VIII Konferencja PLOUG
Kościelisko
Październik 2002

Data Mining w bazie Oracle 9i

Mariusz Byrski
Biuro Matematyki Stosowanej S.C.
mb@bms.krakow.pl

Abstrakt
Szukanie nieoczywistych związków w danych jest przydatne, potrzebne a czasami wręcz niezbędne w wielu dziedzinach. Dlatego coraz większym zainteresowaniem cieszą się metody data mining. Umożliwiają one znalezienie ukrytych powiązań w danych. Odpowiedzią Oracle na te potrzeby jest zaimplementowanie w bazie Oracle 9i mechanizmów data mining. Baza ta udostępnia interfejs programistyczny w języku Java, za pomocą którego można komunikować się z Data Mining Server, dostarczając mu danych, ustawiając parametry, uruchamiając obliczenia i w końcu pobierając wyniki. Interfejs jest tak skonstruowany, aby był prosty w użyciu, a zarazem umożliwiał doświadczonym użytkownikom ustawianie zaawansowanych parametrów obliczeń. W referacie zostaną omówione wymagania, budowa, właściwości oraz zasady tworzenia rozwiązań z wykorzystaniem narzędzi data mining wbudowanych w bazę Oracle 9i.

1. Wstęp

Firmy przetwarzają coraz większe ilości danych. Efektem tego zjawiska jest coraz trudniejsza, a czasami wręcz niemożliwa, ręczna analiza zbieranych informacji. Skutkiem tego może być np. pogorszenie obsługi klienta czy spadek jakości produktu. Rozwiązanie tego problemu oferują matematyczne metody wyszukiwania ukrytych powiązań w danych, zwane Data Mining czy też Eksploracją Danych, których odpowiednie zastosowanie może poprawić pracę w wielu działach firmy.

Korporacja Oracle jest obecna na rynku rozwiązań Data Mining od kilku lat. Zaistniała na nim wskutek nabycia firmy Thinking Machines Corporation, zajmującej się rozwiązaniami opartymi na algorytmach uczących się. Oracle rozszerzył swoją ofertę handlową o stworzoną przez Thinking Machines Corporation aplikację Darwin. Przekazując do użytku nową wersję bazy Oracle 9i, korporacja zdecydowała się zaimplementować w nią mechanizmy Data Mining. Obecnie podstawową różnicą między Darwinem a Data Mining w bazie Oracle 9i jest brak w tej ostatniej algorytmów opartych na sieciach neuronowych, oraz graficznego interfejsu wspomagającego analizy (jest tylko programistyczny interfejs API).

W referacie zostaną omówione algorytmy zaimplementowane w bazie Oracle 9i oraz sposób ich wykorzystania. Ze względu na ograniczoną ilość miejsca, w opisie algorytmów uwzględnione będą tylko ich podstawowe cechy. Omówione zostaną także poszczególne etapy przetwarzania danych w procesie Eksploracji Danych oraz wyniki, jakie otrzymujemy na każdym etapie. Cały opis dotyczy Oracle Data Mining zaimplementowanego w bazę Oracle 9i wydanie 2. Referat został oparty głównie na Oracle 9i Data Mining Concepts Release 2.

2. Algorytmy w bazie Oracle 9i

Ze względu na sposób nauki algorytmy *Oracle Data Mining* (ODM) możemy zaklasyfikować do dwóch grup:

- z nauczycielem (ang. supervised),
- bez nauczyciela (ang. unsupervised).

Pierwszej grupy używamy przede wszystkim do przewidywania wartości na podstawie wcześniej przetworzonej próbki. Druga grupa algorytmów przydatna jest natomiast głównie do szukania ukrytych struktur, relacji lub podobieństw zawartych w danych.

Innym podziałem, jaki możemy zastosować, jest rozgraniczenie ze względu na sposób przetwarzania [2]. Możemy tutaj wyodrębnić następujące grupy:

- klasyfikacja – z nauczycielem (ang. classification),
- analiza skupień – bez nauczyciela (ang. clustering),
- reguły kojarzące – bez nauczyciela (ang. association rules),
- stopień ważności atrybutów – z nauczycielem (ang. attribute importance).

