python".

Łatwość rozszerzania
: Standardowo język Python oferuje wiele bibliotek, w tym obsługujących funkcje matematyczne, analizę danych XML (*Extensible Markup Language*) i pobieranie stron internetowych. Niestandardowe biblioteki używane w książce, takie jak analizator RSS (*Really Simple Syndication*) i interfejs bazy danych SQLite, z łatwością i za darmo można pobrać, a następnie zainstalować i użytkować.

Interaktywność
: Podczas realizowania przykładu przydatna jest możliwość wypróbowywania funkcji w trakcie ich pisania bez konieczności tworzenia innego programu tylko na potrzeby testowania. Język Python umożliwia uruchamianie programów bezpośrednio z poziomu wiersza poleceń, a ponadto oferuje interaktywny wiersz zachęty, który pozwala wpisywać wywołania funkcji, tworzyć obiekty i interaktywnie testować pakiety.

Multiparadygmat
: Język Python obsługuje następujące style programowania: obiektowy, proceduralny i funkcyjny. Algorytmy uczenia maszynowego różnią się znacząco. Najbardziej oczywisty sposób implementacji algorytmu może się opierać na paradygmacie różniącym się od używanego w przypadku innego sposobu. Czasami przydatne jest przekazywanie funkcji jako parametrów, a czasami przechwytywanie stanu w obiekcie. Język Python obsługuje oba warianty.

Obsługa wielu platform i darmowość
: Język Python oferuje jedną implementację referencyjną dla wszystkich głównych platform, a ponadto jest bezpłatny dla nich wszystkich. Kod opisany w książce będzie działać na platformach Windows, Linux i Macintosh.

Wskazówki związane z językiem Python

Osobom początkującym, które są zainteresowane nauką programowania w języku Python, polecam lekturę książki *Python. Wprowadzenie* autorstwa Marka Lutza i Davida Aschera (wydawnictwo Helion). W książce dokonano znakomitego przeglądu tego języka. Programiści, którzy używają innych języków, powinni uznać kod Python za stosunkowo prosty do prześledzenia. Trzeba jednak mieć świadomość tego, że w książce używam części składni specyficznej dla języka Python, ponieważ pozwala mi ona bardziej bezpośrednio wyrazić algorytm lub podstawowe pojęcia. Oto krótki przegląd dla osób, które nie są programistami korzystającymi z języka Python.

Konstruktory list i słowników

Język Python oferuje dobry zestaw typów podstawowych oraz dwa typy, które intensywnie są stosowane w książce, czyli *lista* i *słownik*. Lista to uporządkowana lista wartości dowolnego typu tworzona za pomocą nawiasów kwadratowych:

```
lista_liczbowa=[1,2,3,4]
lista_łańcuchowa=['a', 'b', 'c', 'd']
lista_mieszana=['a', 3, 'c', 8]
```

Słownik to nieuporządkowany zbiór par złożonych z klucza i wartości, który przypomina tabelę mieszającą z innych języków. Słownik tworzony jest za pomocą nawiasów klamrowych:

```
wiek={'Jan':24,'Sara':28,'Michał':31}
```

Dostęp do elementów list i słowników można uzyskać za pomocą nawiasów kwadratowych wstawionych po nazwie listy:

```
lista_łańcuchowa[2]    # Zwraca 'b'.
wiek['Sara']     # Zwraca 28.
```

Znaczący znak odstępu

W przeciwieństwie do większości języków, język Python używa wcięcia kodu do definiowania bloków. Przyjrzyj się następującemu fragmentowi kodu:

```
if x==1:
  print 'x wynosi 1'
  print 'Nadal w bloku if'
print 'Poza blokiem if'
```

Interpreter wie, że pierwsze dwie instrukcje są wykonywane, gdy `x` wynosi 1, ponieważ użyto wcięcia kodu. Wcięcie może być złożone z dowolnej liczby spacji, pod warunkiem że jest jednolite. W książce używane jest wcięcie w postaci dwóch spacji. Podczas wprowadzania kodu konieczne będzie zwracanie uwagi na poprawne skopiowanie wcięcia.

Wyrażenia listowe

Wyrażenie listowe to wygodny sposób przekształcania jednej listy w drugą przez jej filtrowanie i stosowanie dla niej funkcji. Wyrażenie listowe jest zapisywane w następującej postaci:

```
[wyrażenie for zmienna in lista]
```

lub

```
[wyrażenie for zmienna in lista if warunek]
```

Na przykład następujący kod:

```
l1=[1,2,3,4,5,6,7,8,9]
print [v*10 for v in l1 if v1>4]
```

spowoduje wyświetlenie następującej listy:

```
[50,60,70,80,90]
```

Wyrażenia listowe są często używane w książce, ponieważ zapewniają wyjątkowo zwięzły sposób stosowania funkcji dla całej listy lub usuwania niepoprawnych pozycji. Inny sposób częstego ich wykorzystania powiązany jest z konstruktorem `dict`:

```
l1=[1,2,3,4,5,6,7,8,9]
timesten=dict([(v,v*10) for v in l1])
```

Powyższy kod utworzy słownik z pierwotną listą w roli kluczy oraz każdą pozycją pomnożoną przez 10 jako wartością:

```
{1:10,2:20,3:30,4:40,5:50,6:60,7:70,8:80,9:90}
```

Otwarte interfejsy API

Algorytmy syntetyzujące inteligencję zbiorową wymagają danych od wielu użytkowników. Oprócz algorytmów uczenia maszynowego, w książce omówiono kilka otwartych, internetowych interfejsów API (*Application Programming Interface*). Dzięki tym interfejsom firmy umożliwiają uzyskanie darmowego dostępu do danych z ich witryn internetowych z wykorzystaniem określonego protokołu. Możesz napisać programy, które pobierają i przetwarzają dane. Dane te składają się zwykle z wartości wprowadzonych przez użytkowników witryny. Mogą one być eksplorowane pod kątem nowych spostrzeżeń. W niektórych sytuacjach dostępna jest biblioteka języka Python, która zapewnia dostęp do tych interfejsów API. Jeśli tak nie jest, w naprawdę prosty sposób możesz utworzyć własny interfejs zapewniający dostęp do danych za pomocą wbudowanych bibliotek języka Python, które służą do pobierania danych i analizowania danych XML.

Oto niektóre witryny internetowe z otwartymi interfejsami API, które uwzględniono w książce.

del.icio.us
: Serwis społecznościowy do tworzenia zakładek, którego otwarte interfejsy API umożliwiają pobieranie odnośników według znaczników lub według konkretnego użytkownika.

Kayak
: Serwis podróżniczy z interfejsem API umożliwiającym przeprowadzanie wyszukiwań lotów i hoteli z poziomu własnych programów.

eBay
: Serwis aukcji internetowych z interfejsem API umożliwiającym kierowanie zapytań dotyczących pozycji, które aktualnie znajdują się w sprzedaży.

Hot or Not
: Serwis randkowy i oceniający osoby, z interfejsem API służącym do wyszukiwania osób oraz uzyskiwania związanych z nimi ocen i informacji demograficznych.

Akismet
: Interfejs API przeznaczony do filtrowania spamu w ramach pracy grupowej.

Ogromna liczba potencjalnych aplikacji może zostać zbudowana przez przetwarzanie danych z jednego źródła, łączenie danych z wielu źródeł, a nawet łączenie zewnętrznych informacji z danymi wprowadzonymi przez użytkowników naszego własnego oprogramowania. Możliwość wykorzystania na różne sposoby danych utworzonych przez użytkowników w różnych witrynach stanowi zasadniczy element tworzenia inteligencji zbiorowej. Witryna ProgrammableWeb (*http://www.programmableweb.com/*) to dobre miejsce, od którego warto zacząć poszukiwanie innych witryn internetowych z otwartymi interfejsami API.

