Zastosowanie funkcji naiveBayes w języku R

Zadanie polegało na zastosowaniu tytułowego klasyfikatora do pogrupowania win. Dane na których pracowaliśmy zawierały informacje o 178 winach, każda z nich zawierała 14 pozycji, z czego 13 stanowiły atrybuty wina, a pozostałą pozycją była informacja dotycząca grupy do której klasyfikuje się dane wino (podział wynikał z odmiany ang. Cultivar) winogrona z którego powstało wino. Atrybuty wina stanowiły:

1) Alcohol – procentowa zawartość alkoholu

2) Malic acid – zawartość kwasu jabłkowego

3) Ash - popiół

4) Alcalinity of ash – zasadowość popiołu

5) Magnesium – zawartość magnezu

6) Total phenols – całkowita zawartość fenoli

7) Flavanoids – zawartość flawonoidów

8) Nonflavanoid phenols – zawartość nie flanelowych flawonoidów

9) Proanthocyanins – zawartość proantycyjanidy

10)Color intensity – intensywność barwy

11)Hue - odcień

12)OD280/OD315 of diluted wines – wsp. absorbancji wina

13)Proline – zawartość proliny

Metodą klasyfikacji przez nas użytą była funkcja naiveBayes, która opiera swoje działanie na popularnym twierdzeniu Bayesa, które sformułowane jest w następujący sposób: $P(A\backslash X) = \frac{P(X\backslash A)*P(A)}{P(X)}$, gdzie w naszym przypadku:

P(A/X) to prawdopodobieństwo odmiany A przy danych cechach X,

P(X/A) to prawdopodobieństwo danych cech X przy założeniu odmiany A,

P(A) to a priori prawdopodobieństwo odmiany A,

P(X) to a priori prawdopodobieństwo danych cech X.

Naiwność użytego klasyfikatora wynika z jego założenia, że wszystkie cechy są niezależne warunkowo względem odmiany. Uproszczenie to jest często oczywistą niedokładnością, gdyż wiele cech jest mocno skorelowanych, jednakże pozwala ono znacząco uprościć obliczenia i w rzeczywistości algorytm sprawdza się bardzo dobrze w zadaniach klasyfikacyjnych. Implementacja tej metody w R składała się z następujących kroków:

- Podział danych na część treningową i część testową w proporcjach 7:3.

- Trenowanie modelu naiwnego Bayesa na danych treningowych (za pomocą funkcji naiveBayes)

- Przewidywanie odmian na podstawie danych testowych

- Ocena dokładności – konfrontacja danych przewidywanych z danymi wejściowymi

- Przedstawienie rezultatów oraz dokładności modelu za pomocą m.in. macierzy konfuzji

Preprocesing

Pierwszym krokiem podjętym zaraz po wczytaniu danych był ich preprocesing. Nie była to wymagająca część realizacji projektu, gdyż dane były dobrze opracowane, nie zawierały brakujących wyników. Jedyną czynnością której wymagały była konwersja większość danych atrybutów z typu charakter na typ numeric, było to niezbędne do prawidłowego zastosowania funkcji klasyfikującej.

Implementacja klasyfikatora

Na przygotowanych przez preprocessing danych możemy wykonać wyżej opisane kroki wykorzystania metody naiwnego Bayesa do klasyfikacji win.

Zadanie wykonamy dwukrotnie, zaczynając od wykorzystania wszystkich dostępnych w danych cechach, następnie dowolnie wybranych.

1) Wykorzystanie wszystkich cech

Pierwszym krokiem, który należy wykonywać to podział danych. Dokonujemy tego za pomocą funkcji ‘createDataPartition’ z pakietu ‘caret’, która stworzy nam zbiory treningowy i testowy. Zbioru treningowego użyjemy do nauczania naszego modelu, natomiast zbiór testowy zostanie wykorzystany do sprawdzenia jego jakości i skuteczności. Następnie wywołując funkcję naiveBayes tworzymy model przy użyciu wszystkich zmiennych niezależnych do przewidywania zmiennej ‘Class’. Na końcu funkcją ‘predict’ generujemy przewidywania dla zbioru testowego przy użyciu wytrenowanego modelu i tworzymy macierz konfuzji, która obrazuje nam jak nasz model sklasyfikował wina.

Funkcja ‘print(confusion_matrix)’ zwraca następujące informacje:

```
> print(confusion_matrix)
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3
         1 19  0  0
         2  1 18  0
         3  0  1 14

Overall Statistics

               Accuracy : 0.9623
                 95% CI : (0.8702, 0.9954)
    No Information Rate : 0.3774
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9429

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3
Sensitivity            0.9500   0.9474   1.0000
Specificity            1.0000   0.9706   0.9744
Pos Pred Value         1.0000   0.9474   0.9333
Neg Pred Value         0.9706   0.9706   1.0000
Prevalence             0.3774   0.3585   0.2642
Detection Rate         0.3585   0.3396   0.2642
Detection Prevalence   0.3585   0.3585   0.2830
Balanced Accuracy      0.9750   0.9590   0.9872
```

Na samej górze widzimy macierz konfuzji, którą należy odbierać w następujący sposób: suma liczb w pierwszej kolumnie jest równa dwadzieścia, co oznacza że w naszym zbiorze test_data wystąpiło dwadzieścia win o klasie 1. Natomiast w poszczególnych wierszach pierwszej kolumny widać w jaki sposób nasz model dopasował wino z danej klasy. Na przykładzie klasy 1 widzimy, że model na dwadzieścia win dziewiętnaście sklasyfikował poprawnie, natomiast jedno niepoprawnie.