1. 2. Klasyfikacja

Za pomocą tych metod możemy klasyfikować obiekty do dwóch lub większej liczby klas. Odbywa się to w dwóch etapach: w pierwszym, posiadając już sklasyfikowane obiekty, uczymy model, w drugim nauczony wcześniej model automatycznie klasyfikuje nowe obiekty. Każdy obiekt posiada pewną ilość atrybutów opisujących go oraz jeden taki, który przewidujemy (jego wartość określa, do jakiej klasy będzie należał dany obiekt).

Narzędzie to użyteczne jest we wszelkiego typu segmentacjach czy w przewidywaniu przyszłych zachowań. Przykładem jego zastosowania może być wyszukanie tych klientów banku, którzy byliby potencjalnie zainteresowani kredytem mieszkaniowym, w celu skierowania do nich odpowiedniej kampanii reklamowej. Na podstawie posiadanych przez bank historycznych danych o klientach (przychód, wiek, płeć, itd.) korzystających z kredytu, zbudować można model, a następnie za jego pomocą wytypować takich klientów, którzy mogliby być zainteresowani pożyczką.

W tej grupie zostały zaimplementowane trzy algorytmy:

- Adaptive Bayes Network (wspiera drzewa decyzyjne),
- Naive Bayes,
- Model Seeker.

2.1.1. Adaptive Bayes Network

Algorytm ten wspiera drzewa decyzyjne [1, 2, 5]. Oznacza to, że oprócz zaklasyfikowania obiektu do danej klasy, zdefiniuje on jeszcze reguły zrozumiałe dla odbiorcy. Określają one jakie wartości atrybutów musi mieć dany obiekt, aby został zaliczony do konkretnej klasy. Przykładowo taka reguła może brzmieć: „Jeżeli przychód miesięczny klienta wynosi 4 000 zł i klient jest w wieku 30 lat, to może on być zainteresowany kredytem mieszkaniowym".

Algorytm może przewidywać odpowiedź w postaci dwuwartościowej (klient weźmie kredyt, nie weźmie kredytu) lub w postaci wielowartościowej (klient należy do grupy: rozwijającej, stabilnej, nieznaczącej lub upadającej).

Każdy obiekt, oprócz określenia klasy do jakiej należy, posiada dodatkowo zdefiniowane prawdopodobieństwo mówiące o dopasowaniu go do tej klasy.

2.1.2. Algorytm Naive Bayes

Algorytm Naive Bayes do przewidywań używa teorii Bayes'a [2, 4]. Umożliwia on szukanie odpowiedzi na podobne pytania jak w przypadku Adaptive Bayes Network. Nie podaje jednak jasnych i czytelnych reguł zrozumiałych dla odbiorcy i jest szybszy. Może także pracować z wyjściem dwu– lub wielowartościowym. Po przetworzeniu otrzymujemy sklasyfikowane obiekty wraz z odpowiednimi prawdopodobieństwami.

Tabela 1. Porównanie algorytmów Adaptive Bayes Network i Naive Bayes

	Adaptive Bayes Network	**Naive Bayes**
Liczba wierszy	Dowolny rozmiar	Dowolny rozmiar
Liczba atrybutów	Dowolna liczba	Najlepiej gdy mniej niż 200
Szybkość	Niezbyt szybki	Szybszy
Precyzja	Bardziej precyzyjny	Mniej precyzyjny
Typy atrybutów	Wyliczeniowy, numeryczny	Wyliczeniowy, numeryczny
Automatyczne formatowanie danych	Tak	Tak
Atrybut przewidywany	Dwu i wielowartościowy	Dwu i wielowartościowy

2.1.3. Model Seeker

Jest to metoda pomocnicza, ułatwiająca wybór algorytmu, który należy zastosować dla konkretnych danych: Naive Bayes czy Adaptive Bayes Network. Mając zbudowane modele, algorytm ten ocenia poprzez wykonanie procesu *testowania* oraz *wyliczenia wzrostu*, który z nich jest lep-

szy. Automatyczna ocena odbywa się na podstawie ważonej sumy poprawnych przewidywań pozytywnych do wszystkich przewidywań pozytywnych i poprawnych przewidywań negatywnych do wszystkich przewidywań negatywnych.

$$FOM = \frac{WL_{PP}}{(W+1)L_{WP}} + \frac{L_{PN}}{(W+1)L_{WN}}$$

Gdzie:

FOM – współczynnik określający jakość przewidywań,

W – współczynnik podawany przez użytkownika, zdefiniowany jako stosunek kosztu błędnych przewidywań negatywnych do kosztu błędnych przewidywań pozytywnych. (więcej o pojęciu kosztu zob. rozdział 4.2),

L_{PP} – liczba poprawnych przewidywań pozytywnych,

L_{WP} – liczba wszystkich przewidywań pozytywnych,

L_{PN} – liczba poprawnych przewidywań negatywnych,

L_{WN} – liczba wszystkich przewidywań negatywnych.

