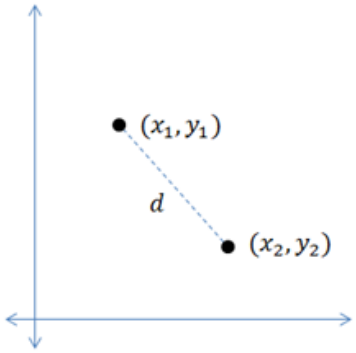
Clasificarea datelor. K-Nearest Neighbors

Clasificarea datelor constituite dintr-o mulțime de elemente de un anumit tip constă în stabilirea categoriei în care se încadrează elementele pe baza unor trăsături, caractersistici sau atribute ale acestora. Categoriile în care trebuie încadrate elementele se numesc *clase* și trebuie să fie în număr finit.
Algoritmii care realizează clasificarea se numesc *clasificatori* – scopul lor este de a stabili o clasă (dintr-o mulțime finită de clase) pentru oricare element dintr-o mulțime sau un domeniu, realizând diverse prelucrări pe baza trăsăturilor acelor elemente.
Clasificarea este un proces de învățare supervizată, ceea ce înseamnă că un clasificator necesită o mulțime finită de elemente deja clasificate (*mulțimea datelor de antrenare*), pe baza cărora învață să clasifice și alte elemente de același tip. Exemplu: Un clasificator care realizează recunoașterea unor persoane din imagini necesită o mulțime de imagini pre-clasificate, unde persoanele să fie deja cunoscute. Pe baza acestor imagini, clasificatorul va învăța să recunoască aceleași persoane în (teoretic) orice altă imagine de același tip.

K-Nearest Neighbors (KNN) este o medodă de clasificare în cadrul căreia, pentru a se stabili clasa de apartenență a unui element, se iau în calcul cei mai apropiați k vecini cunoscuți ai acelui element. Datele de antrenare constau într-o mulțime de elemente deja clasificate. La apariția unui nou element, clasa acestuia se decide funcție de clasele celor mai apropiați k vecini cunoscuți (în cel mai simplu caz, noul element este încadrat în clasa majoritară prezentă printre cei k vencini ai săi). Pentru a se stabili cât de apropiat este un vecin, este nevoie de o metrică de similaritate prin intermediul căreia să se detemine “distanța” dintre oricare două elemente, similar cu abordarea de la laboratorul anterior.

Exemplu:

Dorim să clasificăm o serie de elemente care aparțin mulțimii punctelor din plan. Pentru aceasta, stabilim de la bun început următoarele aspecte:
- clasele în care vom încadra punctele: în cazul de față presupunem două clase, denumite roșu și verde.
- trăsăturile elementelor (caracteristicile elementelor pe baza cărora se va realiza clasificarea): în cazul punctelor din plan, aceste caracteristici vor fi coordonatele x, y
- metrica de similaritate: o modalitate de determinare a distanței dintre două puncte, pe baza trăsăturilor menționate anterior. În cazul de față, vom folosi distanța euclidiană (Fig. 1)



$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Fig 1. Distanța euclidiană dintre două puncte din plan

- o mulțime de date de antrenare: o serie de puncte care sunt deja încadrate într-una din cele două clase. Pe baza acestor puncte, algoritmul de clasificare va putea generaliza – va putea stabili clasa în care se încadreaza oricare punct din plan

Principiul de bază al **KNN** constă în faptul că apartenența unui punct la o clasă se determină pornind de la cei mai apropiați *k* vecini din mulțimea de antrenare.
Fie mulțimea de puncte din Fig. 2. Acestea sunt deja încadrate în clasele roșu sau verde, scopul fiind să determinăm clasa oricărui alt punct.

