Pięć kroków nauki o danych

Spędziliśmy dużo czasu przyglądając się wstępnym analizom danych, w tym opisując typy danych i sposoby podejścia do zestawów danych w zależności od ich typu. Tu skupimy się głównie na trzecim etapie eksploracji. Użyjemy pakietów pandas i matplotlib Pythona do eksploracji różnych zestawów danych.

Wprowadzenie do nauki o danych

Wiele osób pyta o największą różnicę między nauką o danych a analizą danych. Chociaż można argumentować, że nie ma między nimi żadnej różnicy, wielu będzie twierdziło, że są setki! Uważam, że niezależnie od tego, ile różnic jest między tymi dwoma terminami, największą jest to, że nauka o danych postępuje zgodnie z ustrukturyzowanym, krok po kroku procesem, który, gdy jest przestrzegany, zachowuje integralność wyników. Jak każde inne przedsięwzięcie naukowe, proces ten musi być przestrzegany, w przeciwnym razie analiza i wyniki są zagrożone analizą. Mówiąc prościej, przestrzeganie ścisłego procesu może znacznie ułatwić amatorskim naukowcom danych uzyskanie wyników szybciej niż w przypadku eksploracji danych bez wyraźnej wizji. Chociaż te kroki są lekcją przewodnią dla analityków amatorów, stanowią również podstawę dla wszystkich naukowców zajmujących się danymi, nawet tych na najwyższych szczeblach biznesowych i akademickich. Każdy analityk danych dostrzega wartość tych kroków i postępuje zgodnie z nimi w taki czy inny sposób.

Przegląd pięciu kroków

Pięć podstawowych kroków do wykonania analizy danych to:

1. Zadawanie ciekawego pytania

2. Uzyskiwanie danych

3. Eksploracja danych

4. Modelowanie danych

5. Komunikowanie i wizualizacja wyników

Najpierw spójrzmy na pięć kroków w odniesieniu do całościowego obrazu.

Zadaj ciekawe pytanie

To chyba mój ulubiony krok. Jako przedsiębiorca codziennie zadaję sobie (i innym) ciekawe pytania. Traktowałbym ten krok tak, jakbyś traktował sesję burzy mózgów. Zacznij spisywać pytania niezależnie od tego, czy uważasz, że dane, które pozwolą odpowiedzieć na te pytania, w ogóle istnieją. Powód tego jest dwojaki. Po pierwsze, nie chcesz zaczynać stronniczości nawet przed wyszukiwaniem danych. Po drugie, pozyskiwanie danych może wiązać się z wyszukiwaniem zarówno w miejscach publicznych, jak i prywatnych, a zatem może nie być zbyt proste. Możesz zadać pytanie i od razu powiedzieć sobie „Och, ale założę się, że nie ma danych, które mogłyby mi pomóc" i skreślić je z listy. Nie rób tego! Zostaw to na swojej liście.

Uzyskaj dane

Po wybraniu pytania, na którym chcesz się skoncentrować, nadszedł czas, aby przeszukać świat w poszukiwaniu danych, które mogą być w stanie odpowiedzieć na to pytanie. Jak wspomniano wcześniej, dane mogą pochodzić z różnych źródeł; więc ten krok może być bardzo kreatywny!

Przeglądaj dane

Kiedy już mamy dane, korzystamy z lekcji wyciągniętych z Części 2, Typy danych, i zaczynamy rozbijać typy danych, z którymi mamy do czynienia. To kluczowy krok w całym procesie. Po zakończeniu tego kroku analityk zazwyczaj spędza kilka godzin, ucząc się o domenie, używając kodu lub innych narzędzi do manipulowania danymi i ich eksploracji, i ma bardzo dobre wyczucie tego, co dane mogą próbować im przekazać.

Modeluj dane

Ten krok obejmuje wykorzystanie modeli statystycznych i uczenia maszynowego. Na tym etapie nie tylko dopasowujemy i wybieramy modele, ale także wszczepiamy matematyczne metryki walidacji w celu ilościowego określenia modeli i ich skuteczności.

