

Sisteme de recunoașterea formelor – Lab 8

Clasificatorul k-Nearest Neighbor

1. Obiective

Scopul acestui laborator este introducerea clasificatorului k-Nearest Neighbor care, într-un sens, poate fi considerat clasificatorul cel mai elementar. Acesta va fi aplicat pe o problemă de clasificare cu mai multe clase.

2. Fundamente teoretice

Introducere

Un clasificator este definit ca o funcție care returnează clasa unei instanțe. Clasificatorul k-NN poate fi considerat cel mai simplu clasificator deoarece nu construiește un model pentru setul de antrenare. Decizia se ia în funcție de cei mai apropiați k vecini din setul de antrenare. Figura următoare ilustrează acest procedeu unde instanța de test este pătratul albastru din centru înconjurat de instanțe etichetate din setul de antrenare. Interiorul cercului cuprinde 5 vecini pe baza cărora se va face clasificarea.

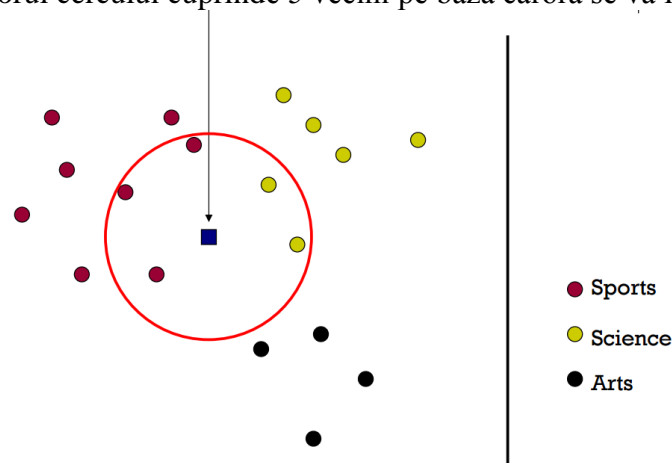


Figure 1. Exemplu de clasificator 5-NN pentru trei clase

Clasificatorul k-NN este un clasificator non-parametric, însemnând că nu construiește un model pentru clasele pe care trebuie să le distingă. Se memorează setul de antrenare în întregime și decizia asupra unei instanțe de test se face online. K-NN poate fi categorizat ca și o metodă de învățare bazată pe instanțe sau învățare leneșă deoarece funcția de decizie este aproximată doar local și decizia asupra instanței se amâna efectiv la clasificare.

Algoritmul de clasificare

Fie setul de antrenare definit ca X , unde X este o matrice de dimensiune $n \times d$. Fiecare linie din X conține un vector de trăsături de dimensiune d denumit X_i , care corespunde unei instanțe de antrenare. Notăm cu y vectorul care conține etichetele pentru clase. Dimensiunea lui y este de $n \times 1$, fiecare instanță de antrenare având o clasă asociată. Elementele din y fac parte din mulțimea $\{1, 2, \dots, C\}$, unde C este numărul de clase.

Pentru o instanță de test x se calculează distanțele dintre x și fiecare exemplu de antrenare:

$$d_i = \text{dist}(x, X_i)$$

Distanțele sunt sortate în mod crescător și cele mai apropiate K instanțe sunt luate în considerare. Fiecare instanță votează pentru clasa lui care este cunoscută din y . Instanța de test va primi clasa cu cele mai multe voturi. O descriere mai formală urmează.

Fie \mathbf{p} permutația care sortează distanțele în ordine crescătoare:

$$d_{p_1} < d_{p_2} < \dots < d_{p_n}$$

Histograma de voturi este construită în următorul mod:

$$\mathbf{h} = \sum_{k=1}^K \mathbf{I}(y_{p_k})$$

unde $\mathbf{I}(y_{p_k})$ este un vector indicator de dimensiune $C \times 1$ care conține 1 pe poziția y_{p_k} și 0 altundeva. Suma acumulează voturile de la cei mai apropiați K vecini. Clasa instanței se alege ca:

$$c = \text{argmax}_k \mathbf{h}_k$$

Există mai multe versiuni ale algoritmului în funcție de tipul de distanță folosită și metodă de votare. De exemplu, voturile pot fi ponderate cu inversa distanței folosind formula:

$$\mathbf{h} = \sum_{k=1}^K \frac{\mathbf{I}(y_{p_k})}{1 + d_{p_k}}$$

unde am adăugat 1 la distanța pentru a evita împărțire la 0 și pentru a obține o pondere de 1 în cazul în care distanța este 0.

Parametrul K controlează câți vecini luăm în considerare. Dacă este egală cu 1 numai vecinul cel mai apropiat este considerat. Creșterea lui K reduce influența zgomotului dar îngreunează separarea claselor. În cazul extrem când $K=n$, setul întreg de antrenare este considerat. Dacă voturile nu sunt ponderate atunci instanța de test este clasificată în clasa cea mai frecventă. De multe ori K este ales să fie un număr impar pentru a rezolva situațiile de egalitate dintre două clase. Pentru a alege valoarea potrivită se evaluează pe un set de validare și se păstrează valoarea care duce la scorul cel mai bun (hyperparameter optimization).

