XI Konferencja PLOUG
Kościelisko
Październik 2005

Oracle Data Mining – odkrywanie wiedzy w dużych wolumenach danych

Mikołaj Morzy

Instytut Informatyki Politechniki Poznańskiej

e-mail: Mikolaj.Morzy@cs.put.poznan.pl

Streszczenie:
Bardzo duże wolumeny danych zawierają w sobie użyteczną wiedzę, ukrytą pod postacią wzorców, trendów, regularności i wyjątków. Tradycyjne metody analizy danych tracą zastosowanie, nie będąc w stanie przetworzyć ogromnych ilości danych gromadzonych przez współczesne organizacje. Eksploracja danych (ang. *data mining*) to techniki, metody i algorytmy umożliwiające odkrywanie użytecznej wiedzy w bardzo dużych bazach danych.
Oracle Data Mining to nowy komponent serwera bazy danych, oferujący użytkownikom wiele różnych metod eksploracji danych. Użytkownicy mogą wykorzystywać takie funkcje eksploracji, jak: klasyfikacja i predykcja, regresja, określanie ważności atrybutów, grupowanie obiektów podobnych, znajdowanie reguł asocjacyjnych, oraz eksploracja dokumentów tekstowych. Metody, modele i wyniki eksploracji są dostępne poprzez interfejs PL/SQL oraz interfejs dla języka Java, dzięki czemu omawiane techniki mogą zostać z łatwością włączone do aplikacji użytkownika. Dodatkowo, użytkownicy mają do dyspozycji program Oracle Data Miner, udostępniający graficzny interfejs użytkownika do całej funkcjonalności Oracle Data Mining.
W referacie autor przedstawi opis architektury systemu, krótką charakterystykę wszystkich funkcji oferowanych w ramach Oracle Data Mining, oraz scenariusze potencjalnego wykorzystania każdej z funkcji.

Informacja o autorze:
Mikołaj Morzy jest adiunktem w Instytucie Informatyki Politechniki Poznańskiej. Wykładał również na uniwersytetach w Niemczech i Stanach Zjednoczonych. Jest autorem 30 publikacji naukowych z dziedziny baz danych, hurtowni danych i eksploracji danych.

1. Wprowadzenie

Obserwowane w ostatnich latach upowszechnienie się systemów baz danych w praktycznie każdej dziedzinie ludzkiej działalności doprowadziło do gwałtownego wzrostu ilości danych, które są gromadzone i zapisywane w postaci cyfrowej. Systemy baz danych są obecne niemal wszędzie: sprzedaż detaliczna, bankowość, finanse, ubezpieczenia, medycyna, edukacja, handel elektroniczny, we wszystkich tych dziedzinach systemy baz danych stanowią niezbędny element infrastruktury informatycznej. Na przestrzeni lat systemy baz danych nieustannie ewoluowały, od prostych systemów plikowych, poprzez systemy sieciowe i hierarchiczne, aż po systemy relacyjne, obiektowe i semistrukturalne. Równolegle z rozwojem modeli danych wykorzystywanych w bazach danych powstawały nowe modele przetwarzania i architektury systemów baz danych. Ewolucja systemów baz danych była podyktowana, z jednej strony, rosnącymi wymaganiami dotyczącymi funkcjonalności baz danych, a z drugiej strony, koniecznością obsługiwania coraz większych kolekcji danych. Hurtownia danych (ang. *data warehouse*) jest architekturą bazy danych umożliwiającą integrację ogromnych ilości danych pochodzących z heterogenicznych źródeł i operującą na wielowymiarowym modelu danych. Głównym zadaniem hurtowni danych jest prezentacja wysokiej jakości zintegrowanych danych na potrzeby analizy biznesowej, weryfikacji hipotez, wspomagania decyzji, oraz odkrywania wiedzy w danych.

Współczesne bazy danych stoją przed licznymi wyzwaniami: skalowalność, efektywność, bogata funkcjonalność, pojemność, to niezbędne cechy dobrego systemu zarządzania bazą danych. Według corocznego raportu Winter Corporation [Wint05] rozmiar największej operacyjnej bazy danych w roku 2005 osiągnął 23 TB (Land Registry for England and Wales), podczas gdy rozmiar największej hurtowni danych przekroczył 100 TB (Yahoo!). W najbliższym czasie należy spodziewać się dalszego gwałtownego wzrostu rozmiarów baz danych. Na potrzeby budowanego w laboratorium CERN akceleratora wiązek protonowych LHC stworzono bazę danych zdolną do składowania niemal exabajta danych (1 EB = 1024 PB = 10^{18} B) [LHC05]. Akcelerator LHC rozpocznie pracę w 2007 roku i corocznie będzie generował około 15 petabajtów danych z oszałamiającą prędkością do 1,5 GB/sek. Eksperymenty zaplanowano na 15 lat, co daje w efekcie astronomiczną ilość 225 PB danych. Inne przykłady gwałtownie zwiększających się baz danych obejmują bazy danych informacji naukowych (projekt poznania ludzkiego genomu, informacje astronomiczne), dane strategiczne (informacje związane z bezpieczeństwem narodowym), oraz komercyjne kolekcje danych (dane POS, dane o ruchu internetowym, dane o transakcjach elektronicznych). Nieustanny wzrost rozmiarów baz danych jest skutkiem postępu w technikach pozyskiwania i składowania informacji. Niestety, postęp ten nie jest równoważony przez zdolność do analizy pozyskanych danych.

Gigantyczne rozmiary współczesnych kolekcji danych skutecznie uniemożliwiają jakiekolwiek próby ręcznej analizy zgromadzonych informacji. Z drugiej strony, większość repozytoriów zawiera użyteczną wiedzę ukrytą w danych pod postacią trendów, regularności, korelacji, czy osobliwości. W ostatnich latach zaproponowano wiele metod automatycznego i półautomatycznego pozyskiwania wiedzy z ogromnych wolumenów danych. Metody te określa się mianem eksploracji danych (ang. *data mining*) lub odkrywania wiedzy w bazach danych (ang. *knowledge discovery in databases*). Wiedza odkryta w procesie eksploracji danych może zostać wykorzystana do wspomagania procesu podejmowania decyzji, predykcji przyszłych zdarzeń, czy określania efektywnych strategii biznesowych [BL97].

Głównym celem niniejszego artykułu jest zapoznanie czytelnika z podstawowymi technikami eksploracji danych oraz przedstawienie rozwiązań oferowanych przez system zarządzania bazą danych Oracle w zakresie eksploracji danych. Artykuł jest zorganizowany w następujący sposób: w punkcie 2 prezentujemy przegląd podstawowych technik eksploracji danych. Punkt 3 zawiera

opis architektury Oracle Data Mining i przedstawienie dostępnych interfejsów programistycznych. Wreszcie, w punkcie 4 dokonujemy podsumowania.