Przegląd rozdziałów

Każdy algorytm zawarty w książce został zainicjowany rzeczywistym problemem, który, mam nadzieję, zostanie z łatwością zrozumiany przez wszystkich czytelników. Próbowałem uniknąć przykładów wymagających obszernej wiedzy, a ponadto skoncentrowałem się na problemach, które choć złożone, z łatwością zostaną zrozumiane przez większość osób.

Rozdział 1., „Inteligencja zbiorowa — wprowadzenie"
: W rozdziale objaśniono pojęcia związane z uczeniem maszynowym, omówiono zastosowanie go w różnych dziedzinach, a także przybliżono, jak może ono zostać wykorzystane do wyciągania nowych wniosków na podstawie danych zgromadzonych od wielu różnych osób.

Rozdział 2., „Tworzenie rekomendacji"
: W rozdziale zaprezentowano techniki *filtrowania w ramach pracy grupowej* używane przez wielu sprzedawców internetowych do rekomendowania produktów lub środków przekazu. Rozdział zawiera podrozdział poświęcony rekomendowaniu internautom odnośników z serwisu społecznościowego umożliwiającego tworzenie zakładek, a także budowaniu systemu rekomendacji filmów przy użyciu zbioru danych MovieLens.

Rozdział 3., „Wykrywanie grup"
: W tym rozdziale, który wykorzystuje pojęcia z rozdziału 2., wprowadzono dwie różne metody *grupowania*, które automatycznie wykrywają grupy podobnych pozycji w dużym zbiorze danych. Zademonstrowano w nim użycie grupowania do znajdowania grup na podstawie zbioru popularnych blogów internetowych i oczekiwań użytkowników serwisu społecznościowego służącego do tworzenia zakładek.

Rozdział 4., „Wyszukiwanie i klasyfikowanie"
: W rozdziale opisano różne części wyszukiwarki internetowej, w tym przeszukiwacz, indeksator i mechanizm zapytań. Omówiono algorytm *PageRank* służący do oceniania stron na podstawie zewnętrznych odnośników, a także wyjaśniono, jak utworzyć *sieć neuronową*, która uczy się, jakie słowa kluczowe są powiązane z różnymi wynikami.

Rozdział 5., „Optymalizacja"
: W rozdziale wprowadzono algorytmy optymalizacji, które mają za zadanie przeszukiwanie milionów możliwych rozwiązań problemu i wybranie najlepszego z nich. Zademonstrowano bardzo różne zastosowania tych algorytmów wraz z przykładami, w których znajdowane są najlepsze połączenia lotnicze dla grupy osób podróżujących w to samo miejsce, najlepszy sposób dopasowania studentów do akademików oraz tworzony jest układ sieci z minimalną liczbą skrzyżowanych linii.

Rozdział 6., „Filtrowanie dokumentów”

W rozdziale omówiono *filtrowanie bayesowskie*, które jest wykorzystywane w wielu darmowych i komercyjnych filtrach spamu do automatycznego klasyfikowania dokumentów na podstawie typu wyrazów i innych właściwości występujących w dokumencie. Takie filtrowanie zostało zastosowane do zestawu wyników wyszukiwania RSS, aby zademonstrować automatyczną klasyfikację wpisów.

Rozdział 7., „Modelowanie przy użyciu drzew decyzyjnych”

W rozdziale wprowadzono *drzewa decyzyjne* jako metodę, która nie tylko tworzy przewidywania, ale też modeluje sposób podejmowania decyzji. Pierwsze drzewo decyzyjne, które zostało zbudowane przy użyciu hipotetycznych danych z dzienników serwera, posłużyło do przewidzenia, czy użytkownik stanie się subskrybentem rozbudowanej usługi. W innych przykładach użyto danych z prawdziwych witryn internetowych w celu modelowania cen nieruchomości i „atrakcyjności”.

Rozdział 8., „Budowanie modelu cen”

W rozdziale zajęto się problemem przewidywania, zamiast klasyfikacji, wartości liczbowych przy użyciu technik *k*-najbliższych sąsiadów. Zastosowano również algorytmy optymalizacji z rozdziału 5. Metody te są używane w połączeniu z interfejsem API serwisu *eBay* do budowania systemu przewidywania końcowych cen licytacji pozycji na podstawie zbioru właściwości.

Rozdział 9., „Zaawansowane klasyfikowanie: metody jądrowe i maszyny wektorów nośnych”

W rozdziale pokazano, jak *maszyny wektorów nośnych* mogą być używane do dopasowywania osób w internetowych serwisach randkowych lub na potrzeby wyszukiwania kontaktów zawodowych. Maszyny wektorów nośnych to dość zaawansowana technika. W rozdziale dokonano porównania jej z innymi metodami.

Rozdział 10., „Znajdowanie niezależnych właściwości”

W rozdziale wprowadzono stosunkowo nową technikę, określaną mianem *nieujemnej faktoryzacji macierzy*, która służy do znajdowania niezależnych właściwości w zbiorze danych. W wielu zbiorach pozycje są tworzone jako zestaw różnych właściwości, które nie są wcześniej znane. W przypadku omawianej techniki chodzi o wykrycie tych właściwości. Użycie techniki zademonstrowano dla zbioru artykułów informacyjnych, gdzie same artykuły służą do wykrycia tematów, spośród których co najmniej jeden może dotyczyć danego artykułu.

Rozdział 11., „Inteligencja rozwojowa”

W rozdziale wprowadzono pojęcie *programowania genetycznego*. Jest to bardzo zaawansowany zestaw technik, który w celu rozwiązania konkretnego problemu wykracza poza optymalizację i faktycznie buduje algorytmy z wykorzystaniem pojęć związanych z ewolucją. Zademonstrowano to za pomocą prostej gry, w której początkowo komputer jest kiepskim graczem, a później zwiększa swoje umiejętności przez ulepszanie własnego kodu wraz z coraz dłuższym czasem poświęconym na grę.

Rozdział 12., „Algorytmy — podsumowanie”

W rozdziale dokonano przeglądu wszystkich opisanych w książce algorytmów uczenia maszynowego i algorytmów statystycznych, a ponadto porównano je za pomocą zestawu sztucznych problemów. Ułatwi to zrozumienie sposobu ich działania i wyobrażenie sobie, jak każdy z nich dzieli dane.

Dodatek A, „Zewnętrzne biblioteki”
: W dodatku podano informacje o zewnętrznych bibliotekach używanych w książce, w tym dotyczące miejsca, gdzie można je znaleźć, a także sposobu ich zainstalowania.

Dodatek B, „Formuły matematyczne”
: Dodatek zawiera formuły, opisy i kod dla wielu pojęć matematycznych wprowadzonych w książce.

Ćwiczenia zamieszczone na końcu każdego rozdziału zawierają przykłady metod pozwalających rozszerzyć algorytmy i sprawić, że będą miały większe możliwości.

Konwencje

W książce zastosowano następujące konwencje typograficzne.

Zwykły tekst
: Ten styl stosowany jest w przypadku standardowej treści.

Kursywa
: Za pomocą tego stylu wyróżniane są nowe terminy, adresy URL, adresy e-mail, nazwy i rozszerzenia plików, ścieżki z nazwami, katalogi i programy narzędziowe systemu Unix.

`Czcionka o stałej szerokości`
: Przy użyciu tego stylu wyróżniane są polecenia, opcje, przełączniki, zmienne, atrybuty, klucze, funkcje, typy, klasy, przestrzenie nazw, metody, moduły, właściwości, parametry, wartości, obiekty, procedury obsługi zdarzeń oraz znaczniki XML i HTML.

```
Listing
```
: Ten styl używany jest dla zawartości plików z kodem lub wyników poleceń.

