Naiwny klasyfikator Bayesa

Kamil Łangowski
Wydział Fizyki Technicznej i Matematyki Stosowanej
Politechnika Gdańska

15 czerwca 2021

1 Teoria

1.1 Wstęp

Naiwny klasyfikator Bayesa[1] (ang. *naive Bayes classifier*) jest modelem nadzorowanego uczenia maszynowego. Stanowi rozwinięcie idei klasyfikatora Bayesa pozwalającego na przypisanie klasy pewnej obserwacji, bazując na pojęciu prawdopodobieństwa warunkowego oraz wiążącego się z nim twierdzeniu Bayesa. "Naiwność" naiwnego klasyfikatora Bayesa bierze się z założenia, że wszystkie rozważane zmienne predykcyjne są niezależnie, co w większości przypadków dla rzeczywistych danych nie jest prawdą. Koncepcja naiwnego klasyfikatora Bayesa pomimo swojej prostoty, przy odpowiednich danych, jest bardzo efektywnym narzędziem klasyfikacyjnym, które może dorównywać (a nawet przewyższać) wydajnością tak zaawansowanym modelom jak np. sztuczne sieci neuronowe. Zacznijmy od przedstawienia twierdzenia Bayesa.

1.2 Twierdzenie Bayesa

Jeżeli $\{B_i\}_{i\in I}$ jest przeliczalnym rozbiciem zbioru zdarzeń elementarnych na zdarzenia o dodatnim prawdopodobieństwie i dla zdarzenia A zachodzi $P(A) > 0$, to dla dowolnego $j \in I$ mamy

$$P(B_j|A) = \frac{P(A|B_j)P(B_j)}{\sum_{i\in I} P(A|B_i)P(B_i)}. \tag{1}$$

Powyższą równość nazywamy wzorem (regułą) Bayesa.

1.3 Klasyfikator Bayesa

Załóżmy, że mamy zmienne objaśniające $X = (X^{(1)}, X^{(2)}, \ldots, X^{(d)})$, niech Y będzie zbiorem klas. W naszych rozważaniach ograniczymy się do przypadku klasyfikacji binarnej, tzn. $Y = \{0, 1\}$. Prawdopodobieństwo przynależności dowolnej obserwacji $x = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})$ do klasy $y \in Y$ wyrażamy za pomocą wzoru Bayesa jako

$$P(Y = y|x) = \frac{P(x|Y = y)P(Y = y)}{P(x)}. \tag{2}$$

W praktyce wartość mianownika $P(x)$ jest pomijana, gdyż dla każdej badanej klasy jest niezmienna (pełni rolę elementu skalującego). Stąd (2) możemy zapisać jako

$$P(Y = y|x) \propto P(x|Y = y)P(Y = y), \tag{3}$$

gdzie symbol "$\propto$" oznacza proporcjonalność (w dalszej części zastępujemy symbolem równości). Obserwację x jednoznacznie przypiszemy do klasy $y = 1$ wtedy i tylko wtedy, gdy

$$f_B(x) = \frac{P(Y = 1|x)}{P(Y = 0|x)} = \frac{P(x|Y = 1)P(Y = 1)}{P(x|Y = 0)P(Y = 0)} > 1 \tag{4}$$

Powyższą funkcję f_B nazywamy klasyfikatorem Bayesa lub klasyfikatorem bayesowskim.

[1] Thomas Bayes (1702-1761) – angielski matematyk i duchowny.

1.4 Naiwny klasyfikator Bayesa

W naiwnym klasyfikatorze Bayesa zakładamy (naiwnie), że wszystkie atrybuty z X są niezależne. Wówczas z własności zmiennych niezależnych możemy zapisać

$$P(x|y) = P(x^{(1)}, x^{(2)}, \ldots, x^{(d)}|y) = \prod_{i=1}^{d} P(x^{(i)}|y). \tag{5}$$

Analogicznie do (4) możemy zapisać funkcję naiwnego klasyfikatora Bayesa

$$f_{NB}(x) = \frac{P(Y=1)}{P(Y=0)} \prod_{i=1}^{d} \frac{P(x^{(i)}|Y=1)}{P(x^{(i)}|Y=0)}. \tag{6}$$