Współczynnik FOM osiągnie tym większą wartość, im więcej będzie poprawnych przewidywań pozytywnych i negatywnych. Współczynnik W decyduje o tym, które z nich (pozytywne czy negatywne) będą mieć większy wpływ na wynik. Przy W równym 1.0 poprawne przewidywania pozytywne mają takie samo znaczenie dla wyniku jak poprawne przewidywania negatywne. Jeżeli W jest mniejsze od 1.0, to poprawne przewidywania pozytywne mają mniejszy wpływ na wynik, niż poprawne przewidywania negatywne. Jeżeli W jest większe niż 1.0, sytuacja jest odwrotna.

2.2. Analiza skupień

Analiza skupień jest metodą umożliwiającą odnajdywanie nieznanych grup w danych. Grupy takie, zwane dalej skupieniami, zawierają obiekty podobne do siebie. Wynikiem działania tego typu algorytmów są pewne reguły czytelne dla odbiorcy. Przykładowo, jeśli rozpatrujemy zbiór danych bankowych, który posiada dwa atrybuty: przychód oraz wiek, to znalezione skupienie może mieć następującą regułę:

Jeśli przychód ≥ 2000 zł i przychód ≤ 2500 zł

i wiek ≥25 lat i wiek ≤ 35 lat

to skupienie = 10

Po odnalezieniu skupienia, reguły definiujące go wykorzystać można jako punkt wyjścia w procesie budowania nowej oferty marketingowej, skierowanej do klientów wchodzących w jego skład.

Po zidentyfikowaniu skupień generowany jest model Bayes'a [2, 4], który może w przyszłości posłużyć do klasyfikacji nowego zestawu danych.

W ODM zastosowano dwa algorytmy analizy skupień: k-means oraz o-cluster. Podstawowe różnice między tymi algorytmami zawarto w Tabela 2.

Tabela 2. Porównanie algorytmów K-means i O-Cluster

	K-means	**O-Cluster**
Metoda	Distance-based	Grid-based
Liczba wierszy	Dowolny rozmiar, jednak zaleca się, aby cała tabela mieściła się w buforze.	Dobrze, gdy tabela ma więcej niż 1000 wierszy
Liczba atrybutów	Dobrze, gdy mała liczba atrybutów	Dobrze, gdy więcej niż 10
Liczba klastrów	Specyfikowana przez użytkownika	Wykrywana automatycznie
Typy atrybutów	Tylko numeryczne	Numeryczne i wyliczeniowe
Klastrowanie hierarchiczne	Tak	Tak
Automatyczna normalizacja	Tak	Tak

2.3. Reguły kojarzące

Metoda ta umożliwia wyszukiwanie związków między danymi. Najbardziej typowym zastosowaniem tego algorytmu jest tak zwana „analiza koszykowa". Polega ona na wyszukaniu powiązań między towarami kupowanymi przez klienta. Przykładem takiej reguły automatycznie wygenerowanej jest zdanie „70% ludzi, kupujących makaron, kupuje także sos".

Metoda ta działa bez nauczyciela, co oznacza, że nie musimy mieć zestawu uczącego. W rozpatrywanym zbiorze danych oblicza ona dla każdego obiektu, jak często jest on związany z innym obiektem.

Na wyjściu otrzymujemy dla każdej reguły dwa parametry:

- wsparcie (ang. support) – informuje jak często dane towary znajdują się razem w jednym koszyku. Formalnie możemy to zapisać jako $P(A \cap B)$,

  Gdzie:

  A – liczba sprzedanych produktów A

  B – liczba sprzedanych produktów B

- zaufanie (ang. confidence) – zdefiniowane jako prawdopodobieństwo warunkowe zdarzenia A pod warunkiem zajścia zdarzenia B.

Te dwa parametry umożliwiają nam znalezienie najbardziej interesujących reguł.