`Listing z pogrubieniem`
: Tym stylem wyróżniane są polecenia lub inny tekst, który bez zmian powinien zostać wprowadzony przez użytkownika.

`Listing z kursywą`
: Za pomocą tego stylu wyróżniany jest tekst, który powinien zostać zastąpiony wartościami podanymi przez użytkownika.

Ta ikona oznacza wskazówkę, sugestię lub ogólną uwagę.

Użycie przykładów z kodem

Książka ma ułatwić pracę. Ogólnie rzecz biorąc: możesz używać w swoich programach i dokumentacji kodu zawartego w książce. Kontakt z nami nie jest wymagany w celu uzyskania zgody, jeśli nie zamierzasz powielać znacznej części kodu. Na przykład pisanie programu, który korzysta z kilku porcji kodu z książki, nie wymaga zgody. Sprzedaż lub dystrybucja dysku CD-ROM z przykładami z książek wydawnictwa *wymaga* zgody. Udzielanie odpowiedzi na pytanie przez zacytowanie fragmentu treści książki oraz przykładowego kodu nie wymaga zgody.

Uwzględnienie znacznej ilości przykładowego kodu z książki w dokumentacji własnego produktu *wymaga* zgody.

Doceniamy podawanie informacji o twórcy, lecz nie wymagamy tego. Informacje te obejmują zwykle tytuł, autora, wydawcę i numer ISBN.

Jeśli uznasz, że użycie przykładów z kodem wykracza poza przyjęte zasady lub wymaga uzyskania zgody, skontaktuj się z nami.

Podziękowania

Chciałbym wyrazić wdzięczność wszystkim osobom z wydawnictwa O'Reilly, które uczestniczyły w pracach związanych z tworzeniem i wydawaniem książki. Przede wszystkim dziękuję Natowi Torkingtonowi za wskazanie mi, że pomysł jest wartościowy i zasługuje na zrealizowanie. Podziękowania kieruję do Mike'a Hendricksona i Briana Jepsona za wysłuchanie moich argumentów i wywołanie we mnie uczucia podekscytowania możliwością napisania książki. Szczególnie dziękuję Mary O'Brien, która przejęła obowiązki redaktora od Briana, a ponadto zawsze łagodziła moje obawy, że projekt jest dla mnie zbyt dużym wyzwaniem.

Chcę podziękować następującym członkom zespołu wydawniczego: Marlowe Shaeffer, Rob Romano, Jessamyn Read, Amy Thomson i Sarah Schneider, za to, że wykonane przeze mnie ilustracje i napisana treść przyjęły postać, która faktycznie może zasługiwać na zainteresowanie się nią.

Dziękuję wszystkim, którzy wzięli udział w recenzowaniu książki. W szczególności są to następujące osoby: Paul Tyma, Matthew Russell, Jeff Hammerbacher, Terry Camerlengo, Andreas Weigend, Daniel Russell i Tim Wolters.

Dziękuję moim rodzicom.

I wreszcie dużą wdzięczność jestem winien kilku moim znajomym, którzy pomogli mi w procesie definiowania pomysłów związanych z książką, a ponadto zawsze okazywali zrozumienie, gdy nie miałem dla nich czasu. Są to: Andrea Matthews, Jeff Beene, Laura Miyakawa, Neil Stroup i Brooke Blumenstein. Prace nad książką byłyby znacznie trudniejsze bez Waszego wsparcia. Z pewnością pominąłbym część bardziej zajmujących przykładów.

ROZDZIAŁ 1.

Inteligencja zbiorowa — wprowadzenie

Netflix to internetowa wypożyczalnia płyt DVD, która umożliwia wybór filmów z wysyłką do domu. Firma podaje rekomendacje na podstawie filmów, które zostały wcześniej wypożyczone przez klientów. Pod koniec 2006 r. firma Netflix poinformowała o nagrodzie w wysokości 1 mln dolarów dla pierwszej osoby, która poprawi dokładność systemu rekomendacji wypożyczalni o 10%. Ponadto każdego roku firma będzie wręczać dodatkowo 50 tys. dolarów aktualnemu liderowi do czasu trwania konkursu. W konkursie wzięły udział tysiące zespołów z całego świata. W kwietniu 2007 r. najlepszemu zespołowi udało się uzyskać poprawę rekomendacji o 7%. Korzystając z danych dotyczących filmów, które spodobały się poszczególnym klientom, firma Netflix ma możliwość rekomendowania filmów innym klientom. Ci klienci mogli nawet nigdy o nich nie słyszeć. Po ich obejrzeniu mogą oni zdecydować się na kolejne filmy. Każdy sposób ulepszenia swojego systemu rekomendacji wart jest dla firmy Netflix mnóstwo pieniędzy.

Wyszukiwarka internetowa firmy Google zaczęła działać w 1998 r. W tamtym czasie istniało już kilka dużych wyszukiwarek. Wiele osób przyjęło, że nowy gracz nie będzie w stanie konkurować z gigantami branżowymi. Jednakże założyciele firmy Google zastosowali zupełnie nową metodę tworzenia rankingów wyników wyszukiwania, korzystając z odnośników na milionach stron internetowych podczas określania, które strony są najodpowiedniejsze. Wyniki wyszukiwania wyszukiwarki Google okazały się znacznie lepsze od oferowanych przez innych graczy, którzy w 2004 r. obsługiwali 85% wyszukiwań w Internecie. Założyciele firmy Google zaliczają się obecnie do najbogatszych ludzi świata.

Co te dwie firmy mają ze sobą wspólnego? Obie doszły do nowych wniosków i stworzyły nowe możliwości biznesowe, korzystając z zaawansowanych algorytmów w celu połączenia danych zebranych od wielu różnych osób. Możliwość gromadzenia informacji i moc obliczeniowa pozwalająca na ich interpretowanie stworzyły ogromne możliwości współpracy oraz lepszego zrozumienia użytkowników i klientów. Tego rodzaju działania mają miejsce w różnych przypadkach. Serwisom randkowym zależy na szybszym znalezieniu najlepiej dopasowanych kandydatów. Pojawiają się firmy przewidujące zmiany cen biletów lotniczych. Niemal każdemu zależy na lepszym zrozumieniu swoich klientów, aby przygotować reklamy trafiające do właściwszych osób.

To tylko kilka przykładów ekscytującej dziedziny inteligencji zbiorowej. Rozpowszechnianie się nowych usług powoduje, że każdego dnia pojawiają się nowe możliwości. Wierzę, że opanowanie uczenia maszynowego i metod statystycznych stanie się jeszcze ważniejsze w przeróżnych dziedzinach, szczególnie w przypadku interpretowania i organizowania ogromnej ilości informacji tworzonych przez ludzi na całym świecie.

Czym jest inteligencja zbiorowa?

Pojęcie *inteligencji zbiorowej* jest używane od dziesięcioleci. Jego znaczenie i popularność zaczęły się zwiększać wraz z pojawieniem się nowych technologii komunikacji. Choć nazwa terminu może przywodzić na myśl pojęcia związane ze świadomością zbiorową lub zjawiskiem nadprzyrodzonym, to używając go, specjaliści od technologii mają zwykle na myśli łączenie zachowań, preferencji lub pomysłów grupy osób w celu uzyskania nowatorskich spostrzeżeń.

Oczywiście inteligencja zbiorowa była możliwa przed pojawieniem się Internetu. Nie jest wymagana sieć WWW, aby zgromadzić dane od różnych grup ludzi, połączyć je i poddać analizie. Jedną z najbardziej podstawowych form inteligencji zbiorowej jest ankieta lub spis ludności. Zbieranie odpowiedzi od dużej grupy ludzi umożliwia uzyskanie wniosków statystycznych dotyczących grupy, których nie określiłby w pojedynkę żaden jej członek. Tworzenie nowych wniosków przy udziale niezależnych uczestników w rzeczywistości jest tym, do czego służy inteligencja zbiorowa.