Komunikuj się i wizualizuj wyniki

To prawdopodobnie najważniejszy krok. Choć może się to wydawać oczywiste i proste, umiejętność podsumowania wyników w przystępnej formie jest znacznie trudniejsza, niż się wydaje. Przyjrzymy się różnym przykładom przypadków, w których wyniki były słabo komunikowane i były bardzo dobrze wyświetlane. Skoncentrujemy się głównie na krokach 3, 4 i 5. Dlaczego pomijamy kroki 1 i 2? Chociaż pierwsze dwa kroki są niewątpliwie niezbędne dla procesu, generalnie poprzedzają systemy statystyczne i programistyczne. W dalszej części tej książki omówimy różne sposoby pozyskiwania danych, jednak w celu skupienia się na bardziej naukowych aspektach procesu, od razu zaczniemy od eksploracji.

Przeglądaj dane

Proces eksploracji danych nie jest zdefiniowany w prosty sposób. Wiąże się to z umiejętnością rozpoznawania różnych typów danych, przekształcania typów danych i używania kodu do systematycznego poprawiania jakości całego zestawu danych w celu przygotowania go do etapu modelowania. Aby jak najlepiej reprezentować i uczyć sztuki eksploracji, przedstawię kilka różnych zestawów danych i użyję pakietu pandas z pakietu Pythona do eksploracji danych. Po drodze natkniemy się na różne wskazówki i triki dotyczące obsługi danych. Istnieją trzy podstawowe pytania, które powinniśmy sobie zadać, gdy mamy do czynienia z nowym zbiorem danych, którego być może wcześniej nie widzieliśmy. Pamiętaj, że te pytania nie są początkiem i końcem nauki o danych; są to pewne wskazówki, których należy przestrzegać podczas eksploracji nowo uzyskanego zestawu danych.

Podstawowe pytania do eksploracji danych

Patrząc na nowy zbiór danych, niezależnie od tego, czy jest Ci znany, czy nie, ważne jest, aby użyć następujących pytań jako wskazówek do wstępnej analizy:

• Czy dane są uporządkowane, czy nie?

Sprawdzamy, czy dane są prezentowane w strukturze wiersz/kolumna. W większości dane będą prezentowane w sposób zorganizowany. U nas ponad 90% naszych przykładów zaczyna się od uporządkowanych danych. Niemniej jednak jest to najbardziej podstawowe pytanie, na które możemy odpowiedzieć, zanim zagłębimy się w naszą analizę. Ogólna zasada jest taka, że jeśli mamy niezorganizowane dane, chcemy je przekształcić w strukturę wiersz/kolumna. Na przykład we wcześniejszej części przyjrzeliśmy się sposobom przekształcania tekstu w strukturę wierszową/kolumnową, licząc liczbę słów/fraz.

• Co reprezentuje każdy rząd?

Gdy mamy już odpowiedź na to, jak dane są zorganizowane i teraz patrzymy na ładny zestaw danych oparty na wierszach/kolumnach, powinniśmy określić, co tak naprawdę reprezentuje każdy wiersz. Ten krok jest zwykle bardzo szybki i może pomóc znacznie szybciej spojrzeć na sprawy z innej perspektywy.

• Co reprezentuje każda kolumna?

Powinniśmy identyfikować każdą kolumnę według poziomu danych oraz tego, czy jest ilościowa/jakościowa, i tak dalej. Ta kategoryzacja może się zmieniać w miarę postępu naszej analizy, ale ważne jest, aby rozpocząć ten krok jak najwcześniej.

• Czy brakuje jakichkolwiek punktów danych?

Dane nie są doskonałe. Czasami może brakować danych z powodu błędu ludzkiego lub mechanicznego. Kiedy tak się dzieje, my, jako badacze danych, musimy podejmować decyzje, jak radzić sobie z tymi rozbieżnościami.

• Czy musimy wykonać jakieś przekształcenia na kolumnach?

W zależności od poziomu/typu danych, na którym znajduje się każda kolumna, może być konieczne wykonanie pewnych typów przekształceń. Na przykład, ogólnie rzecz biorąc, ze względu na modelowanie statystyczne i uczenie maszynowe chcielibyśmy, aby każda kolumna była numeryczna. Oczywiście użyjemy Pythona do wykonania wszelkich przekształceń.

Cały czas zadajemy sobie ogólne pytanie, co możemy wywnioskować ze wstępnych statystyk inferencyjnych? Chcemy być w stanie zrozumieć nasze dane nieco bardziej niż wtedy, gdy je znaleźliśmy.

Zestaw danych - Yelp

Pierwszym zbiorem danych, któremu się przyjrzymy, jest publiczny zbiór danych udostępniony przez witrynę z recenzjami restauracji, Yelp. Wszystkie informacje umożliwiające identyfikację osoby zostały usunięte. Przeczytajmy najpierw dane, jak pokazano tutaj:

```
import pandas as pd

yelp_raw_data = pd.read_csv("yelp.csv")

yelp_raw_data.head()
```

Krótkie podsumowanie tego, co robi poprzedni kod:

• Zaimportuj pakiet pandy i nazwij go jako pd.