Funkcja f_{NB} jest jednym ze sposobów wyrażenia naiwnego klasyfikatora Bayesa, ogranicza się jednak jedynie do klasyfikacji binarnej. Innym podejściem do przypisania obserwacji x etykiety $y \in Y$ jest wykorzystanie zbioru etykiet wyrażenia $P(y)\prod_{i=1}^{d} P(x^{(i)}|y)$ dla jakich osiąga ona maksimum, co zapisujemy jako

$$\hat{y} = \arg\max_{y \in Y} \{P(y) \prod_{i=1}^{d} P(x^{(i)}|y)\}. \tag{7}$$

Powyższą metodę nazywamy metodą maksymalnej wartości a posteriori (ang. *maximum a posteriori method*, w skrócie MAP) lub regułą decyzyjną MAP. Wartość $\hat{y}$ zazwyczaj wyznaczana jest numerycznie np. poprzez zastosowanie metody spadku gradientu.

W zależności od danych naiwny klasyfikator Bayesa postaci (7) może przyjmować inne założenia odnośnie rozkładu prawdopodobieństwa $P(x^{(i)}|y)$, z tego względu możemy wyróżnić:

- gaussowski naiwny klasyfikator Bayesa (ang. *Gaussian Naive Bayes classifier*) – $P(x^{(i)}|y)$ ma rozkład normalny,
- wielomianowy naiwny klasyfikator Bayesa (ang. *Multinomial Naive Bayes classifier*) – $P(x^{(i)}|y)$ ma rozkład wielomianowy,
- naiwny klasyfikator Bayesa o rozkładzie zero-jedynkowym (ang. *Bernoulli Naive Bayes classifier*) – $P(x^{(i)}|y)$ ma rozkład zero-jedynkowy.

1.5 Wydajność naiwnego klasyfikatora Bayesa

Pomimo wymagającego założenia (niezależność predyktorów), które często nie ma pokrycia w rzeczywistości, naiwny klasyfikator Bayesa uznawany jest za jeden z najbardziej optymalnych modeli uczenia maszynowego i czasem stanowi punkt odniesienia dla innych klasyfikatorów. Naiwny klasyfikator Bayesa posiada wiele cech, które są zaskakująco przydatne w praktyce, pomimo iż silne założenie dotyczące niezależności atrybutów często jest nieprawdziwe. Podobnie jak dla każdego klasyfikatora probabilistycznego, który wykorzystuje regułę decyzyjną MAP, klasyfikacja jest poprawna do momentu, w którym przynależność do poprawnej klasy jest bardziej prawdopodobna od innych. Innymi słowy, klasyfikator jest dostatecznie silny, by móc zignorować poważne niedociągnięcia naiwnego probabilistycznego modelu.

2 Przykład

2.1 Opis zbioru

W przykładzie implementacji naiwnego klasyfikatora Bayesa posłużymy się zbiorem danych o nazwie *PimaIndiansDiabetes* z pakietu *mlbench*. Zbiór zawiera informacje na temat cech medycznych kobiet z plemienia Pima[2] oraz faktu, czy dana kobieta choruje na cukrzycę.

2.2 Cel

Zbudujemy model naiwnego klasyfikatora Bayesa, który na podstawie cech medycznych pozwoli zaklasyfikować daną osobę jako potencjalnie chorującą na cukrzycę bądź nie.

2.3 Kod programu

Importujemy niezbędne pakiety:

```
library(mlbench)
library(e1071)
library(OneR)
library(caret)
```