2. Techniki eksploracji danych

Eksploracja danych to proces odkrywania nowych, wcześniej nieznanych, potencjalnie użytecznych, zrozumiałych i poprawnych wzorców w bardzo dużych wolumenach danych [FPSU96]. Eksploracja danych jest dyscypliną łączącą takie dziedziny jak: systemy baz danych, statystyka, systemy wspomagania decyzji, sztuczna inteligencja, uczenie maszynowe, wizualizacja danych, przetwarzanie równoległe i rozproszone, i wiele innych. Techniki eksploracji danych wykorzystują różne modele wiedzy do reprezentowania wzorców obecnych w danych. Modele te obejmują, między innymi, reguły asocjacyjne [AIS93], reguły cykliczne i okresowe [ORS98], reguły dyskryminacyjne i charakterystyczne [Cen87], klasyfikatory bayesowskie [LIT92], drzewa decyzyjne [Qui86,Qui93], wzorce sekwencji [AS95], skupienia obiektów podobnych [ELL01], przebiegi czasowe, oraz osobliwości i wyjątki. Równolegle z modelami wiedzy opracowano wiele algorytmów odkrywania wiedzy w bazach danych [HK00,WF00]. Proces odkrywania wiedzy przebiega najczęściej na danych udostępnianych przez hurtownię danych. Analizowane dane są poddawane wstępnemu przetwarzaniu, transformacji, czyszczeniu, usuwaniu niespójności, czy wzbogacaniu wiedzą domenową. Wiedza odkryta w danych może być postrzegana jako wartość dodana, podnosząca jakość danych i znacząco polepszająca jakość decyzji podejmowanych na podstawie danych.

Techniki eksploracji danych można ogólnie podzielić na dwie zasadnicze kategorie. Techniki predykcyjne starają się, na podstawie odkrytych wzorców, dokonać uogólnienia i przewidywania (np. wartości nieznanego atrybutu, zachowania i cech nowego obiektu, itp.). Przykładami zastosowania technik predykcyjnych mogą być: identyfikacja docelowych grup klientów, ocena ryzyka ubezpieczeniowego związanego z klientem, lub oszacowanie prawdopodobieństwa przejścia klienta do konkurencyjnego usługodawcy. Techniki deskrypcyjne mają na celu wykorzystanie wzorców odkrytych w danych do spójnego opisu danych i uchwycenia ogólnych cech danych. Typowe przykłady technik deskrypcyjnych obejmują odkrywanie grup podobnych klientów, znajdowanie zbiorów produktów często kupowanych razem, lub identyfikacja osobliwości występujących w danych. Inny podział technik eksploracji danych jest związany z charakterem danych wejściowych. W przypadku technik uczenia nadzorowanego (ang. *supervised learning*) dane wejściowe zawierają tzw. zbiór uczący, w którym przykładowe instancje danych są powiązane z prawidłowym rozwiązaniem. Na podstawie zbioru uczącego dana technika potrafi „nauczyć się" odróżniać przykłady należące do różnych klas, a zdobyta w ten sposób wiedza może być wykorzystana do formułowania uogólnień dotyczących przyszłych instancji problemu. Najczęściej spotykanymi technikami uczenia nadzorowanego są techniki klasyfikacji (drzewa decyzyjne [Qui86, Qui93], algorytmy bazujące na *n* najbliższych sąsiadach [Aha92], sieci neuronowe [MI94], statystyka bayesowska [Bol04]), oraz techniki regresji. Drugą klasą technik eksploracji danych są techniki uczenia bez nadzoru (ang. *unsupervised learning*), gdy algorytm nie ma do dyspozycji zbioru uczącego. W takim przypadku algorytm eksploracji danych stara się sformułować model najlepiej pasujący do obserwowanych danych. Przykłady technik uczenia bez nadzoru obejmują techniki analizy skupień [ELL01] (ang. *clustering*), samoorganizujące się mapy [Koh00], oraz algorytmy maksymalizacji wartości oczekiwanej [DLR77] (ang. *expectation-maximization*).

Terminy „eksploracja danych" i „odkrywanie wiedzy w bazach danych" są często stosowane wymiennie, choć drugi termin posiada dużo szersze znaczenie. Odkrywanie wiedzy to cały proces akwizycji wiedzy, począwszy od selekcji danych źródłowych a skończywszy na ocenie odkrytych wzorców. Zgodnie z tą definicją, eksploracja danych oznacza zastosowanie konkretnego algorytmu odkrywania wzorców na wybranych danych źródłowych i stanowi jeden z etapów składowych całego procesu odkrywania wiedzy. Na cały proces składają się [HK00]: sformułowanie problemu, wybór danych, czyszczenie danych, integracja danych, transformacja danych, eksploracja danych, wizualizacja i ocena odkrytych wzorców, i wreszcie zastosowanie wzorców. Postać uzyskanych

wzorców zależy od zastosowanej techniki eksploracji danych. Poniżej przedstawiono opisy najpopularniejszych technik eksploracji. Z konieczności nie jest to lista wyczerpująca, uwzględniono tylko te metody eksploracji danych, które zostały zaimplementowane w pakiecie Oracle Data Mining.

2.1. Reguły asocjacyjne

Pojęcie reguł asocjacyjnych (ang. *association rules*) zostało po raz pierwszy wprowadzone w [AIS93]. Odkrywanie reguł asocjacyjnych polega na znalezieniu w dużej kolekcji zbiorów korelacji wiążącej współwystępowanie podzbiorów elementów. Znalezione korelacje są prezentowane jako reguły postaci $X \Rightarrow Y$ (*wsparcie*, *ufność*), gdzie X i Y są rozłącznymi zbiorami elementów, *wsparcie* oznacza częstotliwość występowania zbioru $X \cup Y$ w kolekcji zbiorów, zaś *ufność* reprezentuje prawdopodobieństwo warunkowe $P(Y|X)$. Na gruncie analizy ekonomicznej reguły asocjacyjne są najczęściej stosowane do analizy koszyka zakupów. W takim przypadku wejściowa kolekcja zbiorów odpowiada bazie danych koszyków zakupów klientów, a odkryte reguły asocjacyjne reprezentują zbiory produktów, które są często nabywane wspólnie. Przykładowo, reguła asocjacyjna odkryta w bazie danych transakcji sklepowych mogłaby mieć postać `{chleb,kiełbasa}⇒{musztarda} (3%,75%)` a jej interpretacja byłaby następująca: 3% klientów sklepu kupiło chleb, kiełbasę i musztardę w trakcie pojedynczej transakcji, przy czym 75% transakcji zawierających chleb i kiełbasę, zawierało również musztardę. Odkryte reguły asocjacyjne mogą być wykorzystane do organizowania promocji i sprzedaży wiązanej, do konstruowania katalogów wysyłkowych, ustalania rozmieszczenia towarów na półkach, itp.