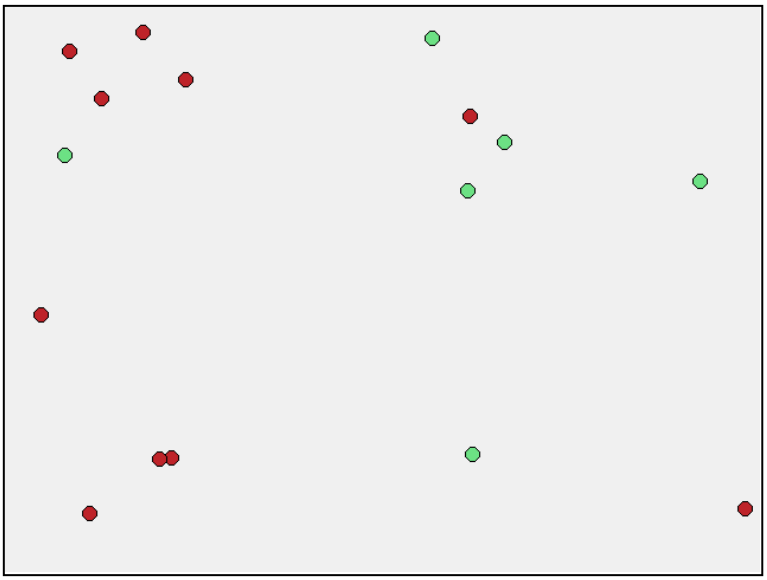
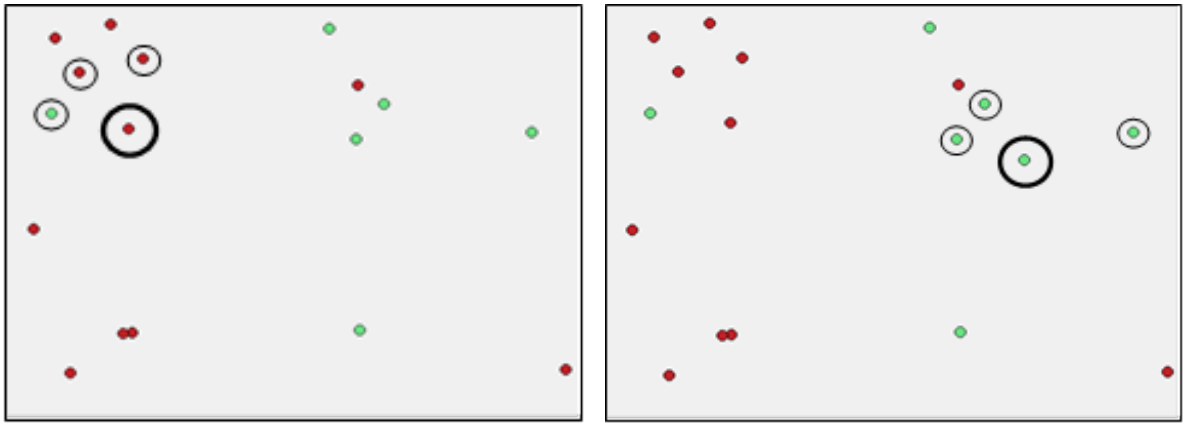


Fig. 2. Un set de puncte împărțite în două categorii. Clasele aferente sunt marcate prin culori diferite.

Stabilim o valoare a paramerului k = 3. Pentru oricare nou punct, clasa în care se încadrează se decide funcție de clasele celor mai apropiate trei puncte dintre cele deja cunoscute. Spre exemplu, dacă dintre cele mai apropiate trei puncte, două aparțin clasei roșu iar al treilea aparține clase verde, putem decide că noul punct aparține clasei roșu, deoarece aceasta este clasa majorității vecinilor săi (Fig. 3(a)). Un alt exemplu este ilustrat în Fig. 3(b), unde noul punct aparține clasei verde, cea a tuturor vecinilor săi.



(a) **(b)**

Fig. 3. Două exemple de clasificare a unor noi puncte. Acestea vor aparține clasei majoritare a celor mai apropiați k=3 vecini

K-nearest neighbor cu ponderi

Până acum, toți vecinii noului punct au contribuit în mod echitabil la stabilirea clasei acestuia, ei contând doar ca număr. Se pot însă defini ponderi pentru fiecare vecin, acestea semnificând importanța vecinilor în procesul de stabilire a clasei noului punct.

Cel mai frecvent se consideră ca pondere inversul pătratului distanței vecinului față de punct. Așadar, cu cât un vecin este mai apropiat de punct, cu atât crește probabilitatea ca punctul să aparțină clasei acelui vecin.

Fața de exemplele anterioare, vecinii nu conteaza doar ca număr, ci decizia se ia funcție de suma ponderilor vecinilor din fiecare clasă. Noul punct va fi asignat clasei cu suma cea mai mare.