• Czytaj w .csv z sieci; wywołanie to yelp_raw_data.

• Spójrz na początek danych (tylko kilka pierwszych wierszy).

	business_id	date	review_id	stars	text	type	user_id	cool	useful	funny
0	9yKzy9PApeiPPOUJEtnvkg	2011-01-26	fWKvX83p0-ka4JS3dc6E5A	5	My wife took me here on my birthday for breakf...	review	rLtl8ZkDX5vH5nAx9C3q5Q	2	5	0
1	ZRUwVLyzEJq7VAhDhYow	2011-07-27	IjZ33sJrzXqU-0X6U9NxyA	5	I have no idea why some people give bad review...	review	Ga7KyELDd3Yb1V6aivbIuQ	0	0	0
2	6oRAC4uyJCsJl1X0WZpVSA	2012-06-14	IESLBzqUCLdSzSqm0eCSxQ	4	love the gyro plate. Rice is so good and I als...	review	0hT2KtfLiobPvh6cDC8JQg	0	1	0
3	_1QQZuf4zZOyFCvXc0o6Vg	2010-05-27	G-WvGaISbqqaMHlNnByodA	5	Rosie, Dakota, and I LOVE Chaparral Dog Park!t...	review	uZetl9T0NcROGOyFfughhg	1	2	0
4	6ozycU1RpktNG2-1BroVtw	2012-01-05	1uJFq2r5QfJG_6ExMRCaGw	5	General Manager Scott Petello is a good egg!!t...	review	vYmM4KTsC8ZfQBg-j5MWkw	0	0	0

Czy dane są uporządkowane, czy nie?

• Ponieważ mamy ładną strukturę wierszy/kolumn, możemy wywnioskować, że te dane wydają się dość uporządkowane.

Co reprezentuje każdy wiersz?

• Wydaje się dość oczywiste, że każdy wiersz reprezentuje użytkownika oceniającego firmę. Następną rzeczą, którą powinniśmy zrobić, jest zbadanie każdego wiersza i oznaczenie go typem danych, które zawiera. W tym momencie możemy również użyć Pythona, aby dowiedzieć się, jak duży jest nasz zbiór danych. Aby to sprawdzić, możemy użyć jakości kształtu Dataframe, jak pokazano:

yelp_raw_data.shape

(10000,10)

• Mówi nam, że ten zbiór danych ma 10000 wierszy i 10 kolumn. Innym sposobem na powiedzenie tego jest to, że ten zbiór danych zawiera 10 000 obserwacji i 10 cech.

Co reprezentuje każda kolumna?

Zauważ, że mamy 10 kolumn:

• business_id: jest to prawdopodobnie unikalny identyfikator firmy, której dotyczy opinia. Byłoby to na poziomie nominalnym, ponieważ nie ma naturalnego porządku tego identyfikatora.

• data: prawdopodobnie jest to data opublikowania recenzji. Zauważ, że wydaje się, że dotyczy tylko dnia, miesiąca i roku. Chociaż czas jest zwykle uważany za ciągły, ta kolumna prawdopodobnie zostałaby uznana za dyskretną i na poziomie porządkowym ze względu na naturalny porządek dat.

• review_id: Jest to prawdopodobnie unikalny identyfikator recenzji, którą reprezentuje każdy post. Byłoby to na poziomie nominalnym, ponieważ znowu nie ma naturalnego porządku tego identyfikatora.

• stars: Po szybkim spojrzeniu (nie martw się, wkrótce przeprowadzimy dalszą analizę), widzimy, że jest to uporządkowana kolumna, która przedstawia ocenę końcową restauracji przez recenzenta. To jest uporządkowane i jakościowe; więc to jest na poziomie porządkowym.

• text: jest to prawdopodobnie nieprzetworzony tekst, który napisał każdy recenzent. Jak w przypadku większości tekstów, umieszczamy to na poziomie nominalnym.

• type: W pierwszych pięciu kolumnach widzimy tylko słowo przegląd. Może to być kolumna, która wskazuje, że każdy wiersz jest recenzją, co oznacza, że może istnieć inny typ wiersza niż recenzja. Przyjrzymy się temu później. Umieszczamy to na poziomie nominalnym.

