Analiza și procesarea datelor prin tehnici de Învățare Automată

3. Tehnici de învățare supervizată 1: Clasificare

Șef Lucrări Dr. Ing. Horia Modran

Contact: horia.modran@unitbv.ro / modranhoria@gmail.com

Tel: 0770171577

Universitatea Transilvania din Brașov

FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

2024 - 2025

Clasificare și Regresie

- Învățarea unei funcții discrete: Clasificare
  - Clasificare binară:
    - fiecare exemplu este clasificat ca adevărat (pozitiv) sau fals (negativ)
  - poate, de asemenea, prezice mai multe clase (3, 4, 5…)
- Învățarea unei funcții continue: Regresie
  - Ex.: Linear Regression, Logistic regression

Clasificare

- Se dă o mulțime de antrenare: o mulțime de instanțe (vectori de antrenare, obiecte) - datele de antrenare
- Instanțele au atribute
- Fiecare instanță are atribute cu anumite valori
- Ultimul atribut este clasa (variabila țintă)

Starea vremii	Temperatură	Umiditate	Vânt	Joc
Soare	Mare	Mare	Absent	Nu
Soare	Mare	Mare	Prezent	Nu
Înnorat	Mare	Mare	Absent	Da
Ploaie	Medie	Mare	Absent	Da
Ploaie	Mică	Normală	Absent	Da
Ploaie	Mică	Normală	Prezent	Nu
Înnorat	Mică	Normală	Prezent	Da
Soare	Medie	Mare	Absent	Nu
Soare	Mică	Normală	Absent	Da
Ploaie	Medie	Normală	Absent	Da
Soare	Medie	Normală	Prezent	Da
Înnorat	Medie	Mare	Prezent	Da
Înnorat	Mare	Normală	Absent	Da
Ploaie	Medie	Mare	Prezent	Nu

Clasificare

- Construcția modelului: set de clase predeterminate
  - se presupune că fiecare tuplu/eșantion aparține unei clase predefinite, așa cum este determinat de eticheta clasei
  - pentru construirea modelului: set de antrenament
  - modelul este reprezentat ca reguli de clasificare, arbori de decizie sau formule matematice
- Utilizare model: pentru clasificarea obiectelor necunoscute
  - estimarea preciziei modelului
  - dacă acuratețea este acceptabilă, utilizați modelul pentru a clasifica date ale căror etichete de clasă nu sunt cunoscute

Tipuri de date

- Există patru tipuri de atribute, organizate pe două coordonate:
- Atribute categoriale (calitative):
  - de tip nominal (ex. culoarea ochilor, nume, sex, CNP)
  - Ordinal (înălțime (mică, medie, mare), ranguri, calificative)
- Atribute numerice (cantitative):
  - de tip interval (Temperatura în °C, date calendaristice)
  - rațional (lungime, distanță, prețuri)

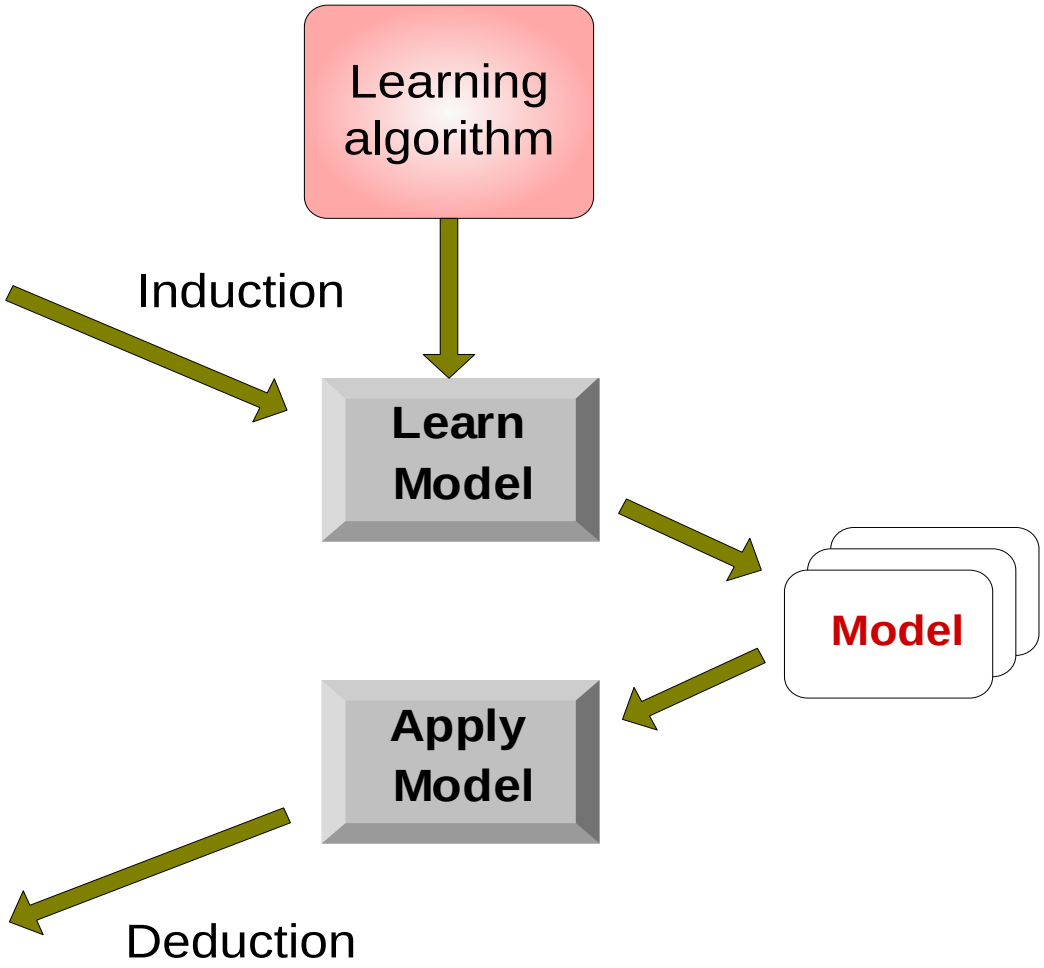
Exemplu de clasificare

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Prepararea datelor

- Curățarea datelor
  - preprocesează datele pentru a reduce zgomotul și a gestiona valorile lipsă
- Analiza relevanței (selectarea caracteristicilor)
  - eliminați atributele irelevante sau redundante
- Transformarea datelor
  - generalizați datele la concepte superioare/discretizare
  - normalizați valorile atributelor

Condiții pentru învățare

- Condiții pentru o învățare “bună”
  - Clasificatoarele trebuie să fie suficient de "