Reguły asocjacyjne doczekały się wielu rozwinięć i modyfikacji. W [SA96a] zaproponowano algorytm służący do znajdowania ilościowych reguł asocjacyjnych (ang. *quantitative association rules*), reprezentujących korelacje między wartościami różnych atrybutów. Model ten umożliwiał także włączenie do eksploracji atrybutów numerycznych, które jednak musiały być uprzednio dyskretyzowane. Przykładem ilościowej reguły asocjacyjnej, która mogłaby być odkryta w bazie danych, jest reguła: `wiek∈(20,30)∧zawod='student'⇒dochod='niski' (2%,60%)`. Modyfikacją oryginalnego sformułowania była propozycja przedstawiona w [SA95]. Celem było uwzględnienie taksonomii elementów wchodzących w skład reguł i umożliwienie odkrywania uogólnionych reguł asocjacyjnych (ang. *generalized association rules*), zawierających elementy z różnych poziomów taksonomii. Dalsze propozycje obejmowały reguły cykliczne [ORS98], czasowo-przestrzenne reguły asocjacyjne [GP05], i wiele innych. Duży wysiłek badawczy włożono w opracowywanie efektywnych algorytmów odkrywania reguł asocjacyjnych. Najbardziej znane przykłady takich algorytmów to Apriori [AS94], FreeSpan [HP+00] oraz Eclat [ZP+97].

2.2. Wzorce sekwencji

Sekwencja jest to uporządkowany ciąg zbiorów elementów, gdzie każdy zbiór posiada dodatkowo znacznik czasowy. Sekwencja może reprezentować zbiory produktów kupowanych przez klientów podczas kolejnych wizyt w sklepie, filmy wypożyczane podczas kolejnych wizyt w wypożyczalni wideo, czy rozmowy telefoniczne wykonywane w określonych przedziałach czasu. Problem znajdowania wzorców sekwencji został po raz pierwszy sformułowany w [AS95] i polega na znalezieniu, w bazie danych sekwencji, podsekwencji występujących częściej niż zadany przez użytkownika próg częstości, zwany progiem minimalnego wsparcia (ang. *minsup*). Przykładem wzorca sekwencji znalezionego w bazie danych księgarni może być następujący wzorzec: `{'Ogniem i mieczem'}⇒{'Potop'}⇒{'Pan Wołodyjowski'} (1,5%)`. Dodatkowo, użytkownik może sformułować ograniczenia dotyczące maksymalnych interwałów czasowych między kolejnymi wystąpieniami elementów sekwencji. Podobnie jak w przypadku reguł asocjacyjnych, także wzorce sekwencji doczekały się rozwinięć (np. uogólnione wzorce sekwencji [SA96b]) oraz efektywnych algorytmów eksploracji, takich jak GSP. Domeny potencjalnego zastosowania wzorców sekwencji praktycznie pokrywają się z regułami asocjacyjnymi i obejmują, między innymi:

telekomunikację, handel detaliczny, zastosowania bankowe, ubezpieczenia, analizę dzienników serwerów WWW, i wiele innych.

2.3. Klasyfikacja

Klasyfikacja (ang. *classification*) jest jedną z najpopularniejszych technik eksploracji danych. Polega na stworzeniu modelu, który umożliwia przypisanie nowego, wcześniej niewidzianego obiektu, do jednej ze zbioru predefiniowanych klas. Model umożliwiający takie przypisanie nazywa się klasyfikatorem. Klasyfikator dokonuje przypisania na podstawie doświadczenia nabytego podczas trenowania i testowania na zbiorze uczącym. W trakcie wieloletnich prac prowadzonych nad klasyfikatorami i ich zastosowaniem w statystyce, uczeniu maszynowym, czy sztucznej inteligencji, zaproponowano bardzo wiele metod klasyfikacji. Najczęściej stosowane techniki to klasyfikacja bayesowska [LIT92], klasyfikacja na podstawie k najbliższych sąsiadów [Aha92], drzewa decyzyjne [BF+84,Qui86,Qui93], sieci neuronowe [Big96], sieci bayesowskie [HGC95], czy algorytmy SVM [Bur98,Vap95] (ang. *support vector machines*). Popularność technik klasyfikacji wynika przede wszystkim z faktu szerokiej stosowalności tego modelu wiedzy. Klasyfikatory mogą być wykorzystane do oceny ryzyka związanego z udzieleniem klientowi kredytu, wyznaczeniem prawdopodobieństwa przejścia klienta do konkurencji, czy znalezienia zbioru klientów, którzy z największym prawdopodobieństwem odpowiedzą na ofertę promocyjną. Podstawową wadą praktycznie wszystkich technik klasyfikacji jest konieczność starannego wytrenowania klasyfikatora i trafnego wyboru rodzaju klasyfikatora w zależności od charakterystyki przetwarzanych danych. Te czynności mogą wymagać od użytkownika wiedzy technicznej, zazwyczaj wykraczającej poza sferę kompetencji analityków i decydentów. Techniką podobną do klasyfikacji jest regresja (ang. *regression*). Różnica między dwiema technikami polega na tym, że w przypadku klasyfikacji przewidywana wartość jest kategoryczna, podczas gdy w regresji celem modelu jest przewidzenie wartości numerycznej.

2.4. Analiza skupień

Analiza skupień (ang. *clustering*) to popularna technika eksploracji danych polegająca na dokonaniu takiego partycjonowania zbioru danych wejściowych, które maksymalizuje podobieństwo między obiektami przydzielonymi do jednej grupy i, jednocześnie, minimalizuje podobieństwo między obiektami przypisanymi do różnych grup. Sformułowanie problemu przypomina problem klasyfikacji, jednak należy podkreślić istotne różnice. Analiza skupień jest techniką uczenia bez nadzoru, stąd nieznane jest „poprawne" przypisanie obiektów do grup, często nie jest znana nawet „poprawna" liczba grup. Jeśli porównywane obiekty leżą w przestrzeni metrycznej, wówczas do określenia stopnia podobieństwa między obiektami wykorzystuje się funkcję odległości zdefiniowaną w danej przestrzeni [SJ99]. Zaproponowano wiele różnych funkcji odległości, do najpopularniejszych należy rodzina odległości Minkowskiego (odległość blokowa, odległość euklidesowa, odległość Czebyszewa), odległość Hamminga (wykorzystywana dla zmiennych zakodowanych binarnie), odległość Levenshteina [Lev65] (zwana odległością edycji), czy popularna w statystyce odległość Mahalanobisa. W przypadku, gdy porównywane obiekty nie leżą w przestrzeni metrycznej, zazwyczaj definiuje się specjalne funkcje określające stopień podobieństwa między obiektami. Specjalizowane funkcje podobieństwa istnieją dla wielu typowych dziedzin zastosowań, takich jak porównywanie stron internetowych, porównywanie sekwencji DNA, czy porównywanie danych opisanych przez atrybuty kategoryczne. Metody analizy skupień najczęściej dzieli się na metody hierarchiczne i metody partycjonujące. Pierwsza klasa metod dokonuje iteracyjnego przeglądania przestrzeni i w każdej iteracji buduje grupy obiektów podobnych na podstawie wcześniej znalezionych grup. Rozróżnia się tutaj metody aglomeracyjne (w każdej iteracji dokonują złączenia mniejszych grup) i metody podziałowe (w każdej iteracji dokonują podziału wybranej grupy na mniejsze podgrupy). Druga klasa metod analizy skupień to metody partycjonujące, które od razu znajdują docelowe grupy obiektów. Do najbardziej znanych algorytmów analizy skupień należą