W bazie Oracle 9i zaimplementowano algorytm Apriori.

2.4. Stopień ważności atrybutów

Algorytmy tego typu umożliwiają automatyczne wyszukanie tych atrybutów, które najmocniej wpływają na wynik. Są one szczególnie użyteczne w sytuacji, gdy posiadamy dużą liczbę tych ostatnich. Wstępna selekcja zmniejsza czas dalszego przetwarzania danych innymi algorytmami.

W bazie Oracle 9i zaimplementowany jest algorytm Predictor Variance.

3. Budowa Oracle Data Mining

Oracle Data Mining składa się z dwóch głównych komponentów:

- Oracle 9i Data Mining API (ODM API),
- Data Mining Server (DMS).

Oracle Data Mining API jest interfejsem programistycznym, pozwalającym na dostęp programom napisanym w Java do *Data Mining Server*.

ODM API bazuje na nowo tworzonym standardzie *Java Data Mining* (JDM). JDM opiera się na kilku standardach takich jak:

- Common Warehouse Metadata (CWM) – opublikowany przez Object Management Group,
- Predictive Model Markup Language (PMML) – opublikowany przez Data Mining Group,
- SQL/MM dla Data Mining – opublikowany przez International Standards Organization.

Data Mining Server jest komponentem wbudowanym w bazę Oracle 9i, wykonującym obliczenia. Udostępnia on także repozytorium, w którym zapisywane są m. in. modele oraz wyniki obliczeń. W praktyce jest to zbiór pakietów procedur oraz tabel.

4. Etapy przetwarzania

W procesie przetwarzania danych w ODM wyróżnić możemy kilka etapów:

- budowa (ang. build),
- test (ang. test),
- wyliczenie wzrostu (ang. compute lift),
- wykorzystanie (ang. apply),
- import modelu za pomocą PMML,
- eksport modelu za pomocą PMML.

Wszystkie typy modeli muszą przejść przez etap *budowy*. Modele klasyfikujące możemy poddać dodatkowo procesowi *testowania* oraz *wyliczenia wzrostu*, w których sprawdzamy ich jakość. Modele *klasyfikujące* oraz *analiza skupień,* w fazie *wykorzystani,* mają możliwość pracy z nowym zestawem danych i przewidywania dla niego wcześniej zadeklarowanego atrybutu.

Tabela 3. Procesy przetwarzania danych dla poszczególnych metod

	Budowa	Test	Wyliczenie wzrostu	Zastosowanie	Import PMML	Eksport PMML
Klasyfikacja	X	X	X	X	Naive Bayes	Naive Bayes
Analiza skupień	X			X		
Reguły kojarzące	X				X	X
Stopień ważności atrybutów	X					