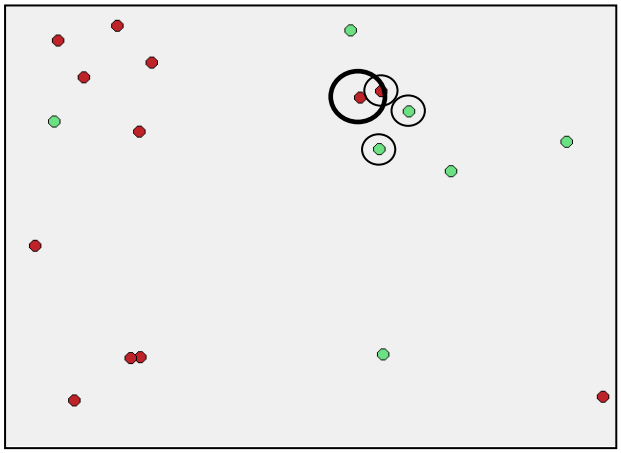


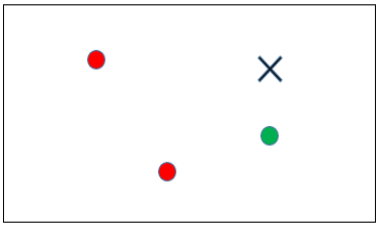
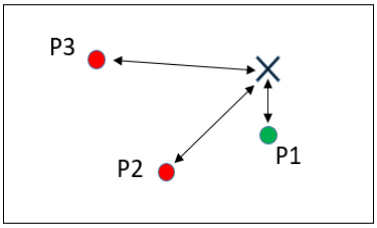
Fig. 4. Punct clasificat ținînd cont de ponderile vecinilor. Vecinul roșu este mult mai apropiat de noul punct decât ceilalți doi vecini, așadar ponderea sa (inversul patratului distanței) este mai mare decât suma ponderilor celorlalți doi vecini.

În Fig. 4, pentru *k* = 3, se observă că, deși vecinii majoritari ai punctului nou adăugat aparțin clasei verde, punctul vizat se va încadra în clasa corespunzătoare vecinului roșu. Acesta, fiind mult mai apropiat decât ceilalți doi, are o pondere mult mai mare. Principul este explicat în detaliu în Tabelul 1.

Tabelul 1. Clasificarea folosind KNN cu și fără ponderi

KNN fără ponderi, K=3	**KNN cu ponderi, K=3**

- clasa predominantă se determină contorizând vecinii
- dintre cei mai apropiați K=3 vecini:
  - doi aparțin clasei roșu
  - unul aparține clasei verde
- astfel, un nou punct X va fi asignat clasei roșu
	- clasa predominantă se determină efectuând suma ponderilor vecinilor:
	- $W_{red} = W_2 + W_3 = 1 / dist(X, P_2)^2 + 1 / dist(X, P_3)^2$
	- $W_{green} = W_1 = 1 / dist(X, P_1)^2$
	- dacă $W_{green} > W_{red}$ atunci lui X I se atribuie clasa verde

Testarea clasificatorului

Testarea presupune verificarea acurateței clasificatorului antrenat. Pentru aceasta, se stabilește o mulțime de puncte de test, a căror clasă se cunoaște deja, dar care nu sunt implicate în procesul de antrenare. Se aplică algoritmul de clasificare pe aceste puncte și se determină, procentual, câte puncte sunt clasificate greșit de către algoritmul de clasificare (rezultă *eroarea de clasificare* a clasificatorului)

Exemplu: presupunem ca mulțimea inițială este formată din 100 puncte. Dintre acestea:
- 60 de puncte se vor folosi pentru antrenare (vor constitui mulțimea de antrenare)
- 40 de puncte se vor folosi pentru testare (vor constitui mulțimea de test)

Aplicăm KNN pe cele 40 puncte de test (individual) folosind vecini preluați dintre punctele din mulțimea de antrenare. Interesează câte dintre cele 40 puncte de test sunt clasificate greșit (pentru câte dintre punctele de test clasa determinată de KNN va fi alta decât cea inițială a punctului). Atunci eroarea de clasificare va fi:

$$err = \frac{nr\_puncte\_clasificate\_gresit}{nr\_total\_de\_puncte\_test}$$

Validarea clasificatorului

Validarea se referă la ajustarea parametrilor clasificatorului astfel încât să se minimizeze eroarea de clasificare. În cazul de față, singurul parametru al KNN este K, numărul de vecini. Scopul este de a deterimina cea mai buna valoare a lui K, cea pentru care eroarea de clasificare este minimă. În general, pentru validare se creează o mulțime separată a punctelor, mulțimea de validare.

Cerințe

1) Implementați algoritmul K-nearest neigbor, variantele fără ponderi, respectiv cu ponderi. Folosind datele furnizate pe lângă documentație, aplicați KNN pentru câteva puncte (altele decat cele din datele inițiale). Pentru fiecare punct, se va determina clasa de apartenență funcție de cei mai apropiați *k* vecini și se va afișa punctul de culoarea corespunzătoare clasei. Ca punct de plecare, se poate utiliza codul pus la dispoziție pe lângă documentație.