algorytmy k-średnich [Har75], samoorganizujące się mapy [Koh00], CURE [GRS98] (ang. *Clustering Using REpresentatives*), Chameleon [KHK99], Cobweb [Fis87], i wiele innych.

2.5. Odkrywanie cech

Wiele przetwarzanych zbiorów danych charakteryzuje się bardzo dużą liczbą wymiarów (atrybutów). Niczyjego zdziwienia nie budzą tabele z danymi wejściowymi zawierające setki atrybutów kategorycznych i numerycznych. Niestety, efektywność większości metod eksploracji danych gwałtownie spada wraz z rosnącą liczbą przetwarzanych wymiarów. Jednym z rozwiązań tego problemu jest wybór cech [AD92,Kit78,LM98] (ang. *feature selection*) lub odkrywanie cech [YP97] (ang. *feature extraction*). Pierwsza metoda polega na wyselekcjonowaniu z dużej liczby atrybutów tylko tych atrybutów, które posiadają istotną wartość informacyjną. Druga metoda polega na połączeniu aktualnie dostępnych atrybutów i stworzeniu ich liniowych kombinacji w celu zmniejszenia liczby wymiarów i uzyskania nowych źródeł danych. Wybór i generacja nowych atrybutów może odbywać się w sposób nadzorowany (wówczas wybierane są atrybuty, które umożliwiają dyskryminację między wartościami atrybutu decyzyjnego), lub też bez nadzoru (wówczas najczęściej wybiera się atrybuty powodujące najmniejszą utratę informacji).

3. Architektura Oracle Data Mining

Oracle Data Mining (ODM) jest opcją serwera bazy danych Oracle w wersji Enterprise Edition. ODM nie wymaga instalacji żadnych dodatkowych komponentów. W skład ODM wchodzi silnik eksploracji (ang. *mining engine*) oraz dwa interfejsy programistyczne: Java API i PL/SQL API. Silnik eksploracji to zbiór klas języka Java oraz pakietów i procedur PL/SQL umieszczonych w schemacie użytkownika DMSYS, który pełni funkcję administratora opcji ODM. Cały proces eksploracji odbywa się w oparciu o dane przechowywane bezpośrednio w schematach użytkowników bazy danych. Poza programistycznym dostępem do metod, modeli i wyników eksploracji, użytkownicy ODM mogą też korzystać z graficznego narzędzia Oracle Data Miner, które ułatwia definiowanie poszczególnych zadań eksploracji, zarządzanie modelami i odkrytymi wzorcami, oraz wizualizację wyników.

Najważniejszą cechą ODM jest ścisła integracja procesu odkrywania wiedzy z systemem zarządzania bazą danych. Większość komercyjnie dostępnych systemów eksploracji danych wykorzystuje relacyjną bazę danych tylko jako źródło danych. Analizowane dane są pobierane z bazy danych (co najczęściej wiąże się z wysokim kosztem transferu bardzo dużych wolumenów danych), a następnie przetwarzane i analizowane w zewnętrznym narzędziu. ODM dokonuje wszystkich procesów składających się na odkrywanie wiedzy wewnątrz relacyjnej bazy danych. Oznacza to, że selekcja danych, przygotowanie i transformacja danych, tworzenie modelu i wreszcie zastosowanie modelu odbywają się w środowisku systemu zarządzania bazą danych. Daje to twórcom aplikacji niepowtarzalną możliwość integracji technik eksploracji z istniejącymi aplikacjami bazodanowymi. Dodatkowo, umożliwia to standaryzację stosowanych rozwiązań, zapewnia skalowalność algorytmów, oraz gwarantuje możliwość łatwej pielęgnacji aplikacji.

3.1. Metody wspierane przez ODM

ODM wspiera zarówno metody uczenia nadzorowanego, jak i uczenia bez nadzoru. Najpopularniejszą metodą uczenia nadzorowanego jest bez wątpienia klasyfikacja. ODM zawiera cztery algorytmy klasyfikujące: naiwny klasyfikator Bayesa, adaptatywną sieć Bayesa, algorytm indukcji drzew decyzyjnych, oraz algorytm SVM. Pokrótce omówimy te algorytmy.

Naiwny klasyfikator Bayesa jest bardzo prostym, a jednocześnie efektywnym w praktyce klasyfikatorem. Podstawą działania klasyfikatora jest twierdzenie Bayesa, które określa prawdopodobieństwo warunkowe hipotezy h_i przy zaobserwowaniu danych D jako $P(h_i|D)=P(D|h_i)*P(h_i)/P(D)$.

Jeśli przyjąć, że h_i reprezentuje przypisanie do *i*-tej klasy, wówczas prawdopodobieństwo przypisania obiektu D do *i*-tej klasy można znaleźć na podstawie prawdopodobieństwa *a posteriori* $P(D|h_i)$, reprezentującego prawdopodobieństwo posiadania przez D pewnych cech jeśli D rzeczywiście należy do *i*-tej klasy. Prawdopodobieństwo to określa się na podstawie danych zawartych w zbiorze trenującym poprzez analizę cech obiektów rzeczywiście należących do *i*-tej klasy, zaś dodatkowym uproszczeniem jest założenie o warunkowej niezależności atrybutów (tzn. założenie, że w ramach danej klasy rozkład wartości każdego atrybutu jest niezależny od pozostałych atrybutów). W zastosowaniach praktycznych to założenie jest często naruszane, jednak okazuje się, że fakt ten nie ma znaczącego ujemnego wpływu na jakość i dokładność klasyfikacji. Podstawową zaletą naiwnego klasyfikatora Bayesa jest prostota i szybkość, tworzenie i wykorzystanie modelu są liniowo zależne od liczby atrybutów i obiektów.