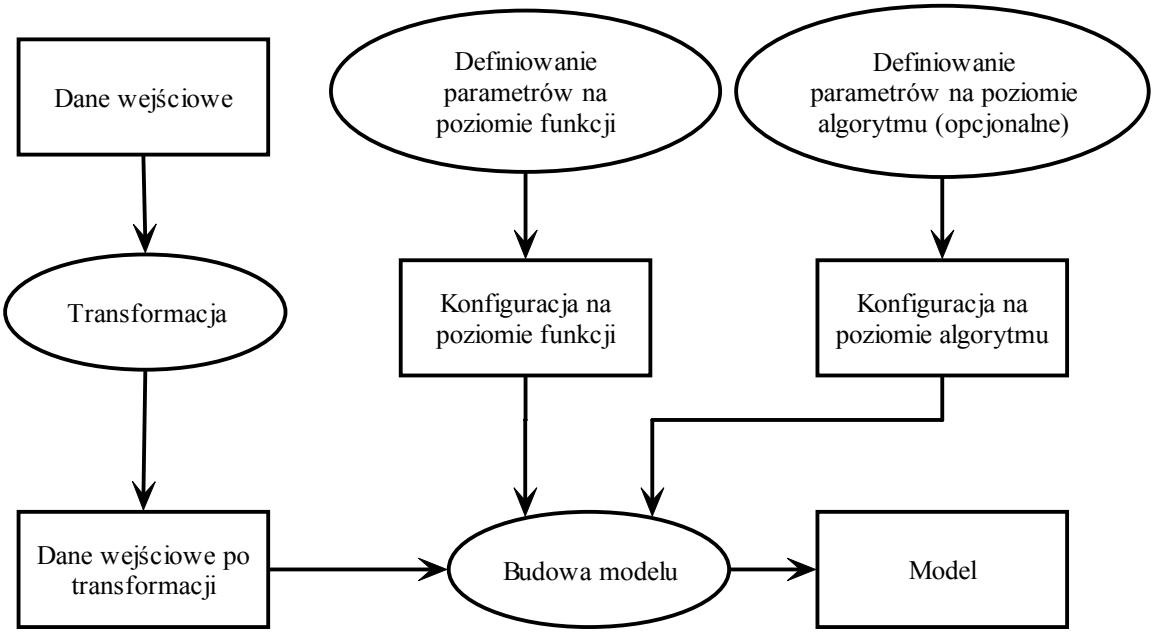
4.1. Budowa modelu

Na tym etapie, w oparciu o zestaw danych, budujemy model. Jest to etap wspólny dla wszystkich algorytmów. Dla większości z nich już teraz otrzymamy pewne wyniki. Mogą to być określo-

ne zestawy reguł (dla algorytmów typu *analiza skupień*, *reguł kojarzących* czy *drzew decyzyjnych*) lub listy istotności atrybutów (dla algorytmów typu *stopień ważności atrybutów*).

Parametry budowy modelu ustawiać można na dwóch poziomach: funkcji oraz algorytmów. Na poziomie funkcji definiujemy pewne dane podstawowe, takie jak: lokalizacja zbioru danych, ich format oraz bardzo ogólne parametry samych algorytmów. Przykładowo możemy określić atrybut wyjściowy dla metod *klasyfikujących,* czy też liczbę skupień dla metod *analizy skupień*. Na poziomie algorytmów ustawiać możemy natomiast parametry bardziej szczegółowe, charakterystyczne dla każdego z nich. Ich podanie nie jest obowiązkowe, Oracle predefiniował ich wartości. W założeniach projektantów, koncepcja taka miała niewątpliwie na celu, aby mniej doświadczeni użytkownicy mogli uruchamiać algorytmy i otrzymywać wyniki bez zapoznawania się z technicznymi detalami ich pracy. Aby móc w pełni wykorzystać to narzędzie, umieć poprawnie interpretować wyniki i dobierać algorytmy do problemu, konieczna jest jednak gruntowna znajomość zasad ich działania.

Model jest budowany przez *Data Mining Server* (DMS), a następnie zapisywany w repozytorium. Można się do niego odwołać poprzez unikalną nazwę.



Rys. 1. Proces budowy modelu

Rys. 1. przedstawia proces budowy modelu. Dane wejściowe są wstępnie przetwarzane, celem doprowadzenia do pewnej standartowej postaci. Dodatkowo definiujemy ustawienia parametrów na poziomie funkcji oraz na poziomie algorytmu (opcjonalnie).

4.2. Testowanie modelu

Dla metod *klasyfikujących* istnieje możliwość uruchomienia procesu testowania jakości prognoz zbudowanego modelu. W tym celu musimy dysponować nowym zbiorem danych, który zawiera rzeczywiste wartości prognoz. Proces *testowania* składa się z dwóch kroków: w pierwszym następuje prognozowanie interesującej nas wartości, w drugim prognozy te porównywane są z poprawnymi danymi. Wynik tej operacji zapamiętywany jest w *macierzy pomyłek* (ang. confusion

matrix). W wierszach macierzy zawarta jest liczba rzeczywistych wartości, w kolumnach – przewidywanych wartości. Rozważmy przykład zapytania, czy klient weźmie kredyt mieszkaniowy. *Macierz pomyłek* może wyglądać następująco:

