O'REILLY®

Potoki danych

Leksykon kieszonkowy

Przenoszenie i przetwarzanie danych na potrzeby ich analizy

Helion

James Densmore

Tytuł oryginału: Data Pipelines Pocket Reference: Moving and Processing Data for Analytics

Tłumaczenie: Robert Górczyński

ISBN: 978-83-8322-338-4

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

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

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

Printed in Poland.

Spis treści

Rozdział 3.
Najczęściej spotykane wzorce potoków danych

Nawet dla doświadczonego inżyniera danych projektowanie nowego potoku danych za każdym razem będzie nie lada wyzwaniem. Jak już wyjaśniłem w rozdziale 2., poszczególne źródła danych i infrastruktura wiążą się zarówno z wyzwaniami, jak i możliwościami. Ponadto potoki danych są budowane w różnych celach oraz z uwzględnieniem różnych ograniczeń. Czy dane muszą być przetwarzane niemalże w czasie rzeczywistym? Czy mogą być uaktualniane raz dziennie? Czy będą modelowane do użycia w panelach, czy może jako dane wejściowe dla modeli uczenia maszynowego (ang. *machine learning*, ML)?

Na szczęście w potokach danych istnieją pewne najczęściej spotykane wzorce, które okazały się skuteczne i możliwe do zastosowania w wielu różnych przypadkach. W rozdziale tym zamierzam zdefiniować te wzorce. Natomiast w kolejnych rozdziałach przedstawię implementację utworzonych na ich podstawie potoków danych.

ETL i ELT

Do prawdopodobnie najbardziej znanych wzorców zaliczamy ETL i jego nowoczesny wariant ETL. Oba te wzorce są powszechnie stosowane w hurtowniach danych i rozwiązaniach typu Business Intelligence. W ostatnich latach stanowiły inspirację dla wzorców potoków danych przeznaczonych do celów naukowych oraz w modelach uczenia maszynowego stosowanych w produkcji. Te wzorce są na tyle dobrze znane, że wiele osób używa wymienionych pojęć jako synonimów dla potoków danych zamiast jako wzorców, które mogą być stosowane w wielu potokach danych.

Gdy weźmie się pod uwagę korzenie wzorców ETL i ELT w hurtowniach danych, najłatwiej będzie je opisać w pewnym kontekście. Takie

podejście zastosowałem w rozdziale. W jego dalszej części wyjaśnię, jak omawiane wzorce są używane w konkretnych sytuacjach.

Wzorce ETL i ELT to podejścia w zakresie przetwarzania danych, którego celem jest przekazanie danych do hurtowni danych oraz zapewnienie użyteczności tych danych na potrzeby narzędzi analizy i raportowania. Różnica między tymi wzorcami kryje się w kolejności dwóch ostatnich kroków (przetwarzaniu i wczytywaniu danych). Implikacje projektowe między tymi wzorcami są poważne i wyjaśnię je w rozdziale. Zacznę jednak od omówienia poszczególnych kroków w ETL i ELT.

Wyodrębnianie (ang. *extract*) polega na zebraniu danych z różnych źródeł w celu ich przygotowania do wczytania i przekształcenia. W rozdziale 2. dokładniej omówiłem różnorodność źródeł i metod zbierania danych.

Wczytywanie (ang. *load*) dostarcza dane niezmodyfikowane (w przypadku wzorca ELT) lub w pełni przetworzone (w przypadku wzorca ETL) do ich miejsca docelowego. Ostatecznym wynikiem jest wczytanie danych do hurtowni danych, jeziora danych bądź też do innego miejsca.

Przekształcanie (ang. *transform*) to etap, w którym niezmodyfikowane dane z poszczególnych systemów źródłowych zostają połączone i sformatowane w taki sposób, aby stały się użyteczne podczas analizy, do wykorzystania w narzędziach wizualizacji bądź też w innych celach, do których został utworzony potok danych. Na tym etapie naprawdę wiele się dzieje, niezależnie od wybranego wzorca (ETL lub ELT). Szczegółowe omówienie tego etapu znajdziesz w rozdziale 6.

