Sisteme de recunoaşterea formelor – Lab 8

Clasificatorul k-Nearest Neighbor

1. Obiective

Scopul acestui laborator este introducerea clasificatorului "cei mai apropiaţi *K* vecini" (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 funcţie care returnează clasa unei instanţe. Instanţa sau exemplul este de obicei reprezentat prin vectorul de trăsături. Clasificatorul k-NN poate fi considerat cel mai simplu clasificator deorece nu construieşte un model pentru setul de antrenare. Decizia se ia în funcţie de cei mai apropiaţi *K* vecini din setul de antrenare. Figura următoare ilustrează acest procedeu unde instanţa de test este pătratul albastru din centru înconjurat de instanţe etichetate din setul de antrenare. Interiorul cercului cuprinde 5 vecini pe baza cărora se va face clasificarea. Cercul are rază variabilă şi întotdeauna cuprinde *K* vecini.

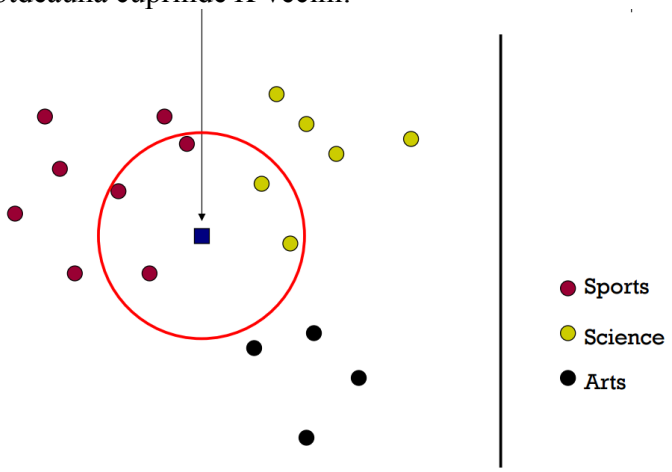

Figure 1. Exemplu de clasificator 5-NN pentru trei clase

Clasificatorul k-NN este un clasificator non-parametric, însemnând că nu construieşte un model pentru clasele pe care trebuie să le distingă. Se memorează setul de antrenare în întregime şi decizia asupra unei instanţe de test se face online. K-NN poate fi categorizat ca şi o metodă de învăţare bazată pe instanţe sau invaţare leneşă deorece funcţia de decizie este aproximată doar local şi decizia asupra instanţei se amâna efectiv la clasificare.

Algoritmul de clasificare

Fie setul de antrenare definit ca *X*, unde *X* este o matrice de dimensiune *nxd*. Fiecare linie din *X* conţine un vector de trăsături de dimensiune *d* denumit X_i, care corespunde unei instanţe de antrenare. Notăm cu $\boldsymbol{y}$ vectorul care conţine etichetele pentru clase. Dimensiunea lui $\boldsymbol{y}$ este de *nx1*, fiecare instanţă de antrenare având o clasă asociată. Elementele din $\boldsymbol{y}$ fac parte din mulţimea *{1,2,...,C}*, unde *C* este numărul de clase.

Pentru o instanţă de test $\boldsymbol{x}$ se calculează distanţele dintre $\boldsymbol{x}$ şi fiecare exemplu de antrenare:

$$d_i = dist(x, X_i)$$

Distanţele sunt sortate în mod crescător şi cele mai apropiate *K* instanţe sunt luate în considerare. Fiecare instanţă votează pentru clasa lui care este cunoscută din $\boldsymbol{y}$. Instanţa de test va primi clasa cu cele mai multe voturi. O descriere mai formală urmează.

Fie **p** permutaţia care sortează distanţele în ordine crescătoare:

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

Histograma de voturi de dimensiune *Cx1* este construită în următorul mod:

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

unde $\mathbf{1}(y_{p_k})$ este un vector indicator de dimensiune *Cx1* care conţine 1 pe poziţia $\boldsymbol{y}_{p_k}$ şi 0 altundeva. Suma acumulează voturile de la cei mai apropiaţi *K* vecini. Clasa instanţei se alege ca:

$$c = argmax_i \boldsymbol{h}_i$$

Există mai multe versiuni ale algoritmului în funcţie de tipul de distanţă folosit şi metoda de votare. De exemplu, voturile pot fi ponderate cu inversa distanţei folosind formula:

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

unde am adăugat 1 la distanţa pentru a evita împărţire la 0 şi pentru a obţine o pondere de 1 în cazul in care distanţa este egală cu 0.