Adaptatywna sieć Bayesa to algorytm probabilistyczny, który generuje model klasyfikacji w postaci zbioru połączonych cech (ang. *network feature*). W zależności od wybranego trybu algorytm może wyprodukować płaski model stanowiący odpowiednik naiwnego klasyfikatora Bayesa (w takim modelu każda cecha połączona będzie zawierać jeden atrybut-predyktor i jedną klasę docelową), model składający się z jednej cechy połączonej (w ramach cechy znajdzie się wiele związanych ze sobą atrybutów-predyktorów, taki model odpowiada drzewu decyzyjnemu), lub model składający się z wielu cech połączonych. Przy wykorzystaniu modelu z jedną cechą połączoną algorytm może zaprezentować model w postaci prostych reguł klasyfikacyjnych. Stanowi to o atrakcyjności metody, ponieważ zwiększa czytelność procesu eksploracji.

Algorytm indukcji drzew decyzyjnych buduje model w postaci drzewa, którego węzły odpowiadają testom przeprowadzanym na wartości pewnego atrybutu, gałęzie odpowiadają wynikom testów, a liście reprezentują przypisanie do pewnej klasy. Algorytm tworzy drzewo decyzyjne w czasie liniowo zależnym od liczby atrybutów-predyktorów. O kształcie drzewa decydują takie parametry jak: kryterium wyboru punktu podziału drzewa, kryterium zakończenia podziałów, czy stopień scalania drzewa. Wielką zaletą drzew decyzyjnych jest fakt, że wygenerowany model można łatwo przedstawić w postaci zbioru reguł, co pomaga analitykom zrozumieć zasady działania klasyfikatora i nabrać zaufania do jego decyzji.

Algorytm SVM (ang. *Support Vector Machines*) może być wykorzystany zarówno do klasyfikacji, jak i regresji. Algorytm SVM dokonuje transformacji oryginalnej przestrzeni w której zdefiniowano problem klasyfikacji, do przestrzeni o większej liczbie wymiarów. Transformacja jest dokonywana w taki sposób, że po jej wykonaniu w nowej przestrzeni obiekty są separowalne za pomocą hiperpłaszczyzny (taka separacja jest niemożliwa w oryginalnej przestrzeni). Głównym elementem transformacji jest wybór funkcji jądra (ang. *kernel function*) odpowiedzialnej za odwzorowanie punktów do nowej przestrzeni. W przypadku regresji algorytm SVM znajduje w nowej przestrzeni ciągłą funkcję, w ε-sąsiedztwie której mieści się największa możliwa liczba obiektów. Algorytmy SVM wymagają starannego doboru funkcji jądra i jej parametrów. Doświadczenia wskazują, że algorytmy te bardzo dobrze się sprawdzają w praktycznych zastosowaniach, takich jak: rozpoznawanie pisma odręcznego, klasyfikacja obrazów i tekstu, czy analiza danych biomedycznych. Implementacja algorytmów SVM w ODM posiada kilka interesujących cech. Oferuje między innymi mechanizm aktywnego uczenia się, którego celem jest wybór z danych źródłowych tylko najbardziej wartościowych przykładów i trenowanie modelu tylko na wyselekcjonowanych danych wejściowych. W trakcie trenowania algorytm SVM dokonuje automatycznego próbkowania. Wybór funkcji jądra i jej parametrów następuje automatycznie. Algorytm SVM może także zostać wykorzystany do wykrywania osobliwości. Stosuje się wówczas specjalną wersję algorytmu, tzw. SVM z jedną klasą docelową, który pozwala identyfikować nietypowe obiekty.

Poza klasyfikacją, regresją i wykrywaniem osobliwości ODM oferuje jeszcze jedną technikę uczenia nadzorowanego, którą jest wybór cech. Jak wspomniano wyżej, czas tworzenia modelu klasyfikacji najczęściej zależy liniowo od liczby atrybutów. Jest również możliwe, że duża liczba atrybutów wpłynie ujemnie na efektywność i dokładność modelu poprzez wprowadzenie niepożą-

danego szumu informacyjnego. Algorytm wyboru cech (ang. *feature selection*) umożliwia wybór, spośród wielu atrybutów, podzbioru atrybutów które są najbardziej odpowiednie do przewidywania klas docelowych. Wybór cech jest więc często wykorzystywany jako technika wstępnego przetworzenia i przygotowania danych, przed rozpoczęciem trenowania klasyfikatora. ODM oferuje metodę wyboru cech bazującą na ważności atrybutów (ang. *attribute importance*) i wykorzystującą zasadę minimalizacji długości opisu [Ris85] (ang. *minimum description length*). W dużym uproszczeniu, metoda ta traktuje każdy atrybut wejściowy jako możliwy predyktor klasy docelowej a następnie bada liczbę bitów potrzebną do przesłania łącznej informacji o wybranym zbiorze atrybutów i przypisaniach klas docelowych w zbiorze treningowym. Wiadomo, że najkrótszym kodowaniem sekwencji symboli jest takie kodowanie, które najbardziej odpowiada prawdziwym prawdopodobieństwom wystąpienia każdego symbolu. Stąd, zasada minimalizacji długości opisu faworyzuje te podzbiory atrybutów, które nie są nadmiarowe, i jednocześnie pozwalają dobrze przewidzieć wartości atrybutów docelowych.

ODM zawiera również metody pozwalające na szacowanie dokładności klasyfikacji i stosowanie klasyfikatorów do nowych, wcześniej nie widzianych obiektów. Podstawowym narzędziem do oceny dokładności klasyfikacji jest macierz pomyłek (ang. *confusion matrix*). Pozwala ona na łatwe porównanie decyzji klasyfikatora z „poprawnym" przypisaniem obiektów. W trakcie oceny wykorzystuje się tzw. zbiór testujący, w którym znane jest *a priori* rzeczywiste przypisanie obiektów do klas, a jednocześnie zbiór ten nie był wykorzystany do uczenia klasyfikatora. Dzięki macierzy pomyłek można łatwo zauważyć fenomen nadmiernego dopasowania (ang. *overfitting*), w którym klasyfikator przejawia nadmierną skłonność do generalizowania wiedzy uzyskanej ze zbioru uczącego. Innym narzędziem do oceny jakości klasyfikacji jest wyznaczanie krzywych lift. Polega to na porównaniu kwantyli zawierających klasyfikowane obiekty posortowane według pewności klasyfikacji z losowymi kwantylami obiektów. Stosunek liczby obiektów należących do docelowej klasy w każdej parze kwantyli wyznacza kolejne wartości lift. Mechanizmem podobnym do krzywych lift jest mechanizm krzywych ROC [PF97] (ang. *Receiver Operating Characteristics*). W tym przypadku krzywa reprezentuje stosunek liczby poprawnych przewidywań klasy docelowej do liczby niepoprawnych przewidywań klasy.

