WSI: Laboratorium

Ćwiczenie: Modele bayesowskie

Paweł Skierś, 310895

Semestr zimowy 2021/22

ZADANIE

Celem ćwiczenia jest implementacja naiwnego klasyfikatora Bayesa. Następnie należy wykorzystać stworzony algorytm do stworzenia klasyfikatora dla zbioru danych wine. Zbiór wine dostępny jest w pakiecie scikit-learn (sklearn.datasets.load wine()). Do zbadania jakości należy wykorzystać algorytm n-krotnej walidacji krzyżowej i porównać wynik z otrzymanym na wydzielonym wcześniej zbiorze testowym. Badanie należy powtórzyć dla różnych podziałów na zbiór uczący i testowy.

DOKUMENTACJA ROZWIĄZANIA

Rozwiązanie składa się z dwóch plików: main.py i Bayesian_classifier.py. Klasa implementująca naiwny klasyfikator Bayesa znajduje się w pliku Bayesian_classifier.py. Klasa posiada metody:

- train – trenuje klasyfikator na zadanym zbiorze. Przyjmuje argumenty:
    a. train_data – wartości dzieci (wartości x)
    b. train_labels – wartości rodziców (wartości y)
- predict – przewiduje na podstawie wytrenowanych prawdopodobieństw i wartości dzieci, wartości rodziców. Przyjmuje argumenty:
    a. obserwations – zaobserwowane wartości dzieci
- train_cross_validate – trenuje I wykonuje n-krotną walidację krzyżową na podanych danych. Przyjmuje argumenty:
    a. full_data – wartości dzieci (zbiór trenujący i walidacyjny)
    b. full_labels – wartości rodziców (zbiór trenujący i walidacyjny)
    c. n – ilu krotna ma być walidacja krzyżowa
- evaluate – zwraca dokładność przewidywań sieci. Przyjmuje argumenty:
    a. predictions – przewidywane wartości
    b. real – prawdziwe wartości

EKSPERYMENTY, WYNIKI I WNIOSKI

W ramach eksperymentów na stworzonym algorytmie, przeprowadzono eksperymenty, mające na celu sprawdzenie wpływu zadanego rozmiaru zbioru testowego i liczby zbiorów w wykonywanej walidacji krzyżowej. Poniżej widoczne są wyniki eksperymentów.

Wpływ rozmiaru zbioru testowego

1.

```
test_size = 0.3
n_validation = 4
```

```
Validation scores:  [0.9230769230769231, 0.8076923076923077, 0.9230769230769231, 0.9230769230769231]
Mean validation score:  0.8942307692307694
Test score:  0.8611111111111112
```

```
Validation scores:  [0.8461538461538461, 0.9615384615384616, 0.9615384615384616, 0.8461538461538461]
Mean validation score:  0.9038461538461539
Test score:  0.875
```

```
Validation scores:  [0.9230769230769231, 1.0, 0.8461538461538461, 0.7692307692307693]
Mean validation score:  0.8846153846153846
Test score:  0.9027777777777778
```

2.

```
test_size = 0.1
n_validation = 4
```

```
Validation scores:  [0.9, 0.975, 0.875, 0.875]
Mean validation score:  0.90625
Test score:  0.7777777777777778
```

```
Validation scores:  [0.875, 0.9, 0.925, 0.85]
Mean validation score:  0.8875000000000001
Test score:  0.9444444444444444
```

```
Validation scores:  [0.9, 0.925, 0.9, 0.875]
Mean validation score:  0.9
Test score:  0.8888888888888888
```

3.

```
test_size = 0.9
n_validation = 4
```

```
Validation scores:  [0.5, 1.0, 1.0, 0.75]
Mean validation score:  0.8125
Test score:  0.7391304347826086
```

```
Validation scores:  [1.0, 0.75, 1.0, 0.75]
Mean validation score:  0.875
Test score:  0.7950310559006211
```

Wnioski

Jak widzimy z wyników, wydzielenie bardzo małego zbioru testowy skutkuje tym, że dokładność modelu na zbiorze testowym ma duży rozrzut. Przy małym zbiorze testowym dokładność w dużym stopniu zależy od wybranych do zbioru testowego elementów. Jest to szczególnie widoczne na naszym małym zbiorze danych. Przy bardzo dużym zbiorze testowym natomiast widzimy duży spadek jakości trenowanych modeli. Dzieje się tak, ponieważ duży zbiór testowy skutkuje małym zbiorem treningowym. Model nie będzie więc miał się na czym uczyć i w rezultacie będzie dość średniej jakości. Gdyby nasz zbiór danych był trochę bardziej skomplikowany, to ten efekty byłby jeszcze bardziej widoczny.

Wpływ ilości zbiorów w wykonywanej walidacji krzyżowej

1.

```
test_size = 0.3
n_validation = 4
```

```
Validation scores:  [0.967741935483871, 0.9032258064516129, 0.8387096774193549, 0.8709677419354839]
Mean validation score:  0.8951612903225807
Test score:  0.8703703703703703
```

2.

```
test_size = 0.3
n_validation = 2
```

```
Validation scores:  [0.9354838709677419, 0.8870967741935484]
Mean validation score:  0.9112903225806451
Test score:  0.8703703703703703
```

3.

```
test_size = 0.3
n_validation = 23
```

```
Validation scores:  [0.8, 1.0, 0.6, 1.0, 1.0, 0.8, 0.8, 1.0, 0.8, 1.0, 1.0, 1.0, 1.0, 0.8, 0.8, 0.8, 1.0, 0.8, 0.8, 1.0, 1.0, 0.6, 1.0]
Mean validation score:  0.8869565217391306
Test score:  0.9259259259259259
```

Wnioski

Dla mniejszych ilości zbiorów, wyniki dokładności dla poszczególnych zbiorów (pierwszy wiersz) są bliższe sobie, za to, jako że zbiory walidacyjne są w tych przypadkach podobne wielkością do zbiorów treningowych to same dokładności powinny być mniejsze. To ostatnie nie jest specjalnie widoczne na naszych eksperymentach, gdyż nasz zbiór treningowy jest stosunkowo łatwy. Dla dużych ilości zbiorów, wyniki dokładności dla poszczególnych zbiorów są znacząco od siebie różne, natomiast dokładności powinny być średnio większe. Co więcej im większa jest liczba zbiorów, tym dłużej trwa walidacja, bo trzeba wytrenować więcej modeli.