Metody Odkrywania Wiedzy

Nie-całkiem-naiwny klasyfikator Bayesa

Dokumentacja końcowa

Mateusz Jamiołkowski, Michał Uziak

31 maja 2013

Spis treści

1 Wstęp

Celem projektu jest implementacja algorytmu budowy sieci bayesowskiej, która będzie uwzględniać zależności między atrybutami klasyfikowanych obiektów. Na podstawie tak zbudowanego modelu zostanie zaimplementowany alogrytm, którego działanie zostanie porównane z dostępnymi modułami klasyfikacji języka R.

2 Algorytm

Na podstawie analizy dostępnych materiałów zdecydowano, że budowa sieci bayesowskiej zostanie zaimplementowana z wykorzystaniem drzew TAN (tree-augmented naive Bayesian Network).

2.1 Opis działania algorytmu tworzącego sieć bayesowską

Opis algorytmu tworzenia drzew TAN:

1. Obliczenie informacji wzajemnej $I_p(A_i, A_j|C)$ dla każdej pary atrybutów A_i, A_j takich, że $i \neq j$, gdzie $I_p(A_i, A_j|C) = \sum_{c \in C} P(A_i, A_j, c) \log \frac{P(A_i, A_j|c)}{P(A_i|c)P(A_j|c)}$,C - klasa obiektu.

2. Budowa grafu zupełnego nieskierowanego, którego wierzchołkami będą atrybuty $A_1, ..., A_n$ natomiast waga krawędzi pomiędzy atrybutami A_i i A_j będzie równa $I_p(A_i, A_j)$

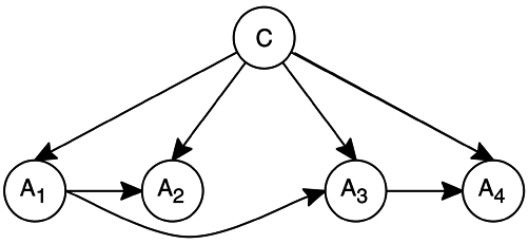
3. Budowa minimalnego drzewa rozpinającego z wykorzystaniem algorytmu Kruskala.

4. Stworzenie drzewa skierowanego, poprzez wybór jednego wierzchołka A_p. Kierunek każdej krawędzi w drzewie zostanie tak dobrany, aby, przechodząc zgodnie z kierunkiem krawędzi, odległość od wierzchołka A_p rosła.

5. Dodanie wierzchołka C(reprezentującego klasę obiektu) do drzewa oraz utworzenie krawędzi skierowanych "od" wierzchołka C pomiędzy tym wierzchołkiem a już istniejącymi wierzchołkami $A_1, A_2...A_n$.

2.2 Opis konstrukcji klasyfikatora

Działanie klasyfikatora będzie polegało na wyborze klasy, która jest najbardziej prawdopodobna dla danego zestawu wartości atrybutów. W odróżnieniu od naiwnego klasyfikatora Bayesa, budowany klasyfikator będzie uwzględniał wzajemne zależności atrybutów. W zbudowanym drzewie krawędź wchodząca do węzła(atrybutu) oznacza zależność od innego węzła(atrybutu). Na rysunku 5.6 przedstawiono przykładowy model drzewa TAN. Zgodnie z przedstawioną zasadą łatwo zauważyć, że atrybut A_2 jest zależny od atrybutu A_1 oraz klasy C.



Rysunek 2.1: Przykładowy model drzewa

Klasyfikator będzie wyznaczał klasę danego obiektu zgodnie z poniższym wzorem:

$$h(x) = \arg\max_{c \in C} P(c|a_1, ..., a_n) = \frac{P(c, a_1, a_2, ..., a_n)}{P(a_1, a_2, ..., a_n)} = \frac{P(c) \cdot \prod_{i=1}^{n} P(a_i|c, a_j)}{P(a_1, a_2, ..., a_n)}$$

Wyrażenie $P(a_i|c, a_j)$ odzwierciedla zależności w zbudowanym drzewie TAN - atrybut A_i jest zależny od A_j
Implementacja algorytmu i powyższy opis zostały stworzone na podstawie artykułu [1]

3 Implementacja

W formie pakietu języka R o nazwie *notSoNaiveBayes* zostały udostępnione następujące funkcje:

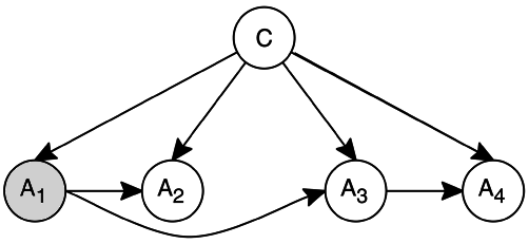
- *notSoNaiveBayes* - funkcja domyślna, która na podstawie danych uczących buduje model, który jest zwracany jako obiekt języka R.
- *predict* - funkcja, która na podstawie modelu uzyskanego w wyniku działania funkcji *notSoNaiveBayes* i wektora (macierzy) atrybutów próbek przypisuje im klasę.

