

Classificatore K-NN

Esercizio: Implementare il classificatore K-NN:

```
[labelAssegnate_test error_test] = mio_knn( ds_train, ds_test, k)
```

Traccia:

- La funzione “**DIST(A,B)**” del toolbox reti neurali calcola la distanza euclidea fra ciascuna riga di A e ciascuna riga di B
- Analogamente la funzione prtools DISTM calcola la distanza euclidea (al quadrato) fra ciascuna riga di A e ciascuna riga di B
- La funzione “**sort**” permette di ordinare una matrice e di ricavare gli indici di ordinamento

```

function [labelAssegnate_test error_test] = ...
mio_knn( ds_train, ds_test, k)

nClassi= length(ds_train.lablist);
nPatterns_trn= ds_train.objsize;
nPatterns_tst= ds_test.objsize;

%train_features = ds_train.data
%train_labels = ds_train.nlab

%calcolo distanze
%distanze_eucl(its,itr) = distanza fra il pattern 'its' di
test e quello 'itr' di train
distanze_eucl = dist (ds_test.data, ds_train.data');

%Ordino le distanze per righe e memorizzo gli indici di
ordinamento
[distanze_sort idx_sort]=sort(distanze_eucl,2);
%NB:
% La riga 'idx_sort(its,:) ' contiene l'indice dei vicini del
pattern 'its'
%prendo gli indici dei k più vicini

```

```

idx_primiVicini=idx_sort(:,1:k);

% Estraggo le labels dei k pattern piu' vicini
%

labels_primiVicini = ds_train.nlab(idx_primiVicini);

labelAssegnate_test = zeros(nPatterns_tst,1);
for its=1:nPatterns_tst,
    %classifico il pattern di test i-esimo
    %conto il numero di occorrenze di ciascuna label
    for c=1:nClassi,
        occorrenze(c) = sum(labels_primiVicini(its) == c);
    end
    %prendo il massimo
    [tmp, classe_max] = max(occorrenze+rand(1,nClassi));
    labelAssegnate_test(its) = classe_max;
end

error_test =mean(labelAssegnate_test ~= ds_test.nlab)

```

```
%il ciclo for interno equivale a  
% occorrenze =...  
    sum([ones(nClassi,1)*labels_primiVicini(its)' == ...  
        arrayIndiciClassi],2);%  
%dove  
% arrayIndiciClassi=[1:nClassi]'*ones(1,k);
```

II K-NN nei PR-Tools

help knnc

KNNC K-Nearest Neighbor Classifier

```
[W,K,E] = KNNC(A,K)
```

```
[W,K,E] = KNNC(A)
```

INPUT

A Dataset

K Number of the nearest neighbors (optional; default: K
is
optimized with respect to the leave-one-out error on
A)

OUTPUT

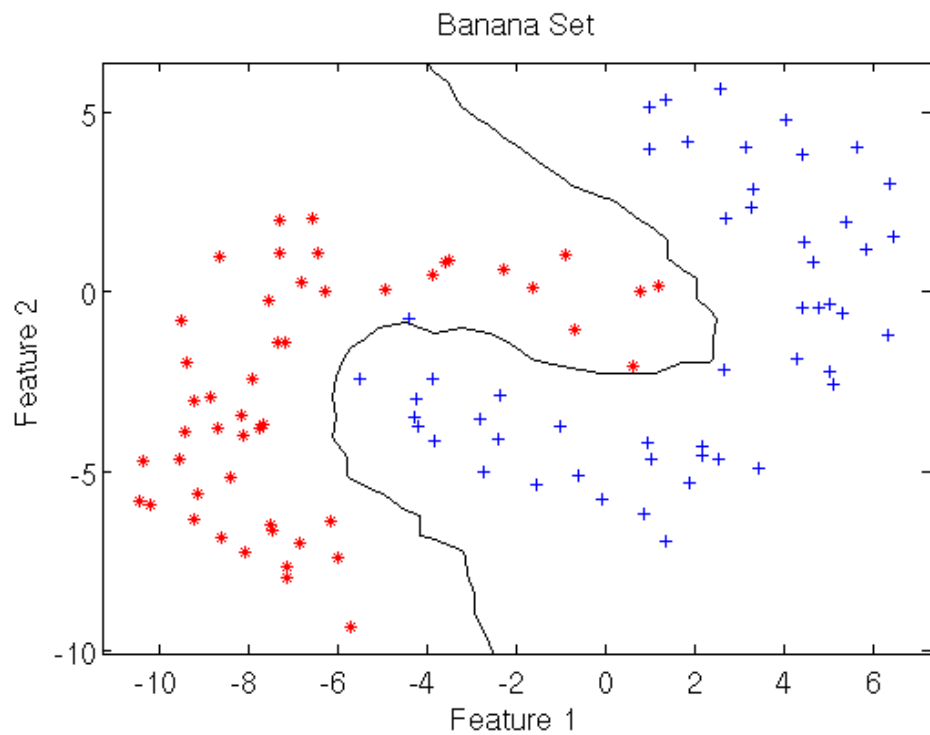
W k-NN classifier

K Number of the nearest neighbors used

E The leave-one-out error of the KNNC

Esempio

```
ds = gendatb(100);  
W = knnc(ds, 5);
```



Per il classificatore KNN esiste una funzione specifica di stima dell'errore

help testk

TESTK Error estimation of the K-NN rule

$E = \text{TESTK}(A, K, T)$

INPUT

A Training dataset
K Number of nearest neighbors (
T Test dataset
(default leave-one-out)

OUTPUT

E Estimated error of the K-NN rule

DESCRIPTION

Tests a dataset T on the training dataset A using the K-NN rule and returns the classification error E. In case no set T is provided, the leave-one-out error estimate on A is returned.