ROZDZIELENIE ETAPÓW WYODRĘBNIANIA I WCZYTYWANIA

Połączenie etapów wyodrębniania i wczytywania jest często określane mianem **pobierania danych**. Szczególnie we wzorcu ELT i w jego podwzorcu EtLT (zwróć uwagę na małą literę *t*), zdefiniowanym w dalszej części rozdziału, wyodrębnianie i wczytywanie są często ściśle ze sobą powiązane i zdefiniowane razem we frameworkach oprogramowania. Jednak podczas projektowania potoków danych najlepiej będzie uznawać je za dwa zupełnie oddzielne etapy, ze względu na złożoność operacji wyodrębniania i wczytywania w różnych systemach i infrastrukturze.

W rozdziałach 4. i 5. znajdziesz znacznie dokładniejsze omówienie technik pobierania danych, a także przykłady ich implementacji za pomocą najczęściej używanych frameworków.

Pojawienie się ELT po ETL

Przez dekady ETL był złotym standardem wzorców potoków danych. Wprawdzie nadal jest stosowany, ale ostatnio pojawił się jego nowy wariant, ELT. Dlaczego? Zanim powstały nowoczesne hurtownie danych, głównie w chmurze (zob. rozdział 2.), zespoły pracujące z danymi nie miały dostępu do magazynów danych o pojemnościach i możliwościach niezbędnych do obsługi ogromnej ilości danych niezmodyfikowanych i ich przekształcania na użyteczne modele danych. Ponadto ówczesne hurtownie danych miały zwykle postać opartych na rekordach baz danych, które świetnie sprawdzały się w rozwiązaniach wymagających obsługi transakcji, ale nie w przypadku charakteryzujących się ogromnym wolumenem danych zapytań będących codziennością w trakcie analizy danych. Dlatego też dane były najpierw wyodrębniane z systemów źródłowych, a następnie przekształcane w innych systemach, zanim wreszcie były wczytywane do hurtowni danych w celu ostatecznego modelowania danych i ich używania przez analityków oraz w narzędziach wizualizacji.

Większość obecnych hurtowni danych jest tworzona w postaci wysoce skalowalnych, kolumnowych baz danych, które mogą zarówno przechowywać, jak i przekształcać ogromne zbiory danych w niezwykle efektywny sposób. To się zmieniło dzięki efektywnym systemom wejścia-wyjścia kolumnowych baz danych, kompresji danych oraz możliwości rozpraszania danych i zapytań między wieloma węzłami współpracującymi ze sobą podczas przetwarzania danych. Teraz lepiej jest skoncentrować się na wyodrębnianiu danych i ich wczytywaniu do hurtowni danych, w której następnie można przeprowadzić niezbędne przekształcenia i tym samym zakończyć działanie potoku danych.

Nie wolno pominąć wpływu różnicy między hurtowniami danych bazującymi na rekordach i kolumnach. Na rysunku 3.1 pokazałem sposób przechowywania rekordów na dysku w opartej na rekordach bazie

danych, takiej jak MySQL lub PostgreSQL. Każdy rekord bazy danych jest przechowywany na dysku w jednym bloku lub w ich większej liczbie, w zależności od wielkości rekordu. Jeżeli rekord jest mniejszy niż blok bądź wielkość rekordu nie dzieli się równo przez wielkość bloku, wówczas pewna ilość miejsca na dysku pozostaje niewykorzystana.

OrderId	CustomerId	ShippingCountry	OrderTotal
1	1258	US	55,25
2	5698	AUS	125,36
3	2265	US	776,95
4	8954	CA	32,16

Blok 1	1, 1258, US, 55,25
Blok 2	2, 5698, AUS, 125,36
Blok 3	3, 2265, US, 776,95
Blok 4	4, 8954, CA, 32,16

Rysunek 3.1. Tabela przechowywana w bazie danych, której działanie opiera się na rekordach. Każdy blok zawiera rekord pochodzący z tabeli

Rozważ przykład użycia bazy danych w rozwiązaniu typu OLTP (ang. *online transaction processing*), np. w aplikacji internetowej typu e-commerce, która do przechowywania informacji wykorzystuje bazę danych MySQL. Generowane przez tę aplikację internetową żądania odczytu i zapisu są wykonywane do bazie danych MySQL i bardzo często obejmują wiele wartości dla danego rekordu, np. szczegóły dotyczące zamówienia wyświetlane na stronie zawierającej podsumowanie zamówienia. W danej chwili możliwe jest uaktualnianie lub pobieranie informacji tylko o jednym zamówieniu. Bazujący na rekordzie magazyn danych jest optymalny, ponieważ na dysku dane aplikacji muszą być przechowywane blisko siebie, a ilość jednocześnie pobieranych danych jest niewielka.