Dodatkowo na potrzeby implementacji zostały napisane następujące funkcje:

- *buildDirectedTree* - funkcja, która na wejściu przyjmuje graf nieskierowany zakodowany w formie macierzy sąsiedztwa oraz numer wierzchołka grafu, który ma zostać korzeniem drzewa jakie powstanie jako argument wyjściowy tej funkcji.
- *kruskal* - jest to implementacja klasycznego algorytmu Kruskala, z jedną różnicą,napisana funkcja szuka maksymalnego drzewa rozpinającego.

3.1 Założenia implementacyjne

Autorzy artykułu [1] na podstawie, którego implementowano klasyfikator Bayesowski oparty o drzewa TAN, nie opisali jednego kroku algorytmu. Mianowicie nie podali informacji jak w punkcie nr 4 algorytmu opisanego w rozdziale 2.1 wybrać „uprzywilejowany" wierzchołek, który będzie zależny tylko od węzła C. Patrz rysunek 3.1, rozważany wierzchołek został oznaczony szarym kolorem.



Rysunek 3.1: Przykładowy model drzewa, w którym jako wierzchołek „uprzywilejowany" został oznaczony kolorem szarym.

Z braku tychże informacji, przeprowadzono testy, aby sprawdzić jak zależy jakość klasyfikacji klasyfikatora, od wyboru wspomnianego wierzchołka. Testy przeprowadzono, na opisanych w rozdziale 4 zbiorach testowych. Okazało się, że wybór tego wierzchołka nie wpływa na jakość klasyfikacji zbudowanego klasyfikatora w żadnym z podanych przypadków. Ta obserwacja pozwala przypuszczać, dlaczego ta informacja została pominięta przez autorów artykułu na podstawie, którego implementowano badany algorytm.

4 Testy

W celu sprawdzenia jakości zbudowanego klasyfikatora jego działanie zostało porównane z innymi algorytmami klasyfikacji dostępnymi w modułach języka R:

- naiwnym klasyfikatorem Bayesa - moduł e1071
- algorytmem kNN - moduł kknn

Do testów zostały wykorzystane dane należące do repozytorium Uniwersytetu Kalifornijskiego w Irvine (Machine Learning Repository, University of California, Irvine), dostępne pod adresem: `http://archive.ics.uci.edu/ml/datasets.html`.
Zbiory testowe:

- chess `http://archive.ics.uci.edu/ml/datasets/Chess`
- nursery `http://archive.ics.uci.edu/ml/datasets/Nursery`
- car `http://archive.ics.uci.edu/ml/datasets/Car+Evaluation`

Testy zostały przeprowadzone za pomocą metody bootstrap. Dla każdego zbioru danych rozlosowano dwie próbki - uczącą i testową. Algorytmy zostały przetestowane dziesięciokrotnie.
Wynikiem była procentowa jakość klasyfikacji, określona za pomocą funkcji języka R:

- *confusionMatrix* - dla algorytmów Bayesowskich
- *simulation* - dla algorytmu knn

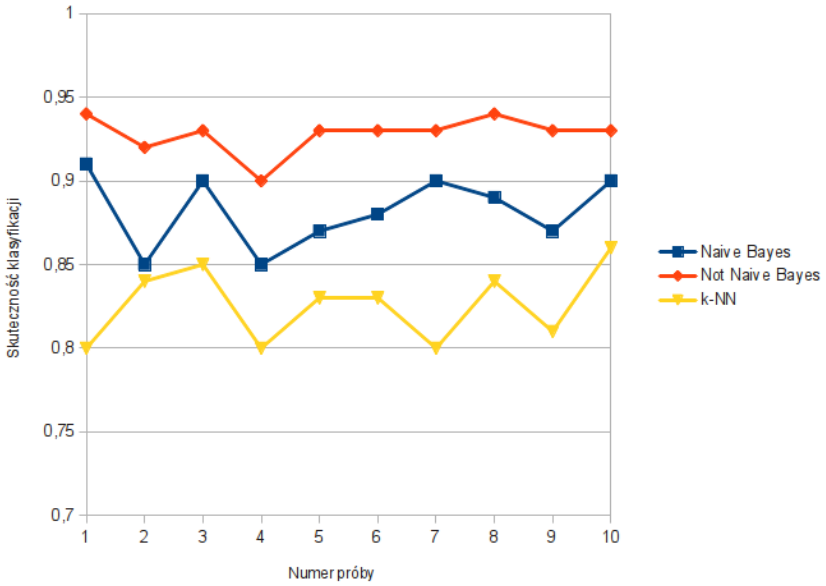
5 Porównanie jakości klasyfikacji

Wykresy przedstawiają porównanie jakości klasyfikacji dla zbiorów testowych. Algorytmy badano kolejno dla losowych danych uczących i testowych w stosunku 2:1.

