Selekcja wektorów referencyjnych dla klasyfikatora kNN

Wstęp

Jednym z głównych ograniczeń klasyfikatora kNN jest jego szybkość działania, a w szczególność szybkość podejmowania decyzji. Konieczność obliczania odległości od pojedynczego przypadku badanego do wszystkich przypadków treningowych wymaga bardzo żmudnych obliczeń, co szczególnie uwypukla się w zastosowaniach algorytmu kNN do analizy bardzo dużych zbiorów danych. Jedną z możliwości obejścia tego ograniczenia jest zastosowanie algorytmów redukujących liczbę wektorów referencyjnych (czyli wektorów z oryginalnego zbioru treningowego). Tego typu algorytmy wnoszą jednak szereg dodatkowych korzyści. Bowiem zwykle poprawiają one generalizację np. poprzez usunięcie wektorów odstających. Inne algorytmy starają się odtworzyć granicę decyzji klasycznego klasyfikatora 1NN przy znacznie zredukowanej liczbie wektorów referencyjnych. Przykładami tych dwóch grup algorytmów są algorytmy Edited Nearest Neighbor Rule (ENN) oraz Condensed Nearest Neighbor Rule (CNN).

Algorytm ENN

Autorem algorytmu jest Wilson. Metoda ta usuwa wszystkie wektory stanowiące szum w zbiorze danych treningowych, dzięki czemu powinna podnosić generalizację algorytmu 1NN. Jej sposób działania opiera się na prostej zasadzie, gdzie dla każdego wektora w zbiorze danych treningowych np. x_i wyznaczanych jest k najbliższych sąsiadów. Następnie tych k- najbliższych sąsiadów użytych jest do głosowania. Jeżeli wynikiem głosowania jest błędna klasa, wówczas wektor taki zostaje oznaczony do usunięcia rem_i=1. Można to również zapisać jako próba klasyfikacji wektora x_i przez pozostałe wektory zbioru treningowego po uprzednim usunięciu wektora x_i, czyli $T\backslash x_i$ gdzie T oznacza cały zbiór treningowy, co można zapisać jako $c_i = kNN(T\backslash x_i, x_i)$, gdzie c_i to obliczona etykieta wektora x_i. W drugim etapie procedury ENN jest usunięcie wszystkich wektorów oznaczonych do usunięcia i zwrócenie jako rezultat owego zredukowanego zbioru danych.

Rezultatem działania tego algorytmu jest usunięcie wektorów odstających oraz wektorów brzegowych, a jedyną wartością nastawną jest k (autor zaleca k = 3).

Schemat 1 Schemat algorytmu ENN

Require: $\mathbf{T}$
  $m \leftarrow sizeof(\mathbf{T})$;
  $rem_i \leftarrow 0$;
  for $i = 1 \ldots m$ **do**
    $\bar{C}(\mathbf{x}_i) = k\text{-NN}\ ((\mathbf{T} \setminus \mathbf{x}_i), \mathbf{x}_i)$;
    if $C(\mathbf{x}_i) \neq \bar{C}(\mathbf{x}_i)$ **then**
      $rem_i = 1$;
    end if
  end for
  for $i = 1 \ldots m$ **do**
    if $rem_i == 1$ **then**
      $\mathbf{T} = \mathbf{T} \setminus \mathbf{x}_i$
    end if
  end for
  return $\mathbf{P}$

Algorytm CNN

Metoda CNN należy do grupy przyrostowych, Co oznacza, że do początkowo pustego zbioru obiektów referencyjnych dodawane są nowe wektory. Rozpoczyna ona od losowo wybranego wektora x_1 jako prototypu P, następnie w pętli klasyfikuje pozostałe przypadki czyli należące do zbioru T\P (zbiór treningowy pomniejszony o zbiór P) gdzie T to zbiór treningowy, a P to zbiór wszystkich wybranych obiektów referencyjnych. Jeżeli któryś z wektorów np. x_i należący do zbioru T\P zostaje błędnie sklasyfikowany przez aktualny zbiór prototypów P jest on do niego dodawany $P = P \cup x_i$. Procedura ta jest powtarzana aż wszystkie wektory zostają sklasyfikowane poprawnie.