• user_id: jest to prawdopodobnie unikalny identyfikator użytkownika, który pisze recenzję. Podobnie jak w przypadku innych unikalnych identyfikatorów, umieszczamy te dane na poziomie nominalnym.

Zauważ, że po przejrzeniu wszystkich kolumn i stwierdzeniu, że wszystkie dane są albo na poziomie porządkowym, albo na poziomie nominalnym, musimy spojrzeć na następujące rzeczy. Nie jest to rzadkie, ale warto o tym wspomnieć.

Czy brakuje jakichś punktów danych?

• Wykonaj operację isnull. Na przykład, jeśli twoja ramka danych nazywa się awesome_dataframe, wypróbuj polecenie python awesome_dataframe. isnull().sum() co pokaże liczbę braków danych w każdej kolumnie.

Czy musimy wykonać jakieś przekształcenia na kolumnach?

• W tym momencie szukamy kilku rzeczy. Na przykład, czy będziemy musieli zmienić skalę niektórych danych ilościowych, czy też musimy utworzyć zmienne fikcyjne dla zmiennych jakościowych? Ponieważ ten zbiór danych zawiera tylko kolumny jakościowe, możemy skupić się tylko na przekształceniach w skali porządkowej i nominalnej.

Zanim zaczniemy, przejrzyjmy krótką terminologię dotyczącą pand, modułu eksploracji danych Pythona.

Ramki danych

Kiedy czytamy w zbiorze danych, Pandas tworzy niestandardowy obiekt o nazwie Dataframe. Pomyśl o tym jako o wersji arkusza kalkulacyjnego w Pythonie (ale o wiele lepiej). W tym przypadku zmienna yelp_raw_data jest ramką danych. Aby sprawdzić, czy tak jest w Pythonie, wpisz następujący kod:

typ(yelp_raw_data)

pandas.core.frame.DataFrame

Ramki danych mają charakter dwuwymiarowy, co oznacza, że są zorganizowane w strukturę wierszy/kolumn, tak jak arkusz kalkulacyjny. Główną zaletą korzystania z Dataframes w porównaniu z, powiedzmy, oprogramowaniem do arkuszy kalkulacyjnych, jest to, że Dataframe może obsługiwać znacznie większe dane niż większość popularnych programów do obsługi arkuszy kalkulacyjnych. Jeśli znasz język R, możesz rozpoznać słowo Dataframe. To dlatego, że nazwa została faktycznie zapożyczona z języka! Ponieważ większość danych, którymi będziemy się zajmować jest zorganizowana, Dataframes są prawdopodobnie najczęściej używanym obiektem w pandach, ustępując tylko obiektowi Series.

Series

Obiekt Series to po prostu Dataframe, ale tylko z jednym wymiarem. Zasadniczo jest to lista punktów danych. Każda kolumna Dataframe jest uważana za obiekt serii. Sprawdźmy to. Pierwszą rzeczą, którą musimy zrobić, to pobrać pojedynczą kolumnę z naszego Dataframe; zazwyczaj używamy tak zwanej notacji nawiasowej. Oto przykład:

yelp_raw_data['business_id'] # pobiera pojedynczą kolumnę Dataframe

Wymienimy kilka pierwszych i kilka ostatnich wierszy:

0 9yKzy9PApeiPPOUJetnvkg

1 ZRJwVLyzEJq1VAihDhYiow

2 6oRAC4uyJCsJl1X0WZpVSA

3 _1QQZuf4zZOyFCvXc0o6Vg

4 6ozycU1RpktNG2-1BroVtw

5 -yxfBYGB6SEqszmxJxd97A

6 zp713qNhx8d9KCJJnrw1xA

Użyjmy funkcji type, aby sprawdzić, czy ta kolumna jest serią:

type(yelp_raw_data['business_id'])

pandas.core.series.Series

Wskazówki dotyczące eksploracji danych jakościowych

Korzystając z tych dwóch obiektów Pandy, zacznijmy przeprowadzać wstępną eksplorację danych. W przypadku danych jakościowych przyjrzymy się konkretnie poziomom nominalnym i porządkowym.