Pentru un nou punct, neclasificat:

- Se determină distanțele față de toate punctele existente
- Se sortează ascendent punctele funcție de distanțe
- Se consideră primele *k* puncte dintre cele sortate (acestea vor fi cei mai apropiați *k* vecini)
- Se determină clasa punctului adăugat
    - Dacă nu se iau în calcul ponderile, se determină histograma vecinilor pe clase (adică se determină numărul vecinilor care aparțin fiecărei clase), iar clasa corespunzătoare punctului adăugat va fi cea cu numărul maxim de vecini
    - Dacă se iau în calcul ponderile, acestea de determină pentru fiecare vecin ca fiind $1/d^2$, unde d este distanța de la vecin până la punctul adăugat. Se sumează aceste ponderi pe clase, iar clasa punctului adăugat se consideră a fi cea cu suma maximă

2) Determinați eroarea de clasificare a KNN pentru datele inițiale, nemodificate (nu se iau în calcul alte puncte adăugate ulterior), pentru o anumită valoare a lui K (poate fi K=3)
Pentru aceasta, împărțiți datele disponibile astfel: 60% pentru antrenare și 40% pentru testare.
Aplicați KNN pe punctele din mulțimea de test preluând vecinii din mulțimea de antrenare.
Determinați eroarea de clasificare pentru ambele abordări ale algoritmului (cu/fără ponderi) ca fiind:

$$err = \frac{nr\_puncte\_clasificate\_gresit}{nr\_total\_de\_puncte\_test}$$

3) Realizați o validare a algoritmului pentru identificarea celei mai bune valori a lui K.
Împărțiți mulțimea punctelor în tre submulțimi astfel:

- 60% dintre puncte pentru antrenare
- 20% pentru validare
- 20% pentru test

Aplicați următoarele două metode:

a) *Validare simplă*:
Pentru fiecare valoare a lui k, de la 1 la o valoare maximă k_{max}, se determină eroarea de clasificare pentru punctele din mulțimea de validare folosind vecinii preluați din mulțimea de antrenare. Cea mai bună valoare a lui k este cea pentru care eroarea este minimă. Testați algoritmul cu această valoare a lui k folosind mulțimea de test.

b) Validare încrucișată (Cross validation):
Se împarte mulțimea datelor într-o mulțime de antrenare si una de test.
Pentru cea de antrenare:
Se formează n subgrupuri egale folosind punctele din setul de date. Așadar, pentru fiecare i =1..n, datele se vor împărți astfel:

- un subgrup i de test
- mulțimea formată din restul de n-1 subgrupuri, cea care se va folosi efectiv pentru antrenare
  O astfel de partiționare se numește *split*, iar grupul i se mai numește *fold*.
  Modalitatea de divizare a mulțimii punctelor se ilustrează în Fig. 5.

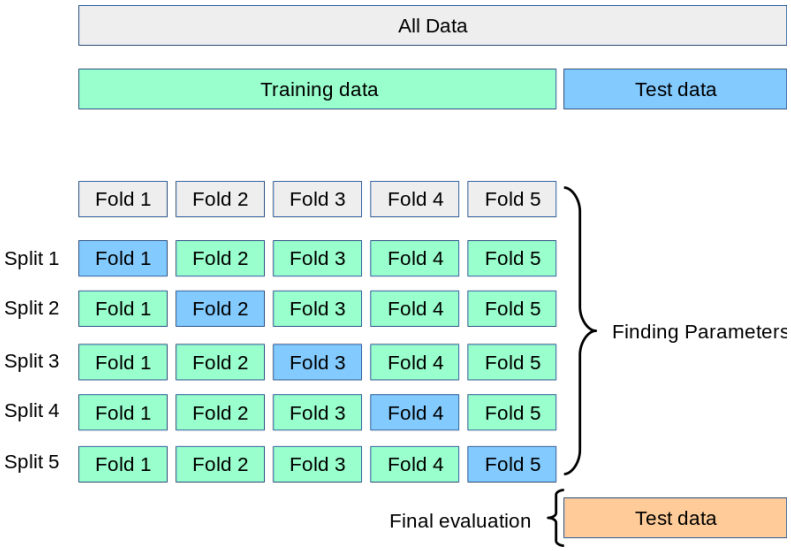


Fig. 5. Modalitatea de partiționare a mulțimii datelor pentru realizarea validării încrucișate și a testării.

- pentru fiecare subgrup i, i = 1..n se obține câte o eroare pentru fiecare valoare a lui k, k=1..k_{max} .
- astfel, pentru fiecare valoare a lui k vom obține n erori de clasificare, care rezultă din validarea efectuată pentru fiecare din cele n subgrupuri.
- pentru fiecare valoare a lui k se determină eroarea medie.

Valoarea finală a lui k este cea cu eroarea medie minimă. În final, se testează clasificatorul cu această valoare a lui k folosind mulțimea de test.