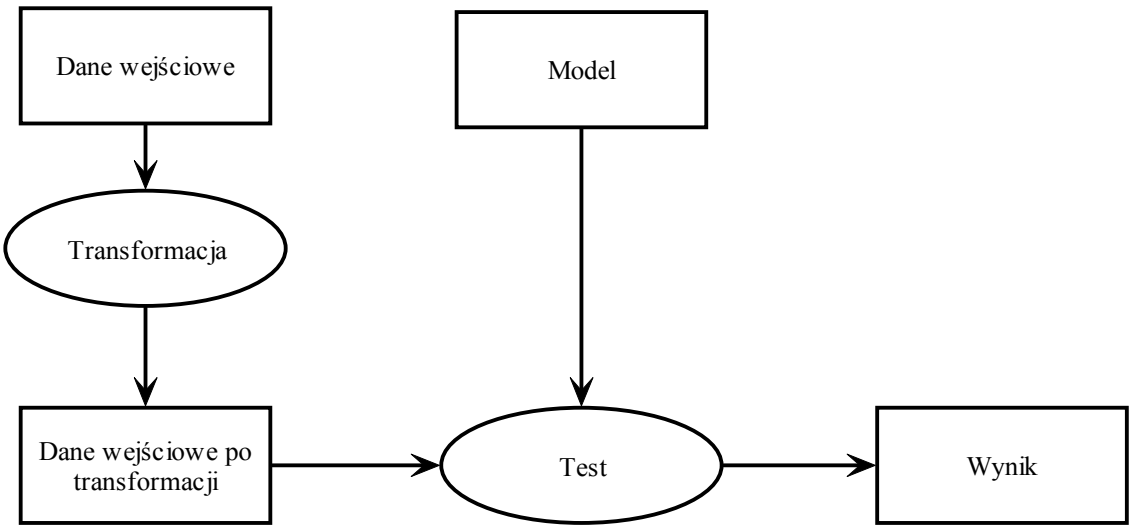
Tabela 4. Macierz pomyłek

		Wartości przewidywane	
		Weźmie	Nie weźmie
Wartości rzeczywiste	Weźmie	516	25
	Nie weźmie	10	725

Należy ją interpretować następująco: w rozpatrywanym zbiorze danych system zidentyfikował 516 klientów jako przyszłych kredytobiorców i osoby te faktycznie skorzystały z kredytu. Następnie znalazł 25 klientów, których określił jako nie zainteresowanych pożyczką, w rzeczywistości jednak zadłużyli się oni. Dla 10 kolejnych system przewidział wzięcie kredytu, jednak klienci ci nie skorzystali z oferty banku. Wreszcie 725 klientów, których system wytypował jako nie zainteresowanych kredytem, w istocie go nie wzięło.

Podsumowując można powiedzieć, że system przewidział poprawnie 516+725=1241 razy, natomiast błędnie 25+10=35 razy. Z tego możemy policzyć błąd przewidywań: 35/1276=0.0274 oraz poprawność przewidywań: 1241/1276=0.9725.

W wielu przypadkach informacja o tak policzonym błędzie jest niewystarczająca. Dzieje się tak w sytuacji, gdy błędne przewidywania negatywne nie mają tej samej wagi, co błędne przewidywania pozytywne. Mówimy wówczas o koszcie błędnych przewidywań pozytywnych i koszcie błędnych przewidywań negatywnych. Przykładowo koszt pomyłki systemu, gdy zaklasyfikował klienta jako zainteresowanego kredytem, a faktycznie klient go nie wziął, zamyka się w kwocie kilku złotych (koszt wysłania listu). Jednak, gdy system popełnił błąd odwrotny, tzn. zaklasyfikował klienta do grupy nie zainteresowanych kredytem, a klient zadłużył się, koszt jest wyraźnie wyższy. Jest to strata, jaką ponieślibyśmy w wyniku nie wzięcia kredytu przez klienta.



Rys. 2. Proces testowania modelu

4.3. Wyliczenie wzrostu

Dla metod *klasyfikacyjnych* obliczyć możemy pewne współczynniki opisujące przetwarzane dane. Na wejście algorytmu, oprócz zestawu danych zawierającego rzeczywistą wartość atrybutu prognozowanego, należy podać wartość, którą uznajemy za pozytywną (np. klienci, którzy wezmą kredyt). Algorytm posortuje wszystkie obiekty począwszy od tych, które mają największe prawdopodobieństwo wyniku pozytywnego, aż do mających najmniejsze prawdopodobieństwo wyniku pozytywnego, a następnie od najmniejszego prawdopodobieństwa wyniku negatywnego, do największego prawdopodobieństwa wyniku negatywnego. W naszym przykładzie będzie to posortowanie klientów od tych, którzy z największym prawdopodobieństwem wezmą kredyt do tych, którzy zrobią to z najmniejszym prawdopodobieństwem, a następnie od tych, którzy z najmniejszym prawdopodobieństwem nie wezmą kredytu do tych, którzy nie zrobią tego z największym prawdopodobieństwem. Tak posortowany zestaw danych jest dzielony na przedziały i dla każdego z nich liczone są następujące współczynniki:

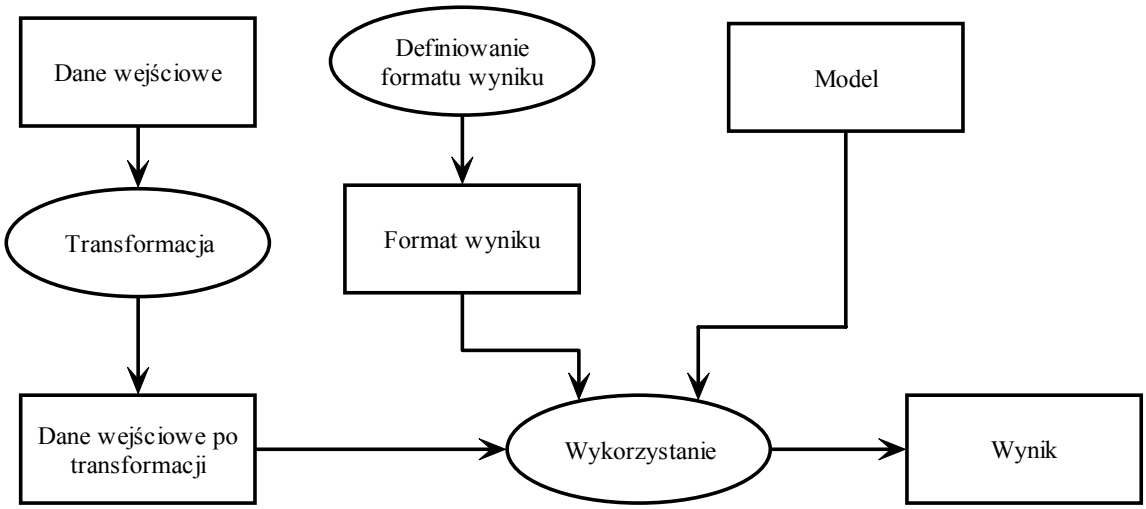
- gęstość atrybutu przewidywanego – jest to liczba rzeczywistych pozytywnych wyników w przedziale do liczby wszystkich obiektów w tym przedziale,
- łączna gęstość atrybutu przewidywanego – jest to gęstość atrybutu przewidywanego policzona dla n pierwszych przedziałów,
- wzrost – współczynnik liczony jako *gęstość atrybutu przewidywanego* policzonego w przedziale przez *gęstość atrybutu przewidywanego* policzonego dla wszystkich danych,
- łączna liczba pozytywnych wyników – jest to liczba rzeczywistych pozytywnych wyników w pierwszych n przedziałach,
- łączna liczba negatywnych wyników – jest to liczba rzeczywistych negatywnych wyników w pierwszych n przedziałach,
- łączny wzrost – jest liczony jako *łączna gęstość atrybutu przewidywanego* przez *gęstość atrybutu przewidywanego* policzonego dla wszystkich danych.

Ogólnie można powiedzieć, że powyższe współczynniki mówią nam jak wiarygodne są prognozy w każdym z przedziałów.

4.4. Wykorzystanie

Proces ten stosujemy dla metod *klasyfikujących* oraz *analizy skupień*. W pierwszym przypadku na wstępie otrzymujemy prawdopodobieństwo przynależności do danej klasy, w drugim prawdopodobieństwo przynależności do danego skupienia.

Dane wejściowe muszą mieć ten sam format, co dane wykorzystane do budowy modelu. Muszą także poddane zostać temu samemu procesowi transformacji.



Rys. 3. Proces zastosowania modelu dla nowych danych

Wynik obliczeń zapisywany jest w tabeli, której format definiuje użytkownik. Mogą się w niej znajdować kolumny z tabeli źródłowej (np. ID), lub kolumny takie jak przewidywana wartość czy prawdopodobieństwo (opcjonalnie). Każdy wiersz tabeli zawiera jeden rozpatrywany obiekt.