Stosowanie modelu może odbywać się za pomocą specjalnego modułu – Scoring Engine (SE). Jest to opcjonalny moduł ODM umożliwiający zapisanie modelu w bazie danych i stosowanie modelu do klasyfikacji nowych danych. W bazie danych, w której zainstalowany jest tylko SE, nie można budować żadnych modeli. Wyłączenie SE jako osobnego modułu ma na celu zapewnienie, że produkcyjna baza danych nie będzie służyć jako środowisko budowania i trenowania modeli eksploracji danych, lecz będzie konsumentem modeli stworzonych w specjalnie do tego przeznaczonym środowisku. Należy pamiętać, że eksploracja danych jest procesem kosztownym obliczeniowo i budowanie modeli stanowi istotne obciążenie dla systemu zarządzania bazą danych. Przenoszenie modeli między środowiskiem rozwojowym i produkcyjną bazą danych odbywa się albo za pomocą importu/eksportu schematu lub całej bazy danych za pomocą narzędzia Oracla Data Pump, albo wykorzystując specjalizowane procedury PL/SQL (`DBMS_DATA_MINING.export_model`, `DBMS_DATA_MINING.import_model`) lub klasy języka Java (`javax.datamining.ExportTask`, `javax.datamining.ImportTask`).

Metody uczenia bez nadzoru wspierane przez ODM obejmują algorytmy analizy skupień, odkrywania asocjacji, oraz ekstrakcji cech. Zaimplementowano dwa algorytmy hierarchicznej analizy skupień. Ulepszona wersja popularnego algorytmu k-średnich dokonuje hierarchicznego partycjonowania zbioru danych wejściowych. W każdym kroku następuje podział wybranego węzła (węzła o największym rozmiarze lub węzła o największej wariancji) na dwa podwęzły. Po podziale następuje ponowne wyznaczenie centroidów wszystkich węzłów. Algorytm zatrzymuje się po uzyskaniu k węzłów, które reprezentują docelowe grupy obiektów podobnych. Dla każdej grupy wyznaczany jest centroid grupy, histogramy wszystkich atrybutów wewnątrz grupy, oraz reguła opisująca grupę w postaci zbioru hiperpłaszczyzn. Po wygenerowaniu modelu nowe obiekty mogą być za jego pomocą przypisywane do grup, a zaletą modelu jest to, że przypisanie obiektu do gru-

py ma charakter probabilistyczny. Podstawową wadą algorytmu k-średnich jest to, że jakość uzyskanego modelu zależy przede wszystkim od wybranej liczby *k* docelowych grup, a określenie poprawnej liczby grup *a priori* jest trudne. Drugim algorytmem analizy skupień zaimplementowanym w ODM jest algorytm ortogonalnego partycjonowania O-Cluster. Algorytm ten dokonuje rzutowania wszystkich obiektów na ortogonalne osie odpowiadające atrybutom wejściowym. Dla każdego wymiaru wyznaczane są histogramy, które następnie są analizowane w poszukiwaniu obszarów mniejszej gęstości. Dane są partycjonowane za pomocą hiperpłaszczyzn przecinających osie atrybutów w punktach mniejszej gęstości. Docelowa liczba grup wyznaczana jest automatycznie na podstawie charakterystyki danych. W przeciwieństwie do algorytmu k-średnich, algorytm O-Cluster nie tworzy sztucznych grup w obszarach o jednostajnej gęstości. W obu algorytmach obecność osobliwości może znacznie pogorszyć wynikowy model, zaleca się więc wstępne usunięcie osobliwości przed rozpoczęciem analizy skupień.

ODM implementuje najpopularniejszy algorytm odkrywania reguł asocjacyjnych Apriori. Specyficzną cechą problemu odkrywania asocjacji jest obecność danych o bardzo małej gęstości (tzn. danych, w których niewielka liczba atrybutów, przeważnie mniej niż 10%, jest niepusta w każdym obiekcie). Dane o dużej gęstości powodują gwałtowny wzrost rozmiaru modelu, szczególnie w przypadku niskich wartości parametru minimalnego wsparcia. ODM rozbija problem odkrywania reguł asocjacyjnych na dwa etapy. W pierwszym etapie znajdowane są wszystkie kombinacje wartości atrybutów występujące wystarczająco często w danych wejściowych. W drugim etapie znalezione zbiory są wykorzystywane do wygenerowania reguł asocjacyjnych. Należy zaznaczyć, że ODM wspiera tylko i wyłącznie generowanie reguł zawierających pojedynczy element w następniku.

Ekstrakcja cech jest zaimplementowana w ODM w postaci algorytmu NMF (ang. *Non-Negative Matrix Factorization*). Polega on na przybliżeniu macierzy V (zawierającej obiekty i wartości atrybutów) za pomocą dwóch macierzy niższego stopnia W i H w taki sposób, że $V \approx W*H$. Macierz W zawiera nowe cechy będące liniową kombinacją oryginalnych cech (atrybutów) zapisanych w macierzy V, przy czym współczynniki liniowych kombinacji są dodatnie. NMF dokonuje przybliżenia macierzy V za pomocą macierzy W i H w sposób iteracyjny, w każdym kroku modyfikując wyznaczone współczynniki. Procedura kończy się po osiągnięciu pożądanego stopnia przybliżenia lub po zadanej liczbie iteracji. Algorytm NMF szczególnie dobrze sprawdza się w przetwarzaniu dokumentów tekstowych, gdzie znalezione liniowe kombinacje atrybutów (słów) odpowiadają zbiorom semantycznie powiązanych słów. Zastosowanie algorytmu NMF prowadzi do zmniejszenia liczby wymiarów analizowanego problemu, i często skutkuje zwiększeniem dokładności i jakości generowanych modeli.

Oprócz wymienionych wyżej metod i technik eksploracji danych, ODM wspiera również metody wstępnego przetwarzania danych. Pierwszą z nich jest przycinanie (ang. *trimming*), które polega na usunięciu określonej części dolnych i górnych wartości atrybutu. Dyskretyzacja (ang. *binning*) atrybutów ciągłych i kategorycznych jest wymaganym krokiem w wielu metodach eksploracji danych. ODM umożliwia dyskretyzację atrybutu na przedziały o równej szerokości, przedziały o równej głębokości, oraz dyskretyzację typu „top-n". Wreszcie normalizacja umożliwia odwzorowanie atrybutu na dowolną domenę, np. przedział $<0,1>$ lub domenę, w której wartości atrybutów są wyrażone w odchyleniach standardowych od średniej.