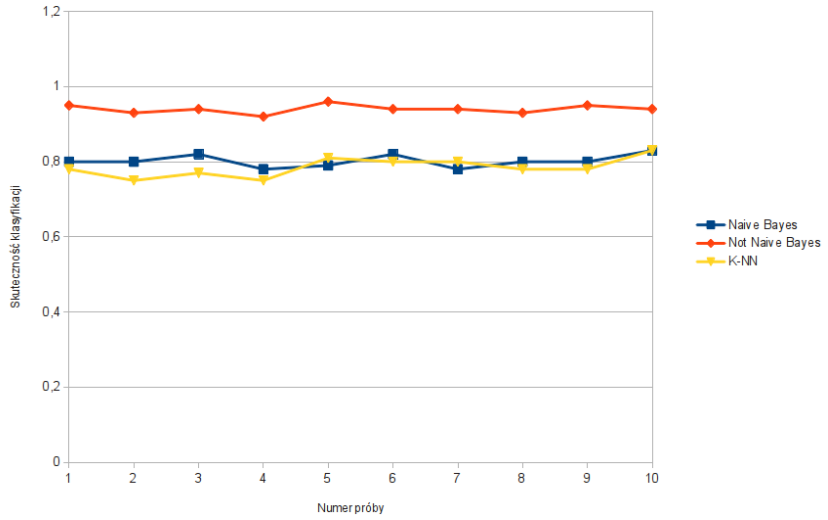
Tablica 5.1: Wyników testów klasyfikacji

Nr próby	Car			Nursery			Chess		
0	Naive	Not Naive	k-NN	Naive	Not Naive	k-NN	Naive	Not Naive	k-NN
1	0,8	0,95	0,78	0,76	0,85	0,86	0,91	0,94	0,8
2	0,8	0,93	0,75	0,75	0,84	0,89	0,85	0,92	0,84
3	0,82	0,94	0,77	0,76	0,85	0,88	0,9	0,93	0,85
4	0,78	0,92	0,75	0,76	0,85	0,88	0,85	0,9	0,8
5	0,79	0,96	0,81	0,76	0,84	0,87	0,87	0,93	0,83
6	0,82	0,94	0,8	0,75	0,85	0,88	0,88	0,93	0,83
7	0,78	0,94	0,8	0,73	0,87	0,88	0,9	0,93	0,8
8	0,8	0,93	0,78	0,75	0,86	0,89	0,89	0,94	0,84
9	0,8	0,95	0,78	0,74	0,85	0,88	0,87	0,93	0,81
10	0,83	0,94	0,83	0,74	0,8	0,87	0,9	0,93	0,86

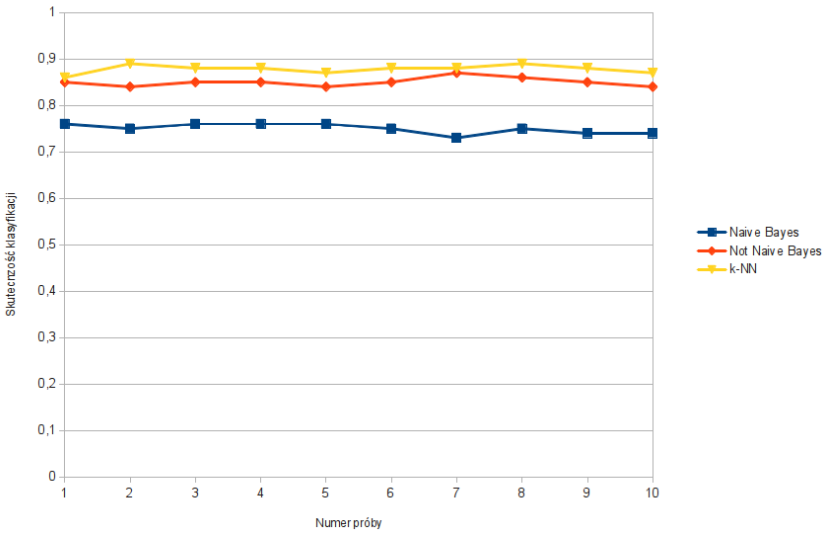
Poniższe wykresy zawierają dane zamieszczone w tabeli 5.1



Rysunek 5.1: Wyniki dla zbioru testowego chess



Rysunek 5.2: Wyniki dla zbioru testowego car



Rysunek 5.3: Wyniki dla zbioru testowego nursery

Na podstawie analizy wykresów można stwierdzić, że zaimplementowany klasyfikator Bayesowski wykazuje się najlepszą skutecznością działania spośród wszystkich przebadanych algorytmów dla zbiorów car i chess, oraz porównywalną jakością do algorytmu k-NN dla zbioru nursery.