W omawianym przykładzie nieefektywny sposób użycia dysku ze względu na pozostawianie przez rekordy pustego miejsca w blokach wydaje się rozsądnym kompromisem, ponieważ znaczenie ma szybkość częstego odczytywania i zapisywania pojedynczych rekordów. Jednak podczas analizy danych sytuacja jest zgoła odmienna. Zamiast konieczności

częstego zapisywania i odczytywania niewielkich ilości danych mamy do czynienia z rzadkim zapisywaniem i odczytywaniem dużych ilości danych. Ponadto istnieje znacznie mniejsze prawdopodobieństwo, że wykonywane w trakcie analizy danych zapytanie będzie wymagało wielu bądź wszystkich kolumn tabeli, a raczej pojedynczej kolumny tabeli zawierającej ich wiele.

Na przykład rozważ tabelę zamówień w naszej fikcyjnej aplikacji typu e-commerce. Zawiera ona wiele informacji, m.in. wartość zamówienia i adres, pod który ma być ono wysłane. W przeciwieństwie do aplikacji internetowej działającej z pojedynczymi zamówieniami analityk używający hurtowni danych będzie chciał analizować zamówienia w większych partiach. Ponadto tabela zawierająca dane zamówień w hurtowni danych ma kolumny dodatkowe, których wartości pochodzą z wielu tabel w bazie danych MySQL. Na przykład to mogą być informacje o kliencie składającym dane zamówienie. Być może analityk będzie chciał zsumować wszystkie zamówienia złożone przez klientów aktualnie posiadających aktywne konta. Wprawdzie takie zapytanie może obejmować miliony rekordów, ale dane będą odczytywane np. tylko z dwóch kolumn, takich jak `OrderTotal` (wartość całkowita zamówienia) i `CustomerActive` (klient z aktywnym kontem). Ostatecznie analiza nie dotyczy tworzenia lub zmiany danych (jak ma to miejsce w OLTP), ale raczej pochodzenia wskaźników i zrozumienia danych.

Jak pokazałem na rysunku 3.2, kolumnowa baza danych, taka jak Snowflake lub Amazon Redshift, przechowuje dane w blokach na dysku na podstawie kolumn, a nie rekordów. W omawianym przykładzie zapytanie wykonywane przez analityka musi jedynie uzyskać dostęp do bloków przechowujących wartości `OrderTotal` i `CustomerActive`, a nie bloków przechowujących całe rekordy, jak ma to miejsce w przypadku bazy danych MySQL. To oznacza mniejsze obciążenie dyskowymi operacjami wejścia-wyjścia i mniejszą ilość danych koniecznych do wczytania, aby przeprowadzić filtrowanie i sumowanie wymagane przez zapytanie analityczne. Kolejną korzyścią jest zmniejszenie poziomu użycia pamięci masowej dzięki możliwości pełnego wykorzystania bloków i zastosowaniu optymalnej kompresji, ponieważ dany blok przechowuje ten sam typ danych, a nie wiele typów jak to zwykle ma miejsce w przypadku danych przechowywanych w rekordach.

OrderId	CustomerId	Shipping Country	Order Total	Customer Active
1	1258	US	55,25	TRUE
2	5698	AUS	125,36	TRUE
3	2265	US	776,95	TRUE
4	8954	CA	32,16	FALSE

Blok 1	1, 2, 3, 4
Blok 2	1258, 5698, 2265, 8954
Blok 3	US, AUS, US, CA
Blok 4	55,25, 125,36, 776,95, 32,16
Blok 5	TRUE, TRUE, TRUE, FALSE

Rysunek 3.2. Przykład tabeli przechowywanej w kolumnowej bazie danych. Każdy blok dysku przechowuje dane pochodzące z tej samej kolumny. Dwie kolumny używane w naszym przykładzie zostały zaznaczone kolorem szarym. W celu wykonania zapytania konieczne jest uzyskanie dostępu jedynie do tych bloków. Każdy blok zawiera dane tego samego typu, co pozwala na optymalne zastosowanie kompresji

Pojawienie się kolumnowych baz danych spowodowało, że przechowywanie, przekształcanie i wykonywanie zapytań do ogromnych zbiorów danych w hurtowni danych stało się efektywne. Inżynierowie danych mogą tę zaletę wykorzystać podczas opracowywania etapów potoków danych specjalizujących się w wyodrębnianiu danych i ich wczytywaniu w hurtowniach danych. Następnie te dane mogą być przekształcane, modelowane i używane przez analityków lub naukowców, którzy lepiej czują się w ramach baz danych. W efekcie ELT stał się idealnym wzorcem dla potoków hurtowni danych, a także w innych zastosowaniach, np. w uczeniu maszynowym lub w tworzeniu produktów.