Kolumny poziomu nominalnego

Ponieważ jesteśmy na poziomie nominalnym, przypomnijmy sobie, że na tym poziomie dane są jakościowe i są opisane czysto z nazwy. W tym zbiorze danych odnosi się to do business_id, review_ id, tekstu, typu i user_id. Użyjmy Pand, aby zanurkować nieco głębiej, jak pokazano tutaj:

yelp_raw_data['business_id'].describe()

count 10000

unique 4174

top JokKtdXU7zXHcr20Lrk29A

freq 37

Funkcja opisu da nam kilka szybkich statystyk na temat kolumny, której nazwę wpisujemy w cudzysłów. Zwróć uwagę, jak Pandas automatycznie rozpoznała, że business_id to kolumna jakościowa i podała nam statystyki, które mają sens. Gdy opis zostanie wywołany w kolumnie jakościowej, zawsze otrzymamy następujące cztery pozycje:

• count: Ile wartości jest wpisanych

• unikalny: Ile unikalnych wartości jest wypełnionych

• u góry: nazwa najczęściej występującego elementu w zbiorze danych

• freq: jak często w zbiorze danych pojawia się najczęstszy element

Na poziomie nominalnym zwykle szukamy kilku rzeczy, które sygnalizowałyby transformację:

• Czy mamy rozsądną liczbę (zwykle poniżej 20) unikalnych pozycji?

• Czy ta kolumna jest wolna od tekstu?

• Czy ta kolumna jest całkowicie unikalna we wszystkich wierszach?

Tak więc dla kolumny business_id mamy 10000. Nie daj się jednak zwieść! Nie oznacza to, że recenzujemy tu 10 000 firm. Oznacza to po prostu, że z 10 000 wierszy opinii kolumna business_id jest wypełniana 10 000 razy. Kolejny kwalifikator, unikalny, informuje nas, że w tym zbiorze danych sprawdzanych jest 4174 unikalnych firm. Najczęściej recenzowaną firmą jest biznes JokKtdXU7zXHcr20Lrk29A, który był recenzowany 37 razy.

yelp_raw_data['review_id'].describe()

count 10000

unique 10000

top eTa5KD-LTgQv6UT1Zmijmw

freq 1

Liczymy 10000, a unikatowe 10000. Zastanów się przez chwilę, czy to ma sens? Zastanów się, co reprezentuje każdy wiersz i co reprezentuje ta kolumna. (tu wstaw piosenkę o niebezpieczeństwie)

Oczywiście, że tak! Każdy wiersz tego zbioru danych ma reprezentować jedną, niepowtarzalną recenzję firmy, a ta kolumna ma służyć jako unikalny identyfikator recenzji; więc sensowne jest, aby kolumna review_id zawierała 10000 unikalnych elementów. Dlaczego więc eTa5KD-LTgQv6UT1Zmijmw jest najczęstszą recenzją? To tylko losowy wybór z 10 000 i nic nie znaczy.

yelp_raw_data['text'].describe()

count 10000

unique 9998

top This review is for the chain in general. The l &helip;

freq 2

Ta kolumna, która reprezentuje rzeczywisty tekst, który napisali ludzie, jest interesująca. Wyobrażamy sobie, że powinno to być również podobne do review_id, ponieważ cały tekst powinien być unikalny, ponieważ byłoby dziwne, gdyby dwie osoby napisały dokładnie to samo; ale mamy dwie recenzje z dokładnie tym samym tekstem! Poświęćmy chwilę, aby dowiedzieć się więcej o filtrowanie ramek danych, aby dokładniej to zbadać.

Filtrowanie w Pandas

Porozmawiajmy trochę o tym, jak działa filtrowanie. Filtrowanie wierszy na podstawie określonych kryteriów w Pandas jest dość łatwe. W Dataframe, jeśli chcemy odfiltrować wiersze na podstawie pewnych kryteriów wyszukiwania, musimy przejść wiersz po wierszu i sprawdzić, czy wiersz spełnia ten konkretny warunek. Pandas radzi sobie z tym, przekazując serię Prawd i Fałszów (Boolean). Dosłownie przekazujemy do Dataframe listę danych Prawda i Fałsz, które oznaczają:

• Prawda: ten wiersz spełnia warunek

• Fałsz: ten wiersz nie spełnia warunku

Więc najpierw ustalmy warunki. W kolejnych wierszach kodu chwycę tekst, który występuje dwukrotnie:

duplicate_text = yelp_raw_data['text'].describe()['top']

Oto fragment tekstu:

„Ta recenzja jest ogólnie dla sieci. Lokalizacja, do której się udaliśmy, jest nowa, więc nie ma jej jeszcze na Yelp. Gdy już będzie, umieszczę tam również tę recenzję …”.