5.1 Analiza podstawowych parametrów statystycznych uzyskanych wyników

W celu pogłębionej analizy jakości uzyskanego klasyfikatora zostaną obliczone podstawowe parametry statystyczne wyników uzyskiwanych przez poszczególne algorytmu i zaprezentowane na wykresach pudełkowych dla poszczególnych zbiorów danych. Dla każdej pary zbiór danych-algorytm przeprowadzono 100 prób typu bootstrap, zbiór danych został podzielony na treningowy i testowy w stosunku 2:1.

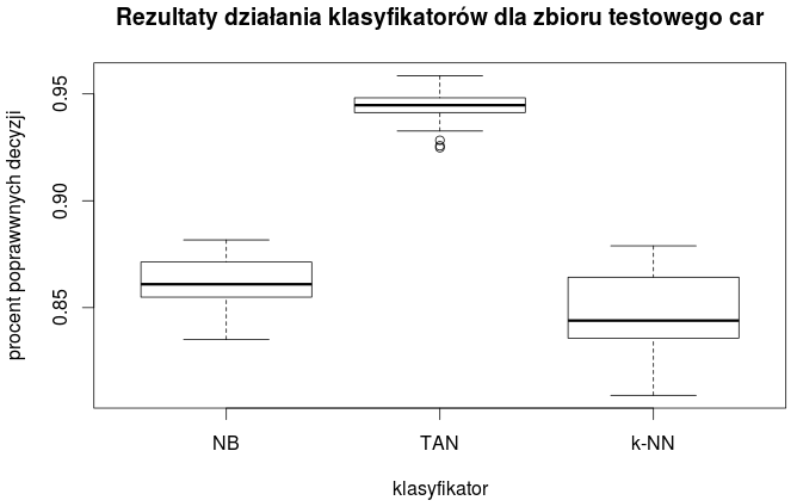
5.1.1 Zbiór danych – chess

Rysunek 5.4: Wykres pudełkowy dla zbioru chess

5.1.2 Zbiór danych – nursery

Rysunek 5.5: Wykres pudełkowy dla zbioru nursery

5.1.3 Zbiór danych – car



Rysunek 5.6: Wykres pudełkowy dla zbioru car

6 Wnioski

Projekt wykazał, że implementacja nie-całkiem naiwnego klasyfikatora Bayesa, opartego o drzewo TAN, wykazuje większą dokładność klasyfikacji od klasycznego klasyfikatora Bayesowskiego. Algorytm kNN osiągał lepszą skutecznością od pozostałych modułów tylko dla zbioru nursery. W pozostałych przypadkach jego jakość była porównywalna z naiwnym klasyfikatorem Bayesa.
Na podstawie analizy wykresów pudełkowych umieszczonych w rozdziale 5.1 można zauważyć, że rozstęp kwartylny dla algorytmu opartego o drzewa TAN jest mniejszy niż pozostałych algorytmów, co jest cechą pozytywną. Pozwala to twierdzić, iż algorytm dla różnych zbiorów danych będzie utrzymywał podobną jakość działania.
W pracy, na której podstawie był implementowany algorytm nie-całkiem naiwnego klasyfikatora Bayesa zostały umieszczone wyniki, jakie otrzymali w wyniku testów ich implementacji algorytmu. Wykorzystali do tego ogólnodostępny zbiór danych chess repozytorium UCI, który również został wykorzystany w tej pracy.
W tabeli poniżej znajduje się porównanie wyników działania „oryginalnej" i „autorskiej" implementacji wspomnianego algorytmu.

Tablica 6.1: Porównanie działania „oryginalna" i „autorskiej" implementacji nie-całkiem naiwnego klasyfikatora Bayesa

	„autorska" impl alg	„oryginalna" impl. alg
średnia jakość działania alg.	0.9268	0.9231
odchylenie standardowe	0.0070	0.0082

Uzyskane rezultaty są porównywalne, pozwala to twierdzić, iż „autorska" implementacja klasyfikatora opartego o drzewa TAN jest poprawna.

Literatura

[1] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayessian Network Classifiers. *Machine Learning*, (29):131–163, 1997.

[2] Kurt Hornik. Package 'rweka'. *RWeka documentation*, 2013.