Jej dobrze znanym przykładem są rynki finansowe, w przypadku których cena nie jest ustalana przez jedną osobę ani nie jest wynikiem skoordynowanego działania, lecz stanowi efekt operacji handlowych wielu niezależnych od siebie osób, które działają zgodnie z tym, co w ich przekonaniu służy ich najlepszemu interesowi. Choć z początku może się to wydawać sprzeczne z intuicją, *rynki kontraktów terminowych*, gdzie wielu uczestników handluje kontraktami, próbując określić ich przyszłe ceny, są uważane za lepsze w przewidywaniu cen niż eksperci, którzy niezależnie przygotowują prognozy. Wynika to stąd, że w przypadku tworzenia przewidywań takie rynki łączą w sobie wiedzę, doświadczenie i spostrzeżenia tysięcy osób, a nie analizują jedynie punkt widzenia jednej osoby.

Chociaż metody inteligencji zbiorowej istniały przed powstaniem Internetu, możliwość gromadzenia informacji od tysięcy, a nawet milionów osób w sieci internetowej stworzyła wiele nowych opcji analizy. Ludzie używają Internetu do robienia zakupów, prowadzenia badań, szukania rozrywki i budowania własnych witryn internetowych. Wszystkie te działania mogą być monitorowane i wykorzystywane do uzyskiwania informacji bez żadnej konieczności wpływania na intencje użytkownika przez zadawanie mu pytań. Istnieje ogromna liczba możliwych metod przetwarzania i interpretowania tych informacji. Oto dwa kluczowe przykłady prezentujące przeciwstawne metody.

- *Wikipedia* to internetowa encyklopedia stworzona w całości w wyniku współpracy użytkowników. Dowolna strona może zostać utworzona lub zmodyfikowana przez każdego. Niewielka liczba administratorów monitoruje powtarzające się nadużycia. Serwis Wikipedia ma więcej wpisów niż jakakolwiek inna encyklopedia. Pomimo manipulacji dokonywanych przez użytkowników o złych intencjach, generalnie może być uważana za dokładną w przypadku większości zagadnień. Jest to przykład inteligencji zbiorowej, ponieważ każdy artykuł jest utrzymywany przez dużą grupę osób. Efektem jest encyklopedia znacznie większa od jakiejkolwiek, która mogłaby zostać stworzona przez dowolną pojedynczą, skoordynowaną grupę. Oprogramowanie encyklopedii Wikipedia nie realizuje żadnych wyszukanych operacji w odniesieniu do uczestniczących użytkowników. Po prostu śledzi zmiany i wyświetla najnowszą wersję.
- Wspomniana wcześniej wyszukiwarka *Google* to najpopularniejsza na świecie wyszukiwarka internetowa. Jako pierwsza zaczęła oceniać strony internetowe na podstawie liczby innych stron, które się do nich odwołują. W przypadku tej metody oceniania uzyskiwane

są informacje o tym, co tysiące osób stwierdziły na temat konkretnej strony internetowej. Informacje te służą do tworzenia rankingu wyników wyszukiwania. Jest to rzykład inteligencji zbiorowej bardzo odmienny od Wikipedii. Serwis Wikipedia wprost zaprasza swoich użytkowników do uczestnictwa, natomiast wyszukiwarka Google wydobywa ważne informacje o tym, jakie działania twórcy treści sieciowych podejmują w obrębie własnych witryn, a następnie wykorzystuje je do generowania wyników dla swoich użytkowników.

Choć Wikipedia stanowi znakomity zasób i wyjątkowy przykład inteligencji zbiorowej, swoje istnienie zawdzięcza bardziej bazie użytkowników dodających informacje niż sprytnym algorytmom zawartym w oprogramowaniu. W książce skoncentrowano się na drugim końcu tego spektrum, czyli na omówieniu algorytmów takich jak PageRank wyszukiwarki Google, który pobiera dane użytkownika i przeprowadza obliczenia w celu utworzenia nowych informacji mogących wpłynąć na poprawę komfortu obsługi użytkownika. Część danych jest gromadzona jawnie, być może przez prośbę o ocenę różnych rzeczy skierowaną do internautów, a część jest zbierana mimochodem przez obserwację tego, co jest przez nich kupowane. W obu przypadkach istotne jest nie samo zbieranie i wyświetlanie informacji, lecz przetwarzanie ich w inteligentny sposób i generowanie nowych wiadomości.

W książce zostaną zaprezentowane metody gromadzenia danych za pośrednictwem otwartych interfejsów API. Przedstawione będą różne algorytmy uczenia maszynowego oraz metody statystyczne. Taka kombinacja umożliwi przygotowanie metod inteligencji zbiorowej dla danych uzyskanych we własnych aplikacjach, a także gromadzenie danych z innych miejsc i eksperymentowanie z ich wykorzystaniem.

Czym jest uczenie maszynowe?

Uczenie maszynowe to dziedzina podlegająca sztucznej inteligencji związanej z algorytmami, które umożliwiają *uczenie* komputerów. W większości przypadków oznacza to, że algorytm otrzymuje zbiór danych i określa wnioski dotyczące ich właściwości. Informacje te umożliwiają tworzenie przewidywań odnośnie do innych danych, które mogą pojawić się w przyszłości. Jest to możliwe, ponieważ niemal wszystkie nielosowe dane zawierają wzorce, które pozwalają maszynie dokonywać uogólnień. W tym celu trenowany jest *model* przy użyciu tego, co maszyna uzna za najważniejsze aspekty danych.

Aby zrozumieć, jak powstają modele, rozważmy prosty przykład ze skomplikowanej dziedziny, jaką jest filtrowanie poczty elektronicznej. Załóżmy, że otrzymywana jest spora ilość spamu, który zawiera słowa „apteka internetowa". Człowiek ma odpowiednie możliwości rozpoznawania wzorców, dlatego potrafi szybko stwierdzić, że każda wiadomość zawierająca te dwa słowa to spam, który powinien trafić bezpośrednio do kosza. Jest to uogólnienie. W rzeczywistości został utworzony myślowy model tego, czym jest spam. Po zgłoszeniu kilku takich wiadomości jako spamu algorytm uczenia maszynowego zaprojektowany do filtrowania spamu powinien być w stanie dokonać takiego samego uogólnienia.

Istnieje wiele różnych algorytmów uczenia maszynowego, cechujących się różną siłą działania i dopasowanych do różnego typu problemów. Niektóre z nich, takie jak drzewa decyzyjne, są transparentne. Oznacza to, że obserwator może w pełni pojąć proces rozumowania realizowany przez maszynę. Inne algorytmy, takie jak sieci neuronowe, to *„czarna skrzynka"*. Oznacza to, że generują one odpowiedź, często jednak bardzo trudne jest odtworzenie związanego z tym rozumowania.

Wiele algorytmów uczenia maszynowego intensywnie korzysta z matematyki i statystyki. Zgodnie z definicją, którą wcześniej podałem, można nawet stwierdzić, że prosta analiza korelacji i regresja to podstawowe formy uczenia maszynowego. W książce nie założono, że czytelnik ma wiedzę z dziedziny statystyki, dlatego podjąłem się próby objaśnienia zastosowanej statystyki w jak najprostszy sposób.

Ograniczenia uczenia maszynowego

Uczenie maszynowe nie jest pozbawione wad. Algorytmy mają różne możliwości uogólniania dużych zbiorów wzorców. Wzorzec, który nie przypomina żadnego wcześniej napotkanego przez algorytm, z dużym prawdopodobieństwem zostanie niewłaściwie zinterpretowany. Ludzie mogą wykorzystywać rozległe doświadczenie i wiedzę o charakterze kulturowym, a także mają niezwykłą zdolność rozpoznawania podobnych sytuacji podczas podejmowania decyzji dotyczących nowych wiadomości. Z kolei metody uczenia maszynowego mogą jedynie uogólniać na podstawie już napotkanych danych, i to w bardzo ograniczony sposób.