ODM umożliwia także przetwarzanie dokumentów tekstowych. Techniki eksploracji przystosowane do przetwarzania dużych ilości tekstu to: odkrywanie reguł asocjacyjnych, klasyfikacja dokumentów tekstowych przy użyciu algorytmu SVM, analiza skupień dokumentów tekstowych przy użyciu ulepszonego algorytmu *k*-średnich, odkrywanie cech za pomocą algorytmu NMF, oraz wykrywanie anomalii w dokumentach tekstowych przy użyciu algorytmu SVM. Rozwiązania dla danych tekstowych zaimplementowane w ODM są niezależne od możliwości oferowanych przez Oracle Text. ODM zawiera także jeden specjalizowany algorytm BLAST (ang. *Basic Local Ali-*

gnment Search Tool), służący do wyszukiwania dopasowania białek w bioinformatycznych bazach danych. Opis tego algorytmu wykracza jednak poza ramy tego artykułu.

3.2. Interfejs PL/SQL

Interfejs PL/SQL składa się z trzech pakietów

- `DBMS_DATA_MINING`: zawiera główne procedury i funkcje odpowiedzialne za eksplorację. W skład pakietu wchodzą metody służące do zarządzania modelami (`CREATE_MODEL`, `RENAME_MODEL`, `DROP_MODEL`), wykorzystywania modelu do nowych danych (`APPLY`, `RANK_APPLY`), odczytywania informacji o modelu (`GET_MODEL_DETAILS`, `GET_MODEL_SETTINGS`, `GET_MODEL_SIGNATURE`), oceniania jakości modelu (`COMPUTE_CONFUSION_MATRIX`, `COMPUTE_LIFT`, `COMPUTE_ROC`), oraz eksportowania i importowania modelu (`EXPORT_MODEL`, `IMPORT_MODEL`). W przypadku wykorzystania interfejsu PL/SQL parametry wejściowe dla algorytmów umieszcza się w tabeli o określonej strukturze, nazwy parametrów są przekazywane jako zmienne pakietowe.
- `DBMS_DATA_MINING_TRANSFORM`: zawiera pakiety pomocnicze służące do wstępnego przetworzenia danych za pomocą przycinania, dyskretyzacji, normalizacji, oraz uzupełniania wartości pustych. Przy każdej metodzie schemat postępowania jest identyczny: w pierwszym kroku tworzona jest tabela służąca do transformacji (`CREATE`), następnie tabela ta jest wypełniana danymi stanowiącymi definicję transformacji (`INSERT`), wreszcie tworzona jest perspektywa przedstawiająca dane po transformacji (`XFORM`). Oryginalne dane nigdy nie zostają zmodyfikowane, wszystkie zmiany są widoczne tylko i wyłącznie poprzez mechanizm perspektyw.
- `DBMS_PREDICTIVE_ANALYTICS`: pakiet przeznaczony dla początkujących użytkowników. Zawiera metodę służącą do oceny ważności atrybutów względem wybranego atrybutu docelowego (`EXPLAIN`), oraz metodę służącą do przewidywania wartości atrybutu na podstawie pozostałych atrybutów, przy założeniu, że w tabeli źródłowej istnieją wiersze z podaną wartością atrybutu docelowego, które zostaną potraktowane jako zbiór trenujący (`PREDICT`). Pakiet umożliwia wykorzystanie klasyfikacji bez jakiejkolwiek znajomości technik eksploracji danych.

Pakiety PL/SQL działają w sposób synchroniczny. Jeśli wymagane jest asynchroniczne budowanie lub testowanie modelu, konieczne staje się wykorzystanie pakietu `DBMS_SCHEDULER`. Oprócz wymienionych wyżej pakietów PL/SQL, ODM oferuje także zbiór nowych funkcji języka SQL. Funkcje języka SQL nie umożliwiają tworzenia i trenowania modeli, umożliwiają jednak zastosowanie modelu do zbioru krotek, odczytanie informacji o prawdopodobieństwie i koszcie przypisania krotki do klasy/grupy, lub odczytanie informacji o cechach odkrytych przez algorytm FE. Funkcje SQL operują bezpośrednio na modelach przechowywanych w bazie danych i umożliwiają bardzo łatwe włączenie eksploracji danych do istniejących aplikacji bazodanowych. Należy dodać, że funkcje języka SQL oraz pakiet `DBMS_PREDICTIVE_ANALYTICS` pojawiły się dopiero w wersji 10.2 ODM.

3.3. Interfejs Java API

Interfejs Java API stanowi alternatywę dla interfejsu PL/SQL i oferuje praktycznie te same możliwości. Na wstępie należy zaznaczyć, że oba interfejsy stały się interoperatywne dopiero w wersji 10.2 ODM, we wcześniejszych edycjach modele stworzone za pomocą Java API były niedostępne z poziomu PL/SQL i *vice versa*. Interfejs Java API w wersji 10.2 został całkowicie zmieniony w porównaniu z wcześniejszymi wersjami. Aktualna wersja ODM Java API jest implementacją standardu Java Data Mining 1.0 rozwijanego pod auspicjami Java Community Process. W skład interfejsu wchodzi zbiór klas umieszczonych w pakietach odpowiadających zada-

niom eksploracji (np. `javax.datamining.association`). Rozszerzenia standardu (takie jak algorytm NMF do ekstrakcji cech, ortogonalny algorytm partycjonowania O-Cluster, adaptatywna sieć Bayesa, metody transformacji, czy rozszerzenia z pakietu `DBMS_PREDICTIVE_ANALYTICS`) umieszczone zostały w pakietach `oracle.dmt.jdm.*`. Podstawową klasą, z której dziedziczą wszystkie klasy obiektów nazwanych, jest klasa `javax.datamining.MiningObject`. Poszczególne klasy obiektów (trwałych i przejściowych) reprezentują kolejno: model, zbiór parametrów tworzenia modelu, zadanie tworzenia modelu, macierz kosztów, metrykę oceny jakości modelu, oraz fizyczny zbiór danych. Zadania tworzenia modelu są wykonywane w Java API w sposób asynchroniczny. ODM Java API do poprawnego funkcjonowania potrzebuje środowiska co najmniej J2SE 1.4.2.

4. Podsumowanie

Eksploracja danych to nowa i niezwykle prężnie rozwijająca się dziedzina. Wiedza odkryta w dużych wolumenach danych może być traktowana jako metadane, oferujące wzbogacony wgląd w dane. Metody eksploracji danych wymagają specjalizowanych narzędzi umożliwiających budowanie modeli, testowanie modeli, stosowanie modeli do nowych danych. Oracle Data Mining jest pierwszym systemem do tego stopnia integrującym eksplorację danych z systemem zarządzania bazą danych. Wykorzystując ODM można przeprowadzić prawie cały proces odkrywania wiedzy, począwszy od selekcji i wstępnego przetwarzania danych źródłowych, aż po wygenerowanie wzorców. Ścisła integracja technik eksploracji danych z bazą danych umożliwia wykorzystanie technik eksploracji w aplikacjach, ułatwia pielęgnację aplikacji, oferuje ogromnie wzbogaconą funkcjonalność aplikacji.