Od razu możemy się domyślać, że może to być jedna osoba, która poszła zrecenzować dwie firmy należące do tej samej sieci i napisała dokładnie tę samą recenzję. Jednak to tylko przypuszczenie.

Zmienna duplikat_tekstu jest typu string

Teraz, gdy mamy ten tekst, użyjmy magii, aby stworzyć tę serię prawdy i fałszu:

text_is_the_duplicate = yelp_raw_data['text'] == duplicate_text

Od razu możesz być zdezorientowany. To, co tutaj zrobiliśmy, to pobranie kolumny tekstowej ramki danych i porównanie jej z ciągiem tekstowym duplikat_tekst. Jest to dziwne, ponieważ wydaje się, że porównujemy listę 10 000 elementów z pojedynczym ciągiem. Oczywiście odpowiedź powinna być fałszywa, prawda? Seria Pand ma bardzo interesującą cechę polegającą na tym, że jeśli porównasz Serie do obiektu, zwróci kolejną Serię Boole'ów o tej samej długości, gdzie każda prawda i fałsz jest odpowiedzią na pytanie, czy ten element jest taki sam jak element, do którego go porównujesz? Bardzo przydatne!

type(text_is_the_duplicate) # it is a Series of Trues and Falses

text_is_the_duplicate.head() # shows a few Falses out of the Series

W Pythonie możemy dodawać i odejmować prawdę i fałsz, tak jakby były odpowiednio 1 i 0. Na przykład True + False – True + False + True == 1. Możemy więc zweryfikować, czy ta seria jest poprawna, dodając wszystkie wartości. Ponieważ tylko dwa z tych wierszy powinny zawierać zduplikowany tekst, suma serii powinna wynosić tylko 2, co jest prawdą! Jest to pokazane w następujący sposób:

sum(text_is_the_duplicate) # == 2

Teraz, gdy mamy już naszą serię wartości logicznych, możemy przekazać ją bezpośrednio do naszej ramki danych, używając notacji nawiasów i uzyskać nasze przefiltrowane wiersze, jak pokazano na ilustracji:

filtered_dataframe = yelp_raw_data[text_is_the_duplicate]

the filtered Dataframe

filtered_dataframe

Wygląda na to, że nasze podejrzenia były słuszne i jedna osoba tego samego dnia przekazała dokładnie tę samą recenzję dwóm różnym identyfikatorom firmy, prawdopodobnie należącym do tej samej sieci. Przejdźmy dalej do reszty naszych kolumn:

yelp_raw_data['type'].describe()

count 10000

unique 1

top review

freq 10000

Pamiętasz tę kolumnę? Okazuje się, że wszystkie są dokładnie tego samego typu, a mianowicie przegląd.

yelp_raw_data['user_id'].describe()

count 10000

unique 6403

top fczQCSmaWF78toLEmb0Zsw

freq 38

Podobnie jak w kolumnie business_id, wszystkie 10000 wartości są wypełniane z 6403 unikalnymi użytkownikami i jednym użytkownikiem sprawdzającym 38 razy! W tym przykładzie nie będziemy musieli wykonywać żadnych przekształceń

Kolumny poziomu porządkowego

Jeśli chodzi o kolumny porządkowe, patrzymy na datę i gwiazdy. Dla każdej z tych kolumn przyjrzyjmy się, co zwraca metoda opisu:

yelp_raw_data['stars'].describe()

count 10000.000000

mean 3.777500

std 1.214636

min 1.000000

25% 3.000000

50% 4.000000

75% 5.000000

max 5.000000

Och! Mimo że ta kolumna jest porządkowa, metoda opisu zwracała statystyki, których moglibyśmy oczekiwać dla kolumny ilościowej. Dzieje się tak, ponieważ oprogramowanie widziało wiele liczb i po prostu założyło, że chcemy mieć statystyki, takie jak średnia lub minimalna i maksymalna. To nie jest problem. Użyjmy metody o nazwie value_counts, aby zobaczyć liczbę

yelp_raw_data['stars'].value_counts()

4 3526

5 3337

3 1461

2 927

1 749

Metoda value_counts zwróci rozkład wartości dla dowolnej kolumny. W tym przypadku widzimy, że ocena w postaci gwiazdek 4 jest najbardziej powszechna, z 3526 wartościami, tuż za nią znajduje się ocena 5. Możemy również wykreślić te dane, aby uzyskać ładny obraz. Najpierw posortujmy według ocen, a następnie użyjmy gotowej metody kreślenia, aby utworzyć wykres słupkowy.

```
dates = yelp_raw_data['stars'].value_counts()

dates.sort()

dates.plot(kind='bar')
```

<img src="images/bar.PNG">

Ludzie zdecydowanie częściej dają dobre oceny w gwiazdkach niż złe! Możemy postępować zgodnie z tą procedurą dla kolumny daty. Zostawię cię, abyś sam spróbował. Na razie spójrzmy na nowy zbiór danych.