Abordarea prezentată poate fi folosită și pentru a realiza regresie dacă în loc de alegerea clasei se face o sumă ponderată a instanțelor de antrenare. Rata de eroare a clasificatorului k -NN tinde către eroarea ideală Bayes și este mărginită de eroarea Bayes înmulțită cu 2 când $n \rightarrow \infty$ (set de antrenare extins).

Trăsături globale ale imaginii

Imaginile color pot fi caracterizate printr-un vector global de trăsături cu scopul de a realiza clasificarea. Un vector de trăsături globale are o dimensiune fixă și în general descrie statistici globale. De multe ori în aceste descrieri se pierde informația referitoare la aranjarea spațială.

Histograma imaginii poate fi considerată o trăsătură globală pentru a descrie imaginea. Histograma se definește ca vectorul care conține numărul de apariții pentru fiecare nivel de intensitate. În acest caz vectorul are dimensiunea de 256 pentru o imagine pe 8 biți. În general, o histogramă cu m binuri conține numărul de apariții pentru intensitățile din fiecare bin. De exemplu, pentru $m=8$ binuri, binul 0 va conține numărul de intensități cuprinse între 0 și $256/m - 1=31$; binul 1 intensități între 32 and 63; ș.a.m.d.

Histograma unei imagini color se formează prin concatenarea histogramelor pe canalele individuale. Mărimea histogramei rezultante este de $3 \times m$.

Evaluarea clasificatorilor

Pentru a evalua performanța clasificatorilor se folosesc mai multe metrici. **Matricea de confuzie** pentru un set etichetat se poate defini ca matricea care conține în fiecare celulă M_{ij} numărul de instanțe clasificate în clasa i și care au clasa j în realitate. Clasificatorul ideal atribuie la fiecare instanță eticheta corectă și deci asigură valori mari pe diagonala matricei de confuzie. În general valorile din matrice arată ce clase sunt confundate între ele.

Acuratețea unui clasificator pentru un set etichetat se definește ca procentajul de instanțe clasificate corect. Este măsura complementară erorii de clasificare. Nu oferă informație relevantă când clasele nu sunt echilibrate (sunt mai multe instanțe într-o clasă decât în cealaltă). Aceasta este o situație tipică, de exemplu un detector de pietoni trebuie să învețe dintr-un set unde instanțele de fundal sunt mult mai numeroase. Un clasificator care returnează întotdeauna clasa mai frecventă poate să obțină o acuratețe ridicată. În acest caz trebuie să utilizăm alte metrici cum ar fi precizia pentru clasele individuale. Acuratețea se poate calcula din matricea de confuzie:

$$Acc = \frac{\sum_{i=1}^C M_{ii}}{\sum_{i=1}^C \sum_{j=1}^C M_{ij}}$$

3. Detalii de implementare

Sugestie pentru antetul funcției care calculează histograma (*hist* a fost alocată):

```
void calcHist(Mat img, int nr_bins, int* hist)
```

Definim denumirile claselor:

```
const int nrclasses = 6;
char classes[nrclasses][10] =
{"beach", "city", "desert", "forest", "landscape", "snow"};
```

Se alocă spațiu pentru matricea de trăsături și vectorul de etichete:

```
Mat X(nrinst, feature_dim, CV_32FC1);
Mat y(nrinst, 1, CV_8UC1);
```

Citirea imaginilor din clasa c ; se calculează histograma și se introduce ca și o linie în X :

```
int c = 0, index = 0, pos = 0;
while(1){
    sprintf(fname, "train/%s/%06d.jpeg", classes[c], index++);
    Mat img = imread(fname);
    if (img.cols==0) break;

    calcHist(img, nr_bins, hist);

    for(int d=0; d<hist_size; d++)
        X.at<float>(pos, d) = hist[d];
    y.at<uchar>(pos) = c;
    pos++;
}
```

Se alocă matricea de confuzie:

```
Mat C(nrclasses, nrclasses, CV_32FC1);
```

4. Activitate practică

1. Implementați o funcție care extrage histograma dintr-o imagine color cu $3 \times m$ binuri.
2. Citiți toate imaginile din setul de antrenare. Calculați histograma pentru fiecare imagine. Salvați histograma ca și o linie în matricea de trăsături X . Salvați eticheta instanței în vectorul y .
3. Implementați clasificatorul k-NN având ca și intrare o imagine și valoarea lui K .
4. Evaluați clasificatorul pe setul de test prin calcularea matricei de confuzie și a acurateței.
5. Încercați să obțineți rezultate cât mai bune folosind diferite valori pentru numărul de binuri m și pentru numărul de vecini K .
6. Converteți imaginea de intrare într-un alt spațiu de culoare (Luv sau HSV) înaintea calculării histogramei.
7. Opțional, încercați trăsături mai complicate (histograma pe regiuni) sau alte funcții de distanță (Manhattan distance, Euclidean ponderat).

Referințe

[1] Wikipedia article - k-NN classifier

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[2] Andrew Ng - Machine Learning: Nonparametric methods & Instance-based learning

<http://www.cs.cmu.edu/~epxing/Class/10701-08s/Lecture/lecture2.pdf>