W artykule przedstawiono podstawowe metody eksploracji danych. Z konieczności, opisy mają charakter wybitnie skrótowy, zainteresowany czytelnik jest kierowany do pozycji wymienionych w spisie literatury. Następnie, przedstawiono architekturę ODM, opisano algorytmy dostępne w ODM, oraz zaprezentowano interfejsy programistyczne udostępniające ODM na poziomie aplikacji.

Bibliografia

[AD92] Almuallin H., Dietterich T.G.: Efficient algorithms for identifying relevant features, in Proc. of 9th Canadian Conference on Artificial Intelligence, pp.38-45, Vancouver BC, 1992

[Aha92] Aha D.: Tolerating noisy, irrelevant, and novel attributes in instance-based learning algorithms. International Journal of Man-Machine Studies 36(2), pp.267-287

[AIS93] Agrawal R., Imielinski T., Swami A.: Mining association rules between sets of items in large databases, Proc. of 1993 ACM SIGMOD International Conference on Management of Data, Washington D.C., May 26-28 1993, pp. 207-216, ACM Press, 1993

[AS94] Agrawal R., Srikant R.: Fast Algorithms for Mining Association Rules, Proc. of 1994 International Conference on Very Large Databases VLDB, Santiago de Chile, September 12-15, pp.487-499. Morgan Kaufman, 1994

[AS95] Agrawal R., Srikant R.: Mining sequential patterns, In Proc. of the 11th International Conference on Data Engineering, Taipei, Taiwan, 1995

[BF+84] Breiman L., Friedman J.H., Olshen R.A., Stone C.J.: Classification and regression trees, Wadsworth, 1984

[Big96] Bigus J.P.: Data mining with neutral networks, McGraw Hill, 1996

[BL97] Berry M.J.A., Linoff G.: Data mining techniques for marketing, sales, and customer support, John Wiley, 1997

[Bol04] Bolstad W.M.: Introduction to Bayesian statistics. Wiley-Interscience, 2004

[Bur98] Burges C.J.C.: A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2(2)

[Cen87] Cendrowska J.: PRISM: An algorithm for inducing modular rules. International Journal of Man-Machine Studies 27(4), pp.25-32, 1987

[DLR77] Dempster A., Laird N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B, 39(1):pp.1-38, 1977

[ELL01] Everitt B.S., Landau S., Leese M.: Cluster analysis, Arnold Publishers, 2001

[Fis87] Fisher D.: Knowledge acquisition via incremental conceptual clustering, Machine Learning 2(2): pp.139-172

[FPSU96] Fayyad U.M., Piatetsky-Shapiro G., Smyth P., Uthurusamy R.: Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, 1996

[GP05] Gidofalvi G., Pedersen T.B.: Spatio-temporal Rule Mining: Issues and Techniques, in Proc. of the 7th International Conference on Data Warehousing and Knowledge Discovery DaWaK 2005, Copenhagen, Denmark, 2005

[GRS98] Guha S., Rastogi R., Shim K.: CURE: An Efficient Clustering Algorithm for Large Databases, In Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 73-84, New York, 1998

[Har75] Hartigan J.A.: Clustering algorithms, John Wiley, 1975

[HGC95] Heckerman D., Geiger D., Chickering D.M.: Learning Bayesian networks: the combination of knowledge and statistical data, Machine Learning 20(3): pp.197-243, 1995

[HK00] Han J., Kamber M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000

[HP+00] Han J., Pei J. et al: FreeSpan: frequent pattern-projected sequential pattern mining. Proceedings of the sixth ACM SIGKDD International conference on Knowledge discovery and data mining, Boston, Massachusetts, United States, pp355-359 , 2000

[KHK99] Karypis G., Han E.-H., Kumar V.: CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling, Technical Report, Department of Computer Science, University of Minnesota, Minneapolis, 1999

[Kit78] Kittler J.: Feature set search algorithms, Pattern recognition and signal processing, Sijthoff an Noordhoff, 1978

[Koh00] Kohonen T.: Self-organizing maps, Springer Verlag, 2000

[Lev65] Levenshtein V.I.: Binary codes capable of correcting deletions, insertions and reversals. Doklady Akademia Nauk SSSR, 163(4):845–848, 1965

[LHC05] LHC Computing Grid, http://lcg.web.cern.ch/LCG/index.html

[LIT92] Langey P., Iba W., Thompson K.: An analysis of Bayesian classifiers. In Proc. of 10th National Conference on Artificial Intelligence, San Jose, CA, AAAI Press, pp.223-228, 1992

[LM98] Liu H., Motoda H.: Feature extraction for knowledge discovery and data mining, Springer Verlag, 1998

[MI94] McCord Nelson M., Illingworth W.T.: Practical guide to neural nets, Addison-Wesley, 1994

[ORS98] Ozden B., Ramaswamy S., Silberschatz A.: Cyclic Association Rules, In Proc. 1998 International Conference on Data Engineering (ICDE'98), pp.412-421, Orlando, FL, 1998

[PF97] Provost F., Fawcett T.: Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions, in Proc. of the 3rd International Conference on Knowledge Discovery and Data Mining, Huntington Beach, CA, 1997

[Qui86] Quinlan J.R.: Induction of decision trees. Machine Learning 1(1),pp.81-106

[Qui93] Quinlan J.R.: C4.5: Programs for machine learning. Morgan Kaufman, 1993

[Ris85] Rissanen J.: The minimum description length principle, Encyclopedia of Statistical Sciences vol.5, pp.523-527, John Wiley, 1985

[SA95] Srikant R., Agrawal R.: Mining Generalized Association Rules, In Proc. of the 21st Int'l Conference on Very Large Databases, Zurich, Switzerland, 1995

[SA96a] Srikant R., Agrawal R.: Mining Quantitative Association Rules in Large Relational Tables, In Proc. of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, 1996

[SA96b] Srikant R., Agrawal R.: Mining Sequential Patterns: Generalizations and Performance Improvements, in Proc. of the 5th International Conference on Extending Database Technology, pp.3-17, Avignon, France, 1996

[SJ99] Santini S., Jain R.: Similarity Measures, IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(9), pp.871-883, 1999

[Vap95] Vapnik V.: The nature of statistical learning theory, Springer Verlag, 1995

[Wint05] Winter Corporation TopTen Program, http://www.wintercorp.com

[WF00] Witten I.H., Frank E.: Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, 2000

[YP97] Yang Y., Pedersen J.O.: A Comparative Study on Feature Selection in Text Categorization, in Proc. of the 14th International Conference on Machine Learning ICML97, pp.412-420, 1997

[ZP+97] Zaki M.J., Parthasarathy S., Ogihara M., Li W.: New Algorithms for Fast Discovery of Association Rules, in Proc. of 3rd International Conference on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, California, USA, 1997