Metoda filtrowania spamu, która zostanie przedstawiona w książce, opiera się na występowaniu słów lub fraz bez względu na ich znaczenie lub na strukturę zdań. Choć teoretycznie możliwe jest zbudowanie algorytmu, który uwzględniałby gramatykę, w praktyce dzieje się to rzadko z powodu wymaganych nakładów nieproporcjonalnie dużych w stosunku do uzyskiwanego ulepszenia algorytmu. Zrozumienie znaczenia słów lub ich powiązania z życiem danej osoby wymagałoby znacznie większej ilości informacji niż ta, do której mogą uzyskać dostęp filtry spamu w swojej obecnej postaci.

Poza tym, choć wszystkie metody uczenia maszynowego różnią się pod tym względem, są podatne na możliwość przesadnego uogólniania. Jak z większością rzeczy w życiu, duże uogólnienia oparte na kilku przykładach rzadko są w pełni dokładne. Z pewnością możliwe jest otrzymanie od znajomego ważnej wiadomości e-mail, która zawiera słowa „apteka internetowa". W tym przypadku poinstruowano by algorytm, że wiadomość nie jest spamem. W rezultacie algorytm mógłby wywnioskować, że komunikaty od tego konkretnego znajomego są możliwe do zaakceptowania. Natura wielu algorytmów uczenia maszynowego jest taka, że mogą one kontynuować proces uczenia wraz z pojawianiem się nowych informacji.

Rzeczywiste przykłady

W Internecie istnieje wiele witryn, które obecnie gromadzą dane od wielu różnych osób, a ponadto stosują uczenie maszynowe i metody statystyczne w celu skorzystania z nich. Wyszukiwarka Google to prawdopodobnie największe rozwiązanie (nie tylko używa łączy internetowych do tworzenia rankingu stron, ale nieustannie zbiera informacje dotyczące momentu kliknięcia reklam przez różnych użytkowników), które umożliwia firmie Google bardziej skuteczne kierowanie reklam. W rozdziale 4. zostaną omówione wyszukiwarki internetowe i algorytm PageRank, który stanowi istotną część systemu rankingowego wyszukiwarki Google.

Inne przykłady obejmują witryny internetowe z systemami rekomendacji. Witryny takich firm, jak Amazon i Netflix, używają informacji o rzeczach kupionych lub wypożyczonych przez ludzi do określania, jacy internauci lub jakie produkty są do siebie podobne, a następnie tworzenia rekomendacji na podstawie historii zakupów. Inne witryny, takie jak Pandora i Last.fm, stosują oceny użytkowników dotyczące różnych zespołów i piosenek, aby tworzyć tematyczne

stacje radiowe z muzyką, która w opinii ich właścicieli powinna być interesująca. W rozdziale 2. omówiono metody budowania systemów rekomendacji.

Rynki prognostyczne to także forma inteligencji zbiorowej. Jednym z najbardziej znanych jest serwis Hollywood Stock Exchange (*http://hsx.com/*), w którym użytkownicy handlują akcjami związanymi z filmami i gwiazdami filmowymi. Możliwe jest kupno lub sprzedaż akcji po aktualnej cenie, jeśli wiadomo, że jej ostateczna cena będzie jedną milionową rzeczywistej kwoty w momencie premiery filmu. Ze względu na to, że cena jest zależna od handlujących akcjami, jej wysokość nie jest ustalana przez żadną konkretną osobę, lecz jako wynik działania grupy. Aktualna cena może być przewidywaniem całej grupy dotyczącym wyniku finansowego filmu po premierze. Przewidywania określane przez serwis Hollywood Stock Exchange są często lepsze od opracowywanych przez poszczególnych ekspertów.

Niektóre serwisy randkowe, takie jak eHarmony, używają informacji zebranych od uczestników do określenia, kto byłby odpowiednim kandydatem. Choć takie firmy utrzymują zwykle stosowane metody dopasowywania osób w tajemnicy, całkiem prawdopodobne jest, że dowolna skuteczna metoda będzie uwzględniać ciągłe ponawianie oceny na podstawie tego, czy wybrani kandydaci faktycznie zostali do siebie pomyślnie dopasowani.

Inne zastosowania algorytmów uczących

Metody opisane w książce nie są nowe. Choć przykłady skupiają się na problemach z inteligencją zbiorową w przypadku zastosowań internetowych, znajomość algorytmów uczenia maszynowego może okazać się pomocna dla twórców oprogramowania w wielu innych dziedzinach. Algorytmy te są szczególnie przydatne w obszarach, w których wykorzystuje się duże zbiory danych przeszukiwane pod kątem interesujących wzorców. Oto przykłady.

Biotechnologia
: Postępy w technologii sekwencjonowania i badania przesiewowego spowodowały utworzenie ogromnych zbiorów różnych rodzajów danych, takich jak sekwencje kodu DNA, struktury białek, przesiewy związków chemicznych i ekspresja RNA. Techniki uczenia maszynowego są intensywnie wykorzystywane w przypadku wszystkich tego rodzaju danych. Ma to na celu znalezienie wzorców, które zwiększają stopień zrozumienia procesów biologicznych.

Wykrywanie oszustw finansowych
: Firmy obsługujące karty kredytowe nieustannie poszukują nowych sposobów wykrywania nielegalnych transakcji. W związku z tym zastosowały one takie techniki, jak sieci neuronowe i logika indukcyjna, aby weryfikować transakcje i wychwytywać przypadki niewłaściwego użycia.

System wizyjny
: Interpretowanie obrazów z kamery wideo do celów wojskowych lub obserwacyjnych to aktywny obszar badań. Wiele technik uczenia maszynowego używanych jest w celu podejmowania próby automatycznego wykrywania intruzów, identyfikowania pojazdów lub rozpoznawania twarzy. Szczególnie interesujące jest zastosowanie technik nienadzorowanych, takich jak *niezależna analiza komponentów*, która umożliwia znajdowanie interesujących właściwości w dużych zbiorach danych.

Marketing produktów

Przez bardzo długi czas zrozumienie demografii i trendów było bardziej formą sztuki niż nauką. Zwiększona w ostatnim czasie możliwość gromadzenia danych od konsumentów zapewniła opcje wykorzystania technik uczenia maszynowego, takich jak grupowanie, aby lepiej zrozumieć naturalne podziały istniejące na rynkach i przygotować precyzyjniejsze przewidywania dotyczące przyszłych trendów.

Optymalizacja łańcucha dostaw

Duże organizacje mogą zaoszczędzić miliony dolarów dzięki efektywnemu funkcjonowaniu ich łańcuchów dostaw i dokładnemu przewidywaniu zapotrzebowania na produkty w różnych obszarach. Liczba możliwych metod tworzenia łańcucha dostaw jest ogromna, tak samo jak liczba czynników, które potencjalnie mogą mieć wpływ na popyt. Optymalizacja i techniki uczenia są często używane do analizowania związanych z tym zbiorów danych.

Analiza rynków giełdowych

Od czasu powstania rynku giełdowego ludzie podejmowali próby wykorzystania matematyki do zarobienia większej ilości pieniędzy. Wraz z coraz większym stopniem zaawansowania uczestników rynku akcji stało się konieczne analizowanie większych zbiorów danych i używanie zaawansowanych technik do wykrywania wzorców.

Bezpieczeństwo narodowe

Ogromna ilość informacji jest gromadzona przez agencje rządowe całego świata. Analiza tych danych wymaga od komputerów wykrywania wzorców i wiązania ich z potencjalnymi zagrożeniami.