e1071 – zawiera funkcję *naiveBayes*, która jest implementacją NKB w języku R, *mlbench* – zawiera zbiory danych z repozytorium UCI, w tym *PimaIndiansDiabetes*. Tworzymy ramkę danych, odczytujemy 6 pierwszych rekordów, strukturę zbioru oraz podsumowanie:

```
data(PimaIndiansDiabetes)
dane <- PimaIndiansDiabetes

head(dane)
str(dane)
```

```
  pregnant glucose pressure triceps insulin mass pedigree age diabetes
1        6     148       72      35       0 33.6    0.627  50      pos
2        1      85       66      29       0 26.6    0.351  31      neg
3        8     183       64       0       0 23.3    0.672  32      pos
4        1      89       66      23      94 28.1    0.167  21      neg
5        0     137       40      35     168 43.1    2.288  33      pos
6        5     116       74       0       0 25.6    0.201  30      neg

'data.frame': 768 obs. of  9 variables:
 $ pregnant: num  6 1 8 1 0 5 3 10 2 8 ...
 $ glucose : num  148 85 183 89 137 116 78 115 197 125 ...
 $ pressure: num  72 66 64 66 40 74 50 0 70 96 ...
 $ triceps : num  35 29 0 23 35 0 32 0 45 0 ...
 $ insulin : num  0 0 0 94 168 0 88 0 543 0 ...
 $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ pedigree: num  0.627 0.351 0.672 0.167 2.288 ...
 $ age     : num  50 31 32 21 33 30 26 29 53 54 ...
 $ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 1 2 2 ...
```

Zbiór zawiera 768 obserwacji i 9 kolumn, są to m.in.:

- *pregnant* – liczba przebytych ciąż przez badaną kobietę (zmienna numeryczna),
- *pressure* – Rozkurczowe ciśnienie krwi (zmienna numeryczna),

[2]Pima – plemię Indian Ameryki Północnej.

- *mass* – wartość wskaźnika BMI (zmienna numeryczna),
- *diabetes* – wynik badania na cukrzycę: *pos* – pozytywny, *neg* – negatywny (zmienna kategoryczna). Jest to nasza zmienna celu.

Jak widać powyżej, pewne wartości cech, np. zerowa wartość rozkurczowego ciśnienia krwi, wydają się niepoprawne. Z tego względu, w celu zwiększenia dokładności modelu należy dokonać porządkowania niepoprawnych danych, jednym ze sposobów jest usunięcie rekordów, które w kolumnach 2-9 przyjmują wartość 0, wiąże się to z dużą stratą danych, jednak bez wiedzy eksperckiej z danej dziedziny skorygowanie niepoprawnych danych może być bardzo trudne, bądź nawet niewykonalne [**?**]. Usuwamy niepoprawne obserwacje:

```
dane <- data.frame(sapply(dane, as.factor))
dane_fix <- dane[!(apply(dane[,2:9], 1, function(y) any(y == 0))),]
```

W wyniku porządkowania liczba rekordów zmniejszyła się z 768 do 392, zatem straciliśmy około 50% danych. Tworzymy podział zbioru danych na zbiór treningowy i zbiór testowy w stosunku $4:1$, a następnie tworzymy model NKB, w którym każdy z atrybutów uznajemy jako zmienną objaśniającą:

```
podzial <- createDataPartition(dane_fix$diabetes, p = 0.80, list = FALSE)

trening <- dane_fix[podzial, ]
test <- dane_fix[-podzial, ]

model <- naiveBayes(diabetes ~ ., data = trening)
```

Dokonujemy predykcji oraz wyświetlamy tablicę pomyłek:

```
predykcja <- predict(model, test)

eval_model(test$diabetes, predykcja)
```

```
Confusion matrix (absolute):
          Actual
Prediction neg pos Sum
       neg  45   7  52
       pos  14  12  26
       Sum  59  19  78

Confusion matrix (relative):
          Actual
Prediction  neg  pos  Sum
       neg 0.58 0.09 0.67
       pos 0.18 0.15 0.33
       Sum 0.76 0.24 1.00

Accuracy:
0.7308 (57/78)

Error rate:
0.2692 (21/78)

Error rate reduction (vs. base rate):
-0.1053 (p-value = 0.7492)
```

Widzimy, że model zaklasyfikował 57 z 78 przypadków poprawnie. Oznacza to, że jego dokładność kształtuje się na poziomie 73.08%. Dokładność modelu można by zwiększyć np. poprzez normalizację i standaryzację danych lub skorygowanie błędnych danych zamiast ich usunięcia.