5. Transformacja danych wejściowych

Przed dostarczeniem danych do któregoś z procesów: *budowy* modelu, *testowania*, *wyliczenia wzrostu*, *wykorzystania* należy poddać je pewnej obróbce. Dzięki temu można pogrupować razem wartości, które są podobne. Efektem tej operacji jest zmniejszenie liczby różnych wartości w kolumnie. Obróbka ta może odbywać się ręcznie, półautomatycznie lub automatycznie. W Oracle 9i zaimplementowane są cztery algorytmy:

- wyraźne wyspecyfikowanie – użytkownik dokładnie określa, które dane mają znaleźć się w danej kategorii,
- N najczęściej występujących wartości – algorytm ten ma zastosowanie tylko do atrybutów wyliczeniowych. Użytkownik podaje tylko liczbę N, która określa liczbę kategorii. ODM automatycznie znajduje N najczęściej występujących kategorii, a pozostałe przypisuje do jednej dużej kategorii „inne”,
- transformacja przedziałami – algorytm ten ma zastosowanie tylko do atrybutów typu numerycznego. Wartości są sortowane, a następnie rozdzielane na pewną liczbę przedziałów definiowaną przez użytkownika,
- automatyczna transformacja – w przypadku, gdy użytkownik nie chce modyfikować danych ręcznie, może skorzystać z pełnego automatu, który zrobi to za niego.

6. Wymagania

Oracle Data Mining wbudowany jest w bazę Oracle 9i Enterprise Edition [3]. Interfejs API wymaga Java 1.3.1_01, używane jest także JDBC 2.0.

7. Podsumowanie

Inicjatywa włączenia do bazy algorytmów Eksploracji Danych, podjęta przez Oracle wydaje się być ciekawym pomysłem. Data Mining jest nierozerwalnie związany z danymi, a co za tym idzie i z bazami danych. W tej sytuacji projektant aplikacji bazodanowej ma potrzebne algorytmy w „zasięgu ręki". Ich różnorodność daje mu silne narzędzie, które może wykorzystać do bardzo wielu zadań. Ma do dyspozycji zarówno algorytmy prognozujące, jak i wyszukujące ukryte związki w danych. Czytelny i dobrze udokumentowany interfejs API daje możliwość wykorzystania konkretnego algorytmu do określonego, wyspecjalizowanego zadania. Wadą jest jednak brak graficznego interfejsu, który umożliwiłby wstępną, szybką analizę. Zanim zdecydujemy się na implementację konkretnego algorytmu w naszej aplikacji, warto przeanalizować zbiór danych i odpowiedzieć sobie na pytanie, który algorytm i z jakimi parametrami jest dla nas najodpowiedniejszy. Jest to o tyle istotne, że oprócz merytorycznych czynników wyboru algorytmu, znaczenie dla nas mogą mieć także inne, np. czas przetwarzania.

Eksploracja Danych w bazie Oracle jest ciągle rzeczą nową. Uwidoczniło się to zwłaszcza w pierwszej wersji bazy 9i, gdzie liczba algorytmów ograniczona była do dwóch i gdzie występowały problemy z ich poprawną pracą. Prawdopodobnie dlatego też korporacja Oracle nie zdecydowała się na dystrybucję ODM razem z pierwszym wydaniem bazy Oracle 9i. Komponenty Oracle Data Mining należało doinstalować do bazy we własnym zakresie. Inaczej sytuacja przedstawia się w drugim wydaniu bazy Oracle 9i. Tutaj ODM jest standardowo dystrybuowany wraz z bazą Oracle 9i Enterprise Edition. Dodano też kilka nowych algorytmów oraz ułatwiono wykorzystanie istniejących. Wniosek z tych obserwacji nasuwa się jeden: jeżeli ktoś zamierza używać mechanizmów eksploracji danych wbudowanych w bazę Oracle, powinien robić to na bazie Oracle 9i Enterprise Edition, wydanie 2.

Bibliografia

1. Berry M.J.A., Linoff G.: Data Mining Techniques for Marketing, Seles and Customer Support, Wiley Computer Publishing, 1997.
2. Oracle: Oracle 9i Data Mining Concepts Release 2 (9.2), Oracle 2002
3. Oracle: Oracle 9i Data Mining Administrator's Guide Release 2 (9.2), Oracle 2002
4. Feller W.: Wstęp do rachunku prawdopodobieństwa, PWN, 1977.
5. Wyrozumski T., Zastosowanie drzew decyzyjnych w systemach wspomagających prace działów IT, VII Konferencja PLOUG, Zakopane, 2001.