To zaledwie kilka przykładów intensywnego wykorzystywania uczenia maszynowego. Z powodu tego, że tendencją jest generowanie większej ilości informacji, prawdopodobnie w większej liczbie dziedzin konieczne będzie wykorzystanie uczenia maszynowego i metod statystycznych, gdy ilość informacji przekroczy ludzkie możliwości zarządzania nimi przy użyciu starych sposobów.

Biorąc pod uwagę, jak dużo nowych informacji udostępnianych jest każdego dnia, oczywiście pojawia się znacznie więcej możliwości. Po poznaniu kilku algorytmów uczenia maszynowego zaczną być zauważalne przeróżne miejsca, w których mogą one zostać wykorzystane.

ROZDZIAŁ 2.

Tworzenie rekomendacji

W celu rozpoczęcia omówienia inteligencji zbiorowej zaprezentuję sposoby użycia preferencji grupy osób do tworzenia rekomendacji dla innych osób. Istnieje wiele zastosowań dla tego typu informacji, takich jak tworzenie rekomendacji produktów na potrzeby zakupów internetowych, proponowanie interesujących witryn internetowych lub ułatwianie internautom znajdowania muzyki i filmów. W tym rozdziale pokazano, jak zbudować system do znajdowania osób, które mają podobne gusta, a także do tworzenia automatycznych rekomendacji na podstawie rzeczy lubianych przez innych.

Prawdopodobnie spotkałeś się wcześniej z mechanizmami rekomendacji, np. podczas robienia zakupów internetowych w takich witrynach jak *Amazon*. Firma *Amazon* śledzi zwyczaje zakupowe wszystkich swoich klientów. Po zalogowaniu użytkownika w witrynie korzysta ona z tego faktu, aby zasugerować produkty, które użytkownikowi mogą się spodobać. Serwis *Amazon* może nawet zaproponować filmy, które mogą przypaść do gustu, niezależnie od tego, że wcześniej zakupiono wyłącznie książki. Niektóre internetowe agencje sprzedające bilety na koncerty będą sprawdzać historię koncertów, na które klienci dotychczas się udali, a następnie powiadomią ich o zbliżających się koncertach, będących ewentualnie dla nich interesującymi. Witryny w rodzaju *reddit.com* umożliwiają głosowanie na odnośniki do innych witryn internetowych, a następnie na podstawie zebranych głosów sugerują inne odnośniki, które mogą być dla użytkowników interesujące.

Powyższe przykłady pokazują, że preferencje mogą być gromadzone na wiele różnych sposobów. Czasami dane dotyczą produktów zakupionych przez ludzi, a opinie na ich temat mogą być reprezentowane przez głosy w postaci tak/nie lub oceny w skali od 1 do 5. W rozdziale przyjrzymy się różnym sposobom reprezentowania tych przypadków tak, aby wszystkie mogły być obsługiwane przez ten sam zestaw algorytmów. Ponadto zostaną utworzone praktyczne przykłady, które wykorzystują oceny krytyków filmowych i zakładki społecznościowe.

Filtrowanie grupowe

Jak wiadomo, zwykła metoda zdobywania rekomendacji produktów, filmów lub rozrywkowych witryn internetowych polega na poproszeniu o to znajomych. Wiadomo również, że niektórzy znajomi mają lepsze „gusta" niż inni. Jest to coś, co stwierdzono z czasem, obserwując, czy zwykle znajomym podobają się te same rzeczy co nam. Wraz z pojawianiem się coraz większej liczby opcji mniej praktyczne staje się decydowanie o tym, co jest niezbędne, przez

pytanie o to niewielkiej grupy osób. Wynika to stąd, że mogą one nie być świadome wszystkich opcji. Z tego właśnie powodu został opracowany zestaw technik określanych mianem *filtrowania grupowego*.

Działanie algorytmu filtrowania grupowego polega zwykle na wyszukiwaniu dużej grupy ludzi i znajdowaniu mniejszego zbioru osób o gustach podobnych do naszych. Algorytm sprawdza inne rzeczy lubiane przez te osoby i łączy je w celu utworzenia sklasyfikowanej listy sugestii. Można wyróżnić kilka różnych metod decydowania, jakie osoby są podobne, a także łączenia dokonanych przez nie wyborów w celu utworzenia listy. W rozdziale zostanie omówionych kilka z tych metod.

Pojęcie *filtrowanie grupowe* po raz pierwszy zostało użyte w 1992 r. przez Davida Goldberga, pracownika firmy Xerox PARC, w dokumencie zatytułowanym *Using collaborative filtering to weave an information tapestry* (Użycie filtrowania grupowego do poradzenia sobie ze złożonością informacji). David zaprojektował system o nazwie *Tapestry*, który umożliwił ludziom ocenianie dokumentów jako interesujących lub nieciekawych. Zebrane informacje służyły do filtrowania dokumentów dla innych osób.

Obecnie istnieją setki witryn internetowych, które wykorzystują algorytm filtrowania grupowego na potrzeby filmów, książek, randek, zakupów, innych witryn, publikacji *podcast*, artykułów, a nawet dowcipów.

Gromadzenie preferencji

Pierwszą niezbędną rzeczą jest metoda reprezentowania różnych osób i ich preferencji. W języku Python bardzo prosty sposób, który to umożliwia, polega na użyciu *słownika zagnieżdżonego*. Aby samemu sprawdzić przykład zamieszczony w tym podrozdziale, należy utworzyć plik o nazwie *recommendations.py*, a następnie umieścić w nim następujący kod w celu utworzenia zbioru danych:

```
# słownik krytyków filmowych i ich ocen niewielkiego
# zestawu filmów
critics={'Lisa Rose': {'Kobieta w błękitnej wodzie': 2.5, 'Węże w samolocie': 3.5,
 'Całe szczęście': 3.0, 'Superman: Powrót': 3.5, 'Ja, ty i on': 2.5,
 'Nocny słuchacz': 3.0},
'Gene Seymour': {'Kobieta w błękitnej wodzie': 3.0, 'Węże w samolocie': 3.5,
 'Całe szczęście': 1.5, 'Superman: Powrót': 5.0, 'Nocny słuchacz': 3.0,
 'Ja, ty i on': 3.5},
'Michael Phillips': {'Kobieta w błękitnej wodzie': 2.5, 'Węże w samolocie': 3.0,
 'Superman: Powrót': 3.5, 'Nocny słuchacz': 4.0},
'Claudia Puig': {'Węże w samolocie': 3.5, 'Całe szczęście': 3.0,
 'Nocny słuchacz': 4.5, 'Superman: Powrót': 4.0,
 'Ja, ty i on': 2.5},
'Mick LaSalle': {'Kobieta w błękitnej wodzie': 3.0, 'Węże w samolocie': 4.0,
 'Całe szczęście': 2.0, 'Superman: Powrót': 3.0, 'Nocny słuchacz': 3.0,
 'Ja, ty i on': 2.0},
'Jack Matthews': {'Kobieta w błękitnej wodzie': 3.0, 'Węże w samolocie': 4.0,
 'Nocny słuchacz': 3.0, 'Superman: Powrót': 5.0, 'Ja, ty i on': 3.5},
'Toby': {'Węże w samolocie':4.5,'Ja, ty i on':1.0,'Superman: Powrót':4.0}}
```

W rozdziale praca z kodem Python będzie odbywać się interaktywnie, dlatego plik *recommendations.py* należy zapisać w miejscu, w którym może zostać znaleziony przez interaktywny interpreter języka Python. Choć może to być katalog *python/Lib*, najprostszym rozwiązaniem będzie uruchomienie interpretera w tym samym katalogu, w którym zapisano plik.