Zestaw danych 2 - tytaniczny

Zbiór danych Titanica zawiera próbkę ludzi, którzy byli na Titanicu, kiedy uderzył w górę lodową w 1912 roku. Przejdźmy dalej i zaimportujmy go, jak pokazano tutaj:

```
titanic = pd.read_csv('short_titanic.csv')

titanic.head()
```

	Survived	Pclass	Name	Sex	Age
0	0	3	Braund, Mr. Owen Harris	male	22
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38
2	1	3	Heikkinen, Miss. Laina	female	26
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35
4	0	3	Allen, Mr. William Henry	male	35

Ta ramka danych ma zwykle więcej kolumn; jednak w naszym przykładzie skupimy się tylko na podanych kolumnach. Te dane są zdecydowanie zorganizowane w strukturę wierszy/kolumn, podobnie jak większość danych w arkuszach kalkulacyjnych. Rzućmy okiem na jego rozmiar, jak pokazano tutaj:

```
titanic.shape

# (891, 5)
```

Mamy więc 891 wierszy i 5 kolumn. Każdy wiersz wydaje się reprezentować jednego pasażera na statku, a jeśli chodzi o kolumny, poniższa lista mówi nam, co one oznaczają:

• Przeżył: Jest to zmienna binarna, która wskazuje, czy pasażer przeżył wypadek (1, jeśli przeżył, 0, jeśli zginął). Byłoby to prawdopodobnie na poziomie nominalnym, ponieważ są tylko dwie opcje.

• Pclass: jest to klasa, w której podróżował pasażer (3 dla trzeciej klasy itd.). To jest na poziomie porządkowym.

• Imię i nazwisko: to imię i nazwisko pasażera, i to zdecydowanie na poziomie nominalnym.

• Płeć: Wskazuje płeć pasażera. Jest na poziomie nominalnym.

• Wiek: Ten jest trochę trudny. Zapewne można umieścić wiek albo na poziomie jakościowym, albo ilościowym, jednak uważam, że wiek należy do stanu ilościowego, a więc do poziomu relacyjnego.

Jeśli chodzi o przekształcenia, zwykle chcemy, aby wszystkie kolumny były numeryczne, niezależnie od ich stanu jakościowego. Oznacza to, że Imię i Płeć będą musiały zostać jakoś przekonwertowane na kolumny liczbowe. W przypadku Płeć możemy zmienić kolumnę tak, aby zawierała 1, jeśli pasażer była kobietą i 0, jeśli był mężczyzną. Użyjmy Pand, aby dokonać zmiany. Będziemy musieli zaimportować inny moduł Pythona, zwany numpy lub Python numeryczny, jak pokazano na ilustracji:

```
import numpy as np

titanic['Sex'] = np.where(titanic['Sex']=='female', 1, 0)
```

Metoda np.where obejmuje trzy rzeczy:

• Lista wartości logicznych (prawda lub fałsz)

• Nowa wartość

• Wartość zapasowa

Metoda zastąpi wszystkie true pierwszą wartością (w tym przypadku 1) i false drugą wartością (w tym przypadku 0), pozostawiając nam nową kolumnę liczbową, która reprezentuje to samo, co oryginalna kolumna Sex.

```
titanic['Sex']

# 0 0

# 1 1

# 2 1

# 3 1

# 4 0

# 5 0

# 6 0

# 7 0
```

Użyjmy skrótu i opiszmy wszystkie kolumny jednocześnie, jak pokazano:

```
titanic.describe()
```

	Survived	Pclass	Sex	Age
count	891.000000	891.000000	891.000000	714.000000
mean	0.383838	2.308642	0.352413	29.699118
std	0.486592	0.836071	0.477990	14.526497
min	0.000000	1.000000	0.000000	0.420000
25%	0.000000	2.000000	0.000000	20.125000
50%	0.000000	3.000000	0.000000	28.000000
75%	1.000000	3.000000	1.000000	38.000000
max	1.000000	3.000000	1.000000	80.000000