The advantage of using TESTK over TESTC is that it enables leave-one-out error estimation.

Stima dei parametri del K-NN

La stima dei parametri del K-NN può essere eseguita mediante l'uso di un validation set (come per gli altri classificatori); oppure mediante il metodo del “*leave-one-out*”.

Il *leave-one-out* può essere visto come una sorta di “numPattern-CrossValidation”, in cui ciascun pattern del training set viene classificato utilizzando tutti gli altri pattern dello stesso data set.

Nota:

Il *leave-one-out* (per la stima di k) è implementato nella funzione KNNC dei PRTools

Esercizio:

Stima del valore ottimale di K tramite validation set

Scelto un dataset, dividetelo in un set di design e un set di test.

Dal set di design estraete un training set ed un validation set.

Costruite un set di classificatori k -nn con vari valori di k .

Valutate l'errore sul train, test, validation.

- quale sarebbe stato l'errore sul test se avessimo scelto il k migliore basandoci sui risultati di errore sul training set?
- quale sarebbe stato l'errore sul test se avessimo scelto il k migliore basandoci sui risultati sul VALIDATION set?
- tabellate i dati e fate un grafico dell'errore al variare di k
- **Visualizzare la superficie di decisione ottenuta con il valore di K ottimale e col K peggiore**

```
%Calcolo andamento delle prestazioni del classificatore KNN  
al variare del parametro K (TRACCIA)
```

```
k_max = 20  
ds_full = gendatb(200);  
  
[ds_trn_full, ds_tst] = gendat(ds_full, 0.5);
```

```
[ds_trn, ds_val]= gendat(ds_trn_full,0.60);

for k=1:k_max,
    train_err(k) = testk(ds_trn, k);
    val_err(k) = testk(ds_trn, k, ds_val);
    tst_err(k) = testk(ds_trn, k, ds_tst);
end

%Calcolo valore ottimale di K
[tmp k_best] = min(val_err)

%Visulizzo andamento errore
figure(1);
plot(val_err, 'k');
hold on;
plot(tst_err, 'r');
plot(train_err, 'b');

legend('validation', 'test', 'train')
hold off;
```

%Creo il classificatore e visualizzo la sup. di decisione

```
W = knnc(ds_trn, k_best);
```

```
figure(2);
```

```
scatterd(ds_tst);
```

```
hold on;
```

```
plotc(W);
```

```
hold off;
```

```
figure(3);
```

```
scatterd(ds_trn);
```

```
hold on;
```

```
plotc(W);
```

```
hold off;
```

Esercizio:

Classificate un dataset (spazio delle features a 2 dimensioni) utilizzando un classificatore k -nn ed un classificatore lineare. Calcolate l'errore e visualizzate le superfici di decisione

TRACCIA

- **Scegliere il dataset**
- **Dividere dataset in train e test**
- **Visualizzare**
- **Creare classificatore Lineare e KNN (quali parametri?)**
- **Classificare training e test (label e prob a posteriori)**
- **Calcolo errore**
- **Visualizzazione superficie decisione di entrambi i classificatori, sovrappone-
dola al training e al test set**

Esempio di svolgimento

```
A=gendatl([500,200],1.5);  
[ds_train, ds_test]=gendat(A,0.2);  
figure(1)  
scatterd(ds_train)  
title('TRAINING SET')  
figure(2)  
scatterd(ds_test)  
title('TEST SET')  
  
wlin=ldc(ds_train,0.2);  
[wknn,k, errTrain_knn]=knnc(ds_train);  
%errore train classificatore lineare  
errTrain_lin=testc(ds_train*wlin)  
  
%classifico il test  
ris_lin= ds_test*wlin;  
probPost_lin=+classc(ris_lin)  
labelAssegnate_lin= labeld(ris_lin)
```

```

ris_knn= ds_test*wknn;
probPost_knn=+classc(ris_knn)
labelAssegnate_knn= labeld(ris_knn)

%calcolo errore sul test
errTest_lin=testc(ris_lin)
% errTest_knn= testc(ris_knn);
errTest_knn=testk(ds_train,k,ds_test)

errore=[errTrain_lin,errTest_lin ; ...
        errTrain_knn, errTest_knn]*100;

% visualizzo superficie decisione

figure(1)
hold on
plotc(wlin); plotc(wknn);

figure(2)
hold on
plotc(wlin); plotc(wknn);

```