W tym słowniku używany jest ranking od 1 do 5 jako sposób wyrażania stopnia polubienia danego filmu przez każdego z krytyków filmowych (oraz mnie). Niezależnie od metody wyrażania preferencji, konieczne jest odwzorowanie ich na wartości liczbowe. Jeśli zostałaby zbudowana witryna obsługująca zakupy, można użyć wartości 1 w celu wskazania, że ktoś w przeszłości kupił produkt, natomiast wartości 0, aby wskazać, że nic takiego się nie zdarzyło. W przypadku witryny, w której internauci głosują na artykuły informacyjne, wartości –1, 0 i 1 mogłyby reprezentować statusy „nie spodobał się", „nie oddano głosu" i „spodobał się" (tabela 2.1).

Tabela 2.1. Możliwe odwzorowania działań użytkowników na miary liczbowe

Bilety na koncerty		Zakupy internetowe		Rekomendowanie witryny	
kupiono	1	kupiono	2	spodobało się	1
nie kupiono	0	przeglądano	1	nie głosowano	0
		nie kupiono	0	nie spodobało się	–1

Użycie słownika jest wygodne w przypadku eksperymentowania z algorytmami, a także do celów ilustracyjnych. Słownik może być w prosty sposób wyszukiwany i modyfikowany. Po uruchomieniu interpretera języka Python należy sprawdzić kilka następujących poleceń:

```
c:\code\collective\chapter2> python
Python 2.4.1 (#65, Mar 30 2005, 09:13:57) [MSC v.1310 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>> from recommendations import critics
>> critics['Lisa Rose']['Kobieta w błękitnej wodzie']
2.5
>> critics['Toby']['Węże w samolocie']=4.5
>> critics['Toby']
{'Węże w samolocie':4.5,'Ja, ty i on':1.0}
```

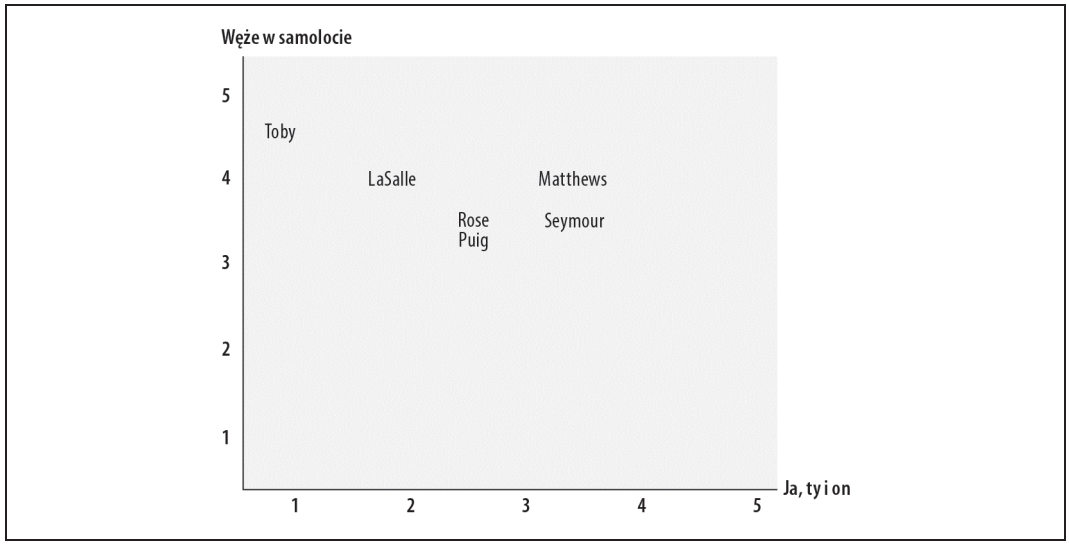
Choć w słowniku może się zmieścić duża liczba preferencji znajdujących się w pamięci, w przypadku bardzo dużych zbiorów danych prawdopodobnie będzie wskazane przechowywanie preferencji w bazie danych.

Znajdowanie podobnych użytkowników

Po zebraniu danych dotyczących rzeczy, które lubią ludzie, konieczne jest utworzenie metody pozwalającej określić, w jakim stopniu ludzie są podobni pod względem swoich gustów. W tym celu należy porównać każdą osobę z pozostałymi i obliczyć *miarę podobieństwa*. Istnieje kilka sposobów, które to umożliwiają. W tym podrozdziale zaprezentuję dwa systemy służące do obliczania miar podobieństwa: *odległość euklidesową* i *korelację Pearsona*.

Miara odległości euklidesowej

Bardzo prostym sposobem obliczenia miar podobieństwa jest użycie miary odległości euklidesowej, w przypadku której pozycje sklasyfikowane przez ludzi jako wspólne są używane jako osie wykresu. Następnie na wykresie można przedstawić osoby i sprawdzić, jak bardzo są do siebie podobne (rysunek 2.1).



Rysunek 2.1. Ludzie w przestrzeni preferencji

Na rysunku pokazano wykres z poszczególnymi osobami umieszczonymi w *przestrzeni preferencji*. Toby znalazł się na wykresie z wartością 4,5 na osi *Węże w samolocie* oraz wartością 1,0 na osi *Ja, ty i on*. Im bliżej siebie są dwie osoby w przestrzeni preferencji, tym bardziej podobne preferencje mają. Ze względu na to, że wykres jest dwuwymiarowy, w danej chwili można przyjrzeć się tylko dwóm rankingom. Jednakże w przypadku większych zestawów rankingów obowiązują identyczne zasady.

Aby obliczyć na wykresie odległość między krytykami Toby i LaSalle, różnicę na każdej osi należy podnieść do kwadratu, zsumować wyniki i obliczyć dla sumy pierwiastek kwadratowy. W języku Python można użyć funkcji `pow(n,2)` do potęgowania liczby. Obliczenie pierwiastka kwadratowego umożliwia funkcja `sqrt`:

```
>> from math import sqrt
>> sqrt(pow(5-4,2)+pow(4-1,2))
3.1622776601683795
```

Formuła ta oblicza odległość, która będzie mniejsza w przypadku osób bardziej do siebie podobnych. Wymagana jest jednak funkcja, która zapewnia wyższe wartości dla podobnych osób. W tym celu do funkcji należy dodać wartość 1 (dzięki temu nie zostanie wygenerowany błąd dzielenia przez zero), a następnie dokonać jej odwrócenia:

```
>> 1/(1+sqrt(pow(5-4,2)+pow(4-1,2)))
0.2402530733520421
```

Ta nowa funkcja zawsze zwraca wartość z przedziału od 0 do 1, gdzie wartość 1 oznacza, że dwie osoby mają identyczne preferencje. W celu utworzenia funkcji obliczającej podobieństwo można wszystko to ze sobą zestawić. Do pliku *recommendations.py* należy dodać następujący kod:

```
from math import sqrt

# zwracanie miary podobieństwa opartej na odległości dla pozycji person1 i person2
def sim_distance(prefs,person1,person2):
  # pobieranie listy wspólnych pozycji
  si={}
  for item in prefs[person1]:
```

```
    if item in prefs[person2]:
       si[item]=1

  # W przypadku braku wspólnych ocen zostanie zwrócona wartość 0.
  if len(si)==0: return 0

  # sumowanie kwadratów wszystkich różnic
  sum_of_squares=sum([pow(prefs[person1][item]-prefs[person2][item],2)
                      for item in prefs[person1] if item in prefs[person2]])
  return 1/(1+sum_of_squares)
```

W celu uzyskania miary podobieństwa funkcja ta może zostać wywołana z dwoma personaliami. W interpreterze języka Python należy wykonać następujące polecenia:

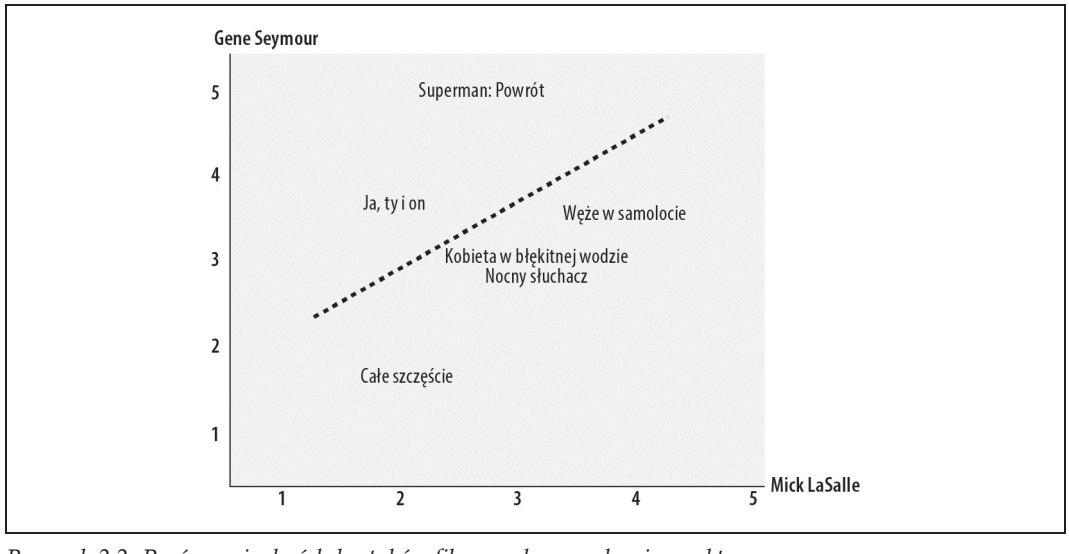
```
>>> reload(recommendations)
>>> recommendations.sim_distance(recommendations.critics,'Lisa Rose','Gene Seymour')
0.148148148148
```

Polecenia pozwalają uzyskać miarę podobieństwa dla Lisy Rose i Gene'a Seymoura. Warto spróbować użyć personaliów innych krytyków, aby przekonać się, czy uda się znaleźć tych, którzy są mniej lub bardziej do siebie podobni.

Miara korelacji Pearsona

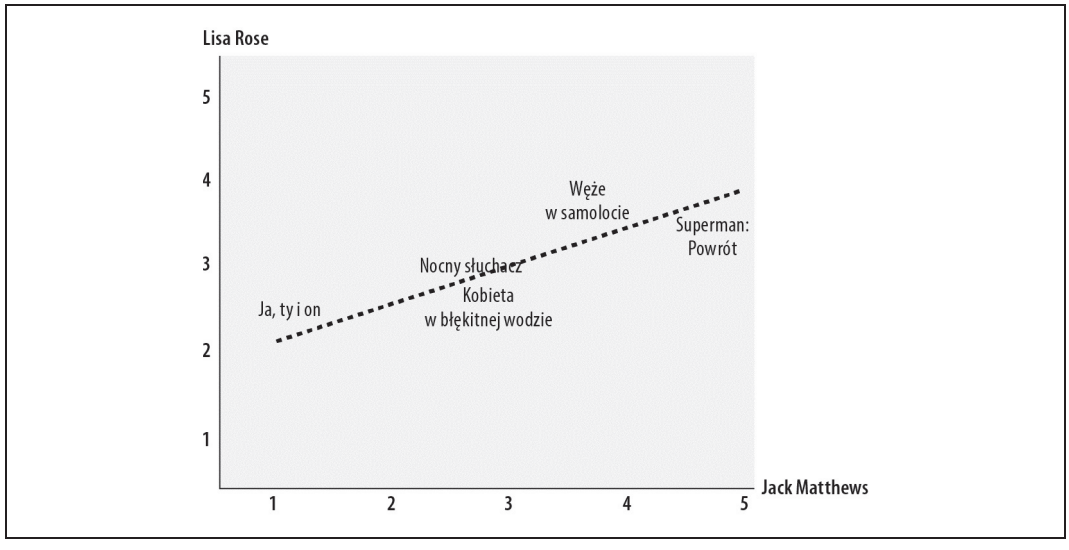
Trochę bardziej zaawansowany sposób określania podobieństwa między zainteresowaniami ludzi polega na użyciu współczynnika korelacji Pearsona. Współczynnik korelacji stanowi miarę tego, jak dobrze dwa zbiory danych dopasowane są do linii prostej. Choć formuła tej metody jest bardziej skomplikowana niż w przypadku miary odległości euklidesowej, zwykle daje lepsze wyniki w sytuacjach, w których dane są dobrze znormalizowane (np. gdy oceny filmowe krytyków są stale surowsze od przeciętnych).

Aby dokonać wizualizacji tej metody, można przedstawić na wykresie oceny dwóch krytyków (rysunek 2.2). Film *Superman: Powrót* otrzymał ocenę 3 od Micka LaSallego, a od Gene'a Seymoura ocenę 5, więc na wykresie został umiejscowiony przy wartości 3,5.



Rysunek 2.2. Porównanie dwóch krytyków filmowych na wykresie punktowym

Na wykresie można również zauważyć linię prostą. Jest ona nazywana *linią najlepszego dopasowania*, ponieważ przebiega tak blisko wszystkich pozycji jak to możliwe. Jeśli dla wszystkich filmów dwaj krytycy wystawili identyczne oceny, linia ta będzie przekątną, dotykając każdej pozycji na wykresie. Zapewniłoby to idealną miarę korelacji o wartości 1. W zilustrowanym przypadku krytycy nie zgadzają się co do kilku filmów, dlatego miara korelacji wynosi ok. 0,4. Na rysunku 2.3 pokazano przykład znacznie wyższej korelacji o wartości wynoszącej ok. 0,75.



Rysunek 2.3. Dwoje krytyków z wysoką miarą korelacji

Interesującym aspektem użycia miary korelacji Pearsona widocznym na rysunku jest to, że poprawia ona *inflację ocen*. Na rysunku Jack Matthews daje zwykle wyższe oceny niż Lisa Rose, ale linia w dalszym ciągu jest dopasowana, gdyż mają oni względnie podobne preferencje. Jeśli jeden z krytyków będzie się skłaniać do dawania wyższych ocen niż drugi, nadal może występować idealna korelacja, gdy zostanie zachowana stała różnica między ich ocenami. Opisana wcześniej miara odległości euklidesowej określi, że dwaj krytycy nie są do siebie podobni, ponieważ jeden jest zawsze surowszy od drugiego, nawet jeśli ich gusta są bardzo podobne. Zależnie od konkretnego zastosowania, takie działanie może być pożądane albo nie.

Kod miary korelacji Pearsona znajduje najpierw pozycje ocenione przez obu krytyków, a następnie oblicza sumę iloczynów ich ocen. Na końcu kod używa tych wyników do wyznaczenia współczynnika korelacji Pearsona, który pogrubiono w poniższym kodzie. W przeciwieństwie do miary odległości, ta formuła nie jest zbyt intuicyjna, ale pozwala określić, jak bardzo zmienne zmieniają się razem podzielone przez iloczyn określający, jak zmieniają się one osobno.

Aby zastosować tę formułę, należy utworzyć nową funkcję o takiej samej sygnaturze co funkcja `sim_distance` w pliku *recommendations.py*:

```
# zwracanie współczynnika korelacji Pearsona dla pozycji p1 i p2
def sim_pearson(prefs,p1,p2):
  # pobranie listy wzajemnie ocenianych pozycji
  si={}
  for item in prefs[p1]:
    if item in prefs[p2]: si[item]=1
```