Zwróć uwagę, jak nasze kolumny jakościowe są traktowane jako ilościowe; jednak szukam czegoś nieistotnego dla typu danych. Zwróć uwagę na wiersz liczenia: Przeżyli, Pklasa i Płeć mają 891 wartości (liczba wierszy), ale Wiek ma tylko 714 wartości. Niektórych brakuje! Aby dwukrotnie zweryfikować, użyjmy funkcji Pandy, zwanych isnull i sum, jak pokazano:

titanic.isnull().sum()

Survived 0

Pclass 0

Name 0

Sex 0

Age 177

To pokaże nam liczbę braków danych w każdej kolumnie. Tak więc Wiek jest jedyną kolumną z brakującymi wartościami, z którą należy się uporać. Kiedy masz do czynienia z brakującymi wartościami, zwykle masz dwie opcje:

• Upuść wiersz z brakującą wartością

• Spróbuj go wypełnić

Upuszczenie rzędu to łatwy wybór; jednak ryzykujesz utratę cennych danych! Na przykład w tym przypadku mamy 177 brakujących wartości wieku (891-714), co stanowi prawie 20% danych. Aby uzupełnić dane, możemy albo wrócić do podręczników historii, znaleźć każdą osobę po kolei i wpisać jej wiek, albo możemy wpisać wiek wartością zastępczą. Uzupełnijmy każdą brakującą wartość w kolumnie Wiek ogólnym średnim wiekiem osób w zbiorze danych. W tym celu zastosujemy dwie nowe metody, zwane mean i fillna. Używamy isnull, aby powiedzieć nam, które wartości są null, oraz funkcji średniej, aby uzyskać średnią wartość kolumny Age. fillna to metoda Pandy, która zastępuje wartości null podaną wartością.

print sum(titanic['Age'].isnull()) # == 177 missing values

average_age = titanic['Age'].mean() # get the average age

titanic['Age'].fillna(average_age, inplace = True) #use the fillna

method to remove null values

print sum(titanic['Age'].isnull()) # == 0 missing values

Skończyliśmy! Zamieniliśmy każdą wartość na 26,69, czyli średni wiek w zbiorze danych.

titanic.isnull().sum()

Survived 0

Pclass 0

Name 0

Sex 0

Age 0

Świetny! Niczego nie brakuje i nie musieliśmy usuwać żadnych rzędów.

titanic.head()

	Survived	Pclass	Name	Sex	Age
0	0	3	Braund, Mr. Owen Harris	0	22
1	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38
2	1	3	Heikkinen, Miss. Laina	1	26
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	35
4	0	3	Allen, Mr. William Henry	0	35

W tym momencie możemy zacząć nieco bardziej skomplikować nasze pytania. Na przykład, jaki jest średni wiek kobiety lub mężczyzny? Aby odpowiedzieć na to pytanie, możemy filtrować według płci i wziąć średni wiek. Pandas ma w tym celu wbudowaną funkcję o nazwie groupby, jak pokazano poniżej:

titanic.groupby('Sex')['Age'].mean()

Oznacza to pogrupowanie danych według kolumny Płeć, a następnie podanie średniej wieku dla każdej grupy. Daje nam to następujący wynik:

Sex

0 30.505824

1 28.216730

Zadamy więcej tych trudnych i złożonych pytań i będziemy mogli na nie odpowiedzieć za pomocą Pythona i statystyk.

Podsumowanie

Chociaż jest to tylko nasze pierwsze spojrzenie na eksplorację danych, nie martw się — to zdecydowanie nie ostatni raz, kiedy wykonamy te kroki w celu nauki i eksploracji danych. Od teraz, za każdym razem, gdy przyjrzymy się nowej porcji danych, wykorzystamy nasze etapy eksploracji, aby przekształcić, podzielić i ujednolicić nasze dane. Kroki opisane w tym rozdziale, choć są jedynie

wskazówkami, stanowią standardową praktykę, którą każdy analityk danych może stosować w swojej pracy. Kroki można również zastosować do dowolnego zestawu danych, który wymaga analizy. Szybko zbliżamy się do części książki, która zajmuje się modelami statystycznymi, probabilistycznymi i uczeniem maszynowym. Zanim naprawdę będziemy mogli wskoczyć do tych modeli, musimy przyjrzeć się podstawom matematyki. W następnej części przyjrzymy się niektórym matematyce niezbędnej do wykonania niektórych bardziej skomplikowanych operacji w modelowaniu, ale nie martw się - matematyka wymagana do tego proces jest minimalny i przejdziemy go krok po kroku.