Podwzorzec EtLT

Gdy ELT został dominującym wzorcem, stało się jasne, że przeprowadzanie pewnych przekształceń już po wyodrębnieniu danych, ale jeszcze przed ich wczytaniem, przynosi określone korzyści. Jednak zamiast przekształcenia wykorzystującego logikę biznesową lub modelowanie danych ten rodzaj przekształcania jest bardziej ograniczony. Oznaczam go *małą literą t — EtLT*.

Oto wybrane przykłady typów przekształceń wpasowujące się w podwzorzec EtLT:

- usuwanie powielonych rekordów w tabeli,
- przetwarzanie parametrów URL na postać poszczególnych komponentów,
- maskowanie bądź w inny sposób zabezpieczanie danych wrażliwych.

Tego rodzaju przekształcenia są w pełni oddzielone od logiki biznesowej bądź, w przypadku zadań takich jak maskowanie danych wrażliwych, są umieszczone jak najwcześniej w potoku ze względów czasowych, prawnych bądź z uwagi na zapewnienie bezpieczeństwa. Poza tym rozsądne jest używanie właściwego narzędzia do wykonania danej pracy. Jak wyjaśnię w rozdziałach 4. i 5., większość nowoczesnych hurtowni danych wczytuje dane w najbardziej efektywny sposób, gdy są one doskonale przygotowane. W przypadku potoków przekazujących ogromne ilości danych lub w sytuacjach, w których opóźnienie ma znaczenie kluczowe, przeprowadzenie pewnych prostych przekształceń między etapami wyodrębniania i wczytywania jest warte włożonego wysiłku.

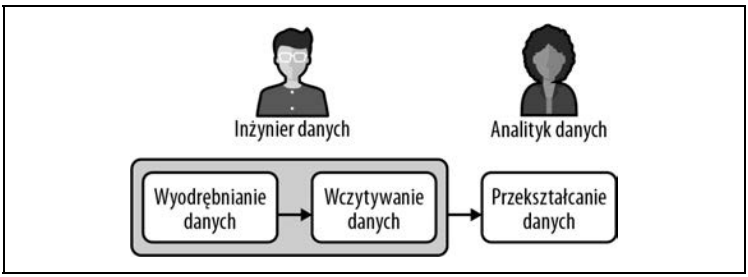
Możesz przyjąć założenie, że pozostałe wzorce dotyczące ELT zostały opracowane również z uwzględnieniem podwzorca EtLT.

ELT w analizie danych

ELT stał się najczęściej stosowanym i w mojej opinii jest najbardziej optymalnym wzorcem dla potoków danych przygotowanych na potrzeby analizy danych. Jak wcześniej wspomniałem, kolumnowe bazy danych świetnie sprawdzają się podczas obsługi ogromnych ilości danych. Zostały opracowane również do obsługi szerokich tabel, czyli tabel zawierających wiele kolumn — dzięki temu, że tylko dane użyte w wykonywanym zapytaniu są skanowane na dysku i wczytywane do pamięci.

Pomijając kwestie techniczne — analiza danych zwykle odbywa się płynnie w SQL. Dzięki ELT inżynierowie danych mogą skoncentrować się na krokach wyodrębniania i wczytywania w potoku (pobieranie danych), podczas gdy analitycy mogą wykorzystać SQL do przekształcenia danych,

które zostały pobrane jako niezbędne na potrzeby raportowania i analizy. Taka separacja jest niemożliwa w przypadku wzorca ETL, ponieważ inżynierowie danych będą potrzebni na każdym etapie działania potoku danych. Jak pokazałem na rysunku 3.3, ELT pozwala zespołom zajmującym się danymi skoncentrować się na swoich umiejętnościach, a mniej na zależnościach i koordynacji.