Overall Statistics:

- **Accuracy (Dokładność)**: 0.9623 (96.23%)

Dokładność to odsetek poprawnych klasyfikacji spośród wszystkich próbek.

- **95% CI (Przedział ufności 95%)**: (0.8702, 0.9954)

Przedział ufności dla dokładności modelu wskazuje, że możemy być pewni na 95%, że rzeczywista dokładność modelu mieści się w tym zakresie.

- **No Information Rate (NIR)**: 0.3774 (37.74%)

NIR to dokładność, jaką można by uzyskać, zawsze wybierając najbardziej liczną klasę.

- **P-Value [Acc > NIR]**: < 2.2e-16

Bardzo niski p-value wskazuje, że dokładność modelu jest istotnie wyższa niż NIR.

- **Kappa**: 0.9429

Wskaźnik Kappa mierzy zgodność pomiędzy klasyfikacjami przewidywanymi a rzeczywistymi, uwzględniając losową zgodność. Wartość bliska 1 oznacza wysoką zgodność.

- **Mcnemar's Test P-Value**: NA

Test McNemara nie jest tutaj stosowany, ponieważ nie występują pary, które można by przetestować pod kątem symetrycznych błędów.

Statistics by class:

- **Sensitivity (Czułość)**:

Czułość to zdolność modelu do poprawnej identyfikacji próbek danej klasy.

- **Specificity (Swoistość)**:

Swoistość to zdolność modelu do poprawnego wykluczenia próbek, które nie należą do danej klasy.

- **Pos Pred Value (Dodatnia wartość predykcyjna)**:

Dodatnia wartość predykcyjna to proporcja prawdziwych pozytywnych wyników wśród wszystkich pozytywnych wyników przewidywanych przez model.

- **Neg Pred Value (Ujemna wartość predykcyjna)**:

Ujemna wartość predykcyjna to proporcja prawdziwych negatywnych wyników wśród wszystkich negatywnych wyników przewidywanych przez model.

- **Prevalence (Częstość występowania)**:

Częstość występowania to proporcja próbek danej klasy w zbiorze danych.

- **Detection Rate (Wskaźnik wykrywania)**:

Wskaźnik wykrywania to częstość występowania danej klasy wśród próbek prawidłowo zaklasyfikowanych przez model.

- **Detection Prevalence (Częstość wykrywania)**:

Częstość wykrywania to proporcja próbek przewidzianych jako dana klasa w całym zbiorze danych.

- **Balanced Accuracy (Zrównoważona dokładność)**:

Zrównoważona dokładność to średnia czułości i swoistości dla danej klasy.

Analizując powyższe statystyki charakteryzujące nasz model można wnioskować, że wykazuje on bardzo wysoką dokładność (96.23%) w klasyfikacji próbek wina, co potwierdzają zarówno czułość, jak i swoistość bliskie lub równe 100% dla poszczególnych klas. Wskaźnik Kappa (0.9429) również wskazuje na wysoką zgodność przewidywań modelu z rzeczywistymi klasami, znacznie przekraczającą losową zgodność.

2)Wykorzystanie wybranych cech:

Do stworzenia modelu wykorzystujemy cechy Alcohol, Malic_Acid, Magnesium oraz Proline. Sam proces nie będzie się różnił od poprzedniego niczym poza zmienionymi danymi, dlatego skupimy się od razu na wynikach.

```
> print(confusion_matrix)
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3
         1 18  0  0
         2  0 18  3
         3  2  1 11

Overall Statistics

               Accuracy : 0.8868
                 95% CI : (0.7697, 0.9573)
    No Information Rate : 0.3774
    P-Value [Acc > NIR] : 1.851e-14

                  Kappa : 0.8285

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3
Sensitivity            0.9000   0.9474   0.7857
Specificity            1.0000   0.9118   0.9231
Pos Pred Value         1.0000   0.8571   0.7857
Neg Pred Value         0.9429   0.9688   0.9231
Prevalence             0.3774   0.3585   0.2642
Detection Rate         0.3396   0.3396   0.2075
Detection Prevalence   0.3396   0.3962   0.2642
Balanced Accuracy      0.9500   0.9296   0.8544
```

Jak łatwo zauważyć wyniki odbiegają od wyników dla pierwszego modelu. Wykazują się niższą dokładnością we właściwie każdym parametrze, co oznacza, że ten model jest gorszy niż poprzedni. Wynika to z tego, że wykorzystaliśmy mniej dobrze tworzących model cech.

Niekoniecznie musi to natomiast być reguła, że wykorzystanie mniejszej liczby cech prowadzi do osłabienia modelu. Mogłaby wystąpić sytuacja, że dana cecha jest ściśle związana z konkretną klasą wina i dla różnych wartości cechy jednoznacznie można określić przynależność do danej klasy.

Bibliografia

https://pl.wikipedia.org/wiki/Twierdzenie_Bayesa

http://dmlab.cs.put.poznan.pl/dokuwiki/doku.php?id=naiwny_klasyfikator_bayesa

https://www.rdocumentation.org/packages/soundgen/versions/2.6.0/topics/naiveBayes

https://cran.r-project.org/web/packages/naivebayes/vignettes/intro_naivebayes.pdf