Schemat 3 Schemat algorytmu CNN

Require: $\mathbf{T}$
  $m \leftarrow sizeof(\mathbf{T})$
  $\mathbf{p}_1 \leftarrow \mathbf{x}_1$
  $flaga \leftarrow$ **true**
  while flaga **do**
    $flaga \leftarrow$ **false**
    for $i = 1 \ldots m$ **do**
      $\bar{C}(\mathbf{x}_i) = k\text{-NN}\ (\mathbf{P}, \mathbf{x}_i)$
      if $\bar{C}(\mathbf{x}_i) \neq C(\mathbf{x}_i)$ **then**
        $\mathbf{P} \leftarrow \mathbf{P} \cup \mathbf{x}_i$;
        $\mathbf{T} \leftarrow \mathbf{T} \setminus \mathbf{x}_i$
        $flaga \leftarrow$ **true**
      end if
    end for
  end while
  return $\mathbf{P}$

Użyta w załączonym programie zmienna *flaga* reprezentuje informacje czy istnieje czy wszystkie wektory zostały poprawnie sklasyfikowane. Jeśli flaga przyjmie wartość *false* wówczas program skończy działanie i zwróci zbiór P.

Do zrobienia w Matlabie

Zaimplementuj funkcje:

P = enn_sel(trening,k)

Oraz

P = cnn_sel(trening)

Gdzie

P – zbiór wektorów treningowych, zapisany zgodnie z wcześniej przyjętym standardem P.X – zbiór atrybutów opisujących, P.Y – zbiór etykiet dla odpowiednich atrybutów.

Trening – zbiór treningowy na podstawie którego ma nastąpić selekcja obiektów referencyjnych

K – parametr algorytmu ENN

W tworzonych funkcjach będą przydatne poniższe wyrażenia umożliwiające dodanie lub usuniecie wiersza lub kolumny z danej zmiennej

A(:,3) = []; - zapis oznacza usunięcie ze zmiennej A kolumny nr 3

A(2,:) = []; - zapis oznacza usunięcie ze zmiennej A wiersza nr 2

x = X(10,:);

P = [P ;x] – zapis oznacza dodanie do zmiennej P nowego wiersza, którym będzie x, czyli 10 wiersz ze zmiennej X

x = X(:,7);

P = [P x] – zapis oznacza dodanie do zmiennej P nowej kolumny, którą będzie x, czyli 7 kolumna ze zmiennej X

Badań dokonaj w oparciu o skrypt:

```
clear; clc;
d = load('  … ');
acc = zeros(1,10);
for I = 1:10
    [trening, test] = podziel(d,0.7);
    P = enn_sel(trening,k);
    M = knn_ucz(P,1);
    wyn = knn_test(M,test);
    acc[i] = dokladnosc(test,wyn);
end;
disp( [ num2str(mean(acc)) '+-' num2str(std(acc))]);
```

Powyższy skrypt umożliwia obliczenie średniej dokładności oraz jej odchylenia standardowego w zależności od tego jak funkcja podziel podzieli się zbiór dany.

Zadania

1) Zaimplementuj przedstawione powyżej algorytmy
2) Zbadaj ich wpływ na dokładność klasyfikacji oraz zdolność redukcji wektorów referencyjnych
3) Badania dokonaj na zbiorach Iris, WBC, Jonosfera oraz Pima
4) Dla algorytmu ENN zbadaj wpływ liczby k na jakość selekcji wektorów referencyjnych
5) Wyniki z zadania 4 czyli dokładność klasyfikacji oraz liczbę wybranych wektorów referencyjnych w stosunku do wszystkich wektorów w zbiorze danych (#P/#T) pokaż na jednym wspólnym wykresie dla każdego zbioru (osobny wykres dla różnych zbiorów danych) , na osi x powinna być liczba k
6) Wyniki z zadania 3 czyli liczbę wektorów referencyjnych w stosunku do liczby wektorów treningowych oraz dokładność umieść na jednym wspólnym wykresie (wszystkie wyniki na jednym wykresie), tak iż oś x będzie odpowiadała liczbie wektorów referencyjnych, a oś Y dokładności.