Rysunek 3.3. Wzorzec ELT umożliwia podział odpowiedzialności między inżynierami danych i analitykami (bądź naukowcami). Każdy z nich może działać niezależnie, korzystając z narzędzi i języków, w pracy z którymi czuje się komfortowo

Ponadto wzorzec ELT zmniejsza potrzebę dokładnego przewidywania, co analityk będzie robił z danymi w trakcie procesów wyodrębniania i wczytywania. Wprawdzie ogólne zrozumienie sposobu pracy jest niezbędne w celu wyodrębnienia i wczytania poprawnych danych, ale przesunięcie na później kroku przetwarzania zapewnia analitykowi więcej możliwości i dużą elastyczność.

Uwaga

Wraz z pojawieniem się ELT analitycy danych stali się bardziej niezależni i zdolni do lepszego wykorzystania danych, a przy tym nie są „blokowani" przez inżynierów danych. Z kolei inżynierowie danych mogą skoncentrować się na pobieraniu danych i dostarczaniu infrastruktury pozwalającej analitykom na samodzielne tworzenie i wdrażanie utworzonego w SQL kodu przeznaczonego do przekształcania danych. To spowodowało pojawienie się nowych stanowisk, takich jak **inżynier analityk.** W rozdziale 6. dowiesz się więcej o tym, jak ci analitycy danych i inżynierowie danych przekształcają dane w celu tworzenia modeli danych.

ELT dla naukowca

Potoki danych tworzone dla zespołów naukowców są podobne do tych opracowywanych na potrzeby analizy danych w hurtowniach danych. Podobnie jak w przypadku zespołu analityków, tak samo w przypadku zespołu naukowców inżynierowie danych koncentrują się na pobieraniu danych, które następnie trafiają do hurtowni danych bądź jeziora danych. Jednak naukowcy mają wobec danych inne wymagania niż analitycy.

Wprawdzie nauka jest ogólnie dość obszerną dziedziną, ale naukowcy potrzebują dostępu do znacznie dokładniejszych — i czasami niezmodyfikowanych — danych niż wymagane przez analityków. Podczas gdy analitycy danych tworzą modele danych przeznaczone do generowania wskaźników i paneli, naukowcy poświęcają czas na poznawanie danych i opracowywanie modeli prognostycznych. Omówienie szczegółowo zadań naukowca wykracza poza zakres tematyczny książki, ale trzeba wiedzieć, że te różnice na wysokim poziomie mają wpływ na projektowanie potoków danych przeznaczonych dla naukowców.

Jeżeli tworzysz potoki danych przeznaczone do używania przez naukowców, przekonasz się, że etapy wyodrębniania i wczytywania we wzorcu ELT pozostaną praktycznie takie same jak w przypadku rozwiązań tworzonych dla analityków. W rozdziale 4. i 5. znacznie dokładniej omówię te kroki pod względem technicznym. Na etapie przekształcania danych w potoku ELT naukowcy mogą mieć pożytek podczas pracy z niektórymi modelami danych opracowanymi dla analityków (zob. rozdział 6.), ale prawdopodobnie będą używać większości danych otrzymanych na etapach wyodrębniania i wczytywania.

ELT dla produktów danych i uczenia maszynowego

Dane są używane nie tylko podczas analizy, raportowania i tworzenia modeli prognostycznych. Znajdują zastosowanie również w **produktach danych**. Do najczęściej stosowanych produktów danych zaliczamy np.:

- silniki rekomendacji stosowane np. w rozwiązaniach strumieniowania wideo,
- spersonalizowane wyniki wyszukiwania w witrynie internetowej typu e-commerce,

- aplikację przeprowadzającą analizę utworzonych przez użytkowników recenzji restauracji.

Każdy z tych produktów danych prawdopodobnie jest wspierany przez jeden lub więcej modeli uczenia maszynowego, które wymagają danych na etapach trenowania i weryfikacji. Takie dane mogą pochodzić z wielu systemów źródłowych i wymagać przekształceń w celu ich przygotowania do użycia w modelu. Wzorzec typu ELT jest doskonale przygotowany do takich potrzeb, choć pojawia się pewna liczba określonych wyzwań na wszystkich etapach działania potoków danych przeznaczonych do obsługi produktów danych.