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

Abordarea prezentată poate fi folosită şi pentru a realiza regresie dacă în loc de alegerea clasei se face o sumă ponderată a instanţelor de antrenare. Rata de eroare a clasificatorului k-NN tinde către eroarea ideală Bayes şi este mărginită de eroarea Bayes înmulţită cu 2 când numărul de instanţe tinde către infinit ($n \to \infty$).

Trăsături globale ale unei imagini

Imaginile color pot fi caracterizate printr-un vector global de trăsături cu scopul de a realiza clasificarea lor. Un vector de trăsături global are o dimensiune fixă şi descrie statistici globale despre imagine. De multe ori în aceste descrieri se pierde informaţia referitoare la aranjarea spaţială.

Histograma imaginii poate fi considerată o trăsătura globala pentru a descrie imaginea. Histograma se defineşte ca vectorul care conţine numărul de apariţii pentru fiecare nivel de intensitate. În acest caz vectorul are dimensiunea de 256 pentru o imagine pe 8 biţi. În general, o histogramă cu *m* acumulatoare (eng. bins) conţine numărul de apariţii pentru intensităţile din fiecare acumulator. Se împarte intervalul de [0,255] în *m* bucăţi egale. De exemplu, pentru *m=8* acumulatoare*,* primul va conţine numărul de pixeli cu intensităţi cuprinse între [0,*256/m*)*;* al doilea va conţine numărul de pixeli cu intensităţi între *32* and *63*; ş.a.m.d. Histograma unei imagini color se formează prin concatenarea histogramelor pe canalele individuale. Mărimea histogramei rezultante este de *3 x m*.

Evaluarea clasificatorilor

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

Acurateţea unui clasificator pentru un set etichetat se defineşte ca procentajul de instanţe clasificate corect. Este măsura complementară erorii de clasificare. Nu oferă informaţie relevantă când clasele nu sunt echilibrate (sunt mai multe instanţe într-o clasă decât în cealaltă). Aceasta este o situaţie tipică, de exemplu un detector de pietoni trebuie sa înveţe dintr-un set unde instanţele de fundal sunt mult mai numeroase. Un clasificator care returnează întotdeauna clasa mai frecventă poate să obţină o acurateţe ridicată. În acest caz trebuie să utilizăm alte metrici cum ar fi precizia pentru clasele individuale. Acurateţea 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. Setul de date – recunoaşterea scenelor

Setul de date pentru acest laborator conţine diferite scene. Există 6 clase: plajă, oraş, deşert, pădure, peisaj şi zăpadă. Imaginile pentru fiecare clasă sunt organizate în subdirectoare şi sunt numerotate cu numere cu 6 cifre. Setul este neechilibrat din punctul de vedere al numărului de exemple pentru fiecare clasă care variază între 35 şi 277. Setul de antrenare este compus din 672 imagini, iar setul de testare conţine 85 fişiere. Mai jos afişăm exemple reprezentative pentru fiecare clasă.


plajă


oraş


deşert


pădure


peisaj


zăpadă

4. Detalii de implementare

Sugestie pentru antetul funcţiei 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 spaţiu pentru matricea de trăsături si vectorul de etichete:

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

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

```
int c = 0, fileNr = 0, rowX = 0;
while(1){
      sprintf(fname, "train/%s/%06d.jpeg", classes[c], fileNr++);
      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>(rowX, d) = hist[d];
      y.at<uchar>(rowX) = c;
      rowX++;
}
```

Se alocă matricea de confuzie:

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

5. Activitate practică

1. Implementaţi o funcţie care extrage histograma dintr-o imagine color cu *3xm* acumulatoare.
2. Citiţi toate imaginile din setul de antrenare. Calculaţi histograma pentru fiecare imagine. Salvaţi histograma ca şi o linie în matricea de trăsături *X*. Salvaţi eticheta instanţei corespunzătoare pe aceeaşi linie în vectorul ***y*.**
3. Implementaţi clasificatorul k-NN având ca şi intrare o imagine şi valoarea lui *K*.
4. Evaluaţi clasificatorul pe setul de test prin calcularea matricei de confuzie şi a acurateţei.
5. Încercaţi să obţineţi rezultate cât mai bune folosind diferite valori pentru numărul de acumulatoare *m* şi pentru numărul de vecini *K*. Se poate obţine acurateţe de peste 65%.
6. Convertiţi imaginea de intrare într-un alt spaţiu de culoare (Luv sau HSV) înaintea calculării histogramei.
7. Opţional, încercaţi trăsături mai complicate (histograma pe regiuni) sau alte funcţii de distanţă (Manhattan distance, Euclidean ponderat).

Referinţe

[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