Etapy potoku danych dla uczenia maszynowego

Podobnie jak w przypadku potoków danych opracowanych dla analizy, na których to skoncentrowałem się w książce, potoki danych przeznaczone dla uczenia maszynowego są opracowywane z zastosowaniem wzorca podobnego do ELT — przynajmniej na początku potoku. Różnica polega na tym, że zamiast etapu przekształcania koncentrującego się na przekształcaniu danych trafiających do modelu danych, gdy dane zostaną wyodrębnione i wczytane do hurtowni danych bądź jeziora danych, mamy wiele kroków zaangażowanych w tworzenie i uaktualnianie modelu ML.

Jeżeli masz doświadczenie w pracy z tworzeniem modeli uczenia maszynowego, wymienione tutaj kroki będą Ci doskonale znane.

Pobieranie danych

W tym kroku jest stosowany dokładnie ten sam proces, który został omówiony w rozdziałach 4. i 5. Wprawdzie pobierane dane mogą się różnić, ale stosowana podczas tej operacji logika pozostaje praktycznie taka sama w potokach danych przeznaczonych dla analizy danych i ML, przy czym w przypadku tych drugich mamy jeszcze kwestie dodatkowe do uwzględnienia — na przykład zagwarantowanie, że pobierane dane są wersjonowane w sposób, dzięki któremu modele uczenia maszynowego mogą później odwoływać się do określonych zbiorów danych podczas trenowania lub weryfikacji. Istnieje wiele narzędzi i podejść stosowanych do wersjonowania

zbiorów danych. Sugeruję zapoznać się z punktem „Więcej zasobów dotyczących potoków danych dla uczenia maszynowego" w dalszej części rozdziału.

Wstępne przetwarzanie danych

Pobierane dane prawdopodobnie nie będą gotowe do użycia w modelach uczenia maszynowego. Na etapie wstępnego przetwarzania danych są one oczyszczane i przygotowywane do wykorzystania w tych modelach. Na przykład to jest w potoku danych etap, w trakcie którego tekst jest tokenizowany, cechy są konwertowane na wartości liczbowe, zaś wartości danych wejściowych są normalizowane.

Trenowanie modelu

Po pobraniu nowych danych i ich wstępnym przetworzeniu konieczne jest ponowne wytrenowanie modeli uczenia maszynowego.

Wdrażanie modelu

Wdrożenie modeli w produkcji może być największym wyzwaniem podczas przejścia z uczenia maszynowego wykorzystywanego podczas badań do rzeczywistego produktu bazującego na danych. Tutaj niezbędne są nie tylko wersjonowane zbiory danych, ale również wytrenowane modele. Bardzo często API REST używa się podczas wykonywania zapytań do wdrożonego modelu, a punkty końcowe API — dla różnych wersji modelu. To całkiem sporo do śledzenia i wymaga koordynacji między naukowcami, inżynierami uczenia maszynowego oraz inżynierami danych, aby otrzymać produkt możliwy do użycia w produkcji. Doskonale opracowany potok danych ma kluczowe znaczenie w spojeniu całości.

WERYFIKACJA POBRANYCH DANYCH

Jak wyjaśnię w rozdziale 8., weryfikacja danych w potoku ma kluczowe znaczenie i powinna być przeprowadzana. W potokach utworzonych na potrzeby analizy danych weryfikacja często odbywa się po pobraniu danych (etapy wyodrębnienia i wczytywania), a także po modelowaniu danych (przekształcanie). W potokach uczenia maszynowego weryfikacja pobranych danych jest również ważna. Nie należy tego istotnego kroku mylić z weryfikacją modelu uczenia maszynowego, który oczywiście jest standardową częścią tworzenia rozwiązań z zakresu uczenia maszynowego.

Wykorzystanie informacji zwrotnych w potoku

Każdy dobry potok dla uczenia maszynowego będzie umożliwiał zbieranie opinii pozwalających na usprawnienie modelu. Rozważmy np. zawartość modelu rekomendacji dla usługi strumieniowania wideo. Aby w przyszłości można było zmierzyć i poprawić model, konieczne jest śledzenie tego, co jest rekomendowane użytkownikom, jakie rekomendacje zostały przez nich kliknięte, które rekomendacje okazały się trafione itd. To wymaga współpracy z zespołem programistów wykorzystującym model w usłudze strumieniowania. Programiści muszą zaimplementować pewnego rodzaju zbieranie zdarzeń w celu śledzenia rekomendacji przekazanych poszczególnym użytkownikom, wersji modelu użytego do rekomendacji, informacji o tym, kiedy rekomendacja została kliknięta. Następnie powinno odbyć się przejście do prawdopodobnie już zebranych danych dotyczących używania określonych treści przez użytkownika.

Wszystkie te informacje mogą być pobrane z powrotem do hurtowni danych oraz wykorzystane w przyszłych wersjach modelu albo jako dane używane do trenowania modeli, albo na potrzeby analizy przeprowadzanej przez człowieka (prawdopodobnie naukowca) w celu wykorzystania w przyszłych modelach bądź eksperymentach.

Ponadto zebrane dane mogą być pobrane, przekształcone i przeanalizowane przez analityków z wykorzystaniem wzorca ELT omówionego w książce. Analitycy często będą sprawdzali efektywność modeli i tworzyli panele w celu dostarczenia organizacji informacji o kluczowych wskaźnikach modelu. Interesariusze mogą korzystać z tych paneli do poznawania efektywności poszczególnych modeli podczas podejmowania decyzji biznesowych oraz dotyczących klientów.

Więcej zasobów dotyczących potoków danych dla uczenia maszynowego

Tworzenie potoków danych przeznaczonych na potrzeby uczenia maszynowego jest rozbudowanym tematem. W zależności od podjętych decyzji w zakresie infrastruktury i stopnia złożoności środowiska uczenia maszynowego warto sięgnąć po inne książki, które gorąco polecam:

Hannes Hapke i Catherine Nelson, *Building Machine Learning Pipelines*, O'Reilly, 2020.

Aurélien Géron *Uczenie maszynowe z użyciem Scikit-Learn i TensorFlow. Wydanie II*, Helion, Gliwice 2020.

Ponadto wymieniona tutaj pozycja jest świetnym wprowadzeniem do tematu uczenia maszynowego:

Andreas C. Müller i Sarah Guido, *Introduction to Machine Learning with Python*, O'Reilly, 2016.

Skorowidz

R

S

Ś

T

U

W

Z, Ź


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

Poznaj najlepsze praktyki projektowania i implementacji potoków danych!

Poprawnie zaprojektowane i wdrożone potoki danych mają kluczowe znaczenie dla pomyślnej analizy danych, a także w trakcie uczenia maszynowego. Pozyskanie ogromnych ilości danych z różnych źródeł najczęściej nie stanowi problemu. Nieco trudniejsze jest zaprojektowa nie procesu ich przetwarzania w celu dostarczenia kontekstu w taki sposób, aby efektywnie korzystać z posiadanych danych w codziennej pracy organizacji i podejmować dzięki nim rozsądne decyzje.

Oto zwięzły przewodnik przeznaczony dla inżynierów danych, którzy chcą poznać zasady implementacji potoków danych. Wyjaśnia najważniejsze pojęcia związane z potokami danych, opisuje zasady ich tworzenia i implementacji, prezentuje także najlepsze praktyki stosowane przez liderów w branży analizy danych. Dzięki książce zrozumiesz, w jaki sposób potoki danych działają na nowoczesnym stosie danych, poznasz też typowe zagadnienia, które trzeba przemyśleć przed podjęciem decyzji dotyczących implementacji. Dowiesz się, jakie są zalety samodzielnego opracowania rozwiązania, a jakie — zakupu gotowego produktu. Poznasz również podstawowe koncepcje, które mają zastosowanie dla frameworków typu open source, produktów komercyjnych i samodzielnie opracowywanych rozwiązań.

Dowiedz się:

- czym jest potok danych i na czym polega jego działanie
- jak się odbywa przenoszenie i przetwarzanie danych w nowoczesnej infrastrukturze
- jakie narzędzia są szczególnie przydatne do tworzenia potoków danych
- jak używać potoków danych do analizy i tworzenia raportów
- jakie są najważniejsze aspekty obsługi potoków, ich testowania i rozwiązywania problemów

James Densmore jest dyrektorem do spraw infrastruktury danych na potrzeby analityki biznesowej w HubSpot, a także założycielem i głównym konsultantem w Data Liftoff. Od ponad dziesięciu lat kieruje zespołami inżynierów danych i zajmuje się projektowaniem infrastruktury danych.

Helion

helion.pl

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

KOD KORZYŚCI
Sięgnij po więcej! ▶

ISBN 978-83-8322-338-4

9 788383 223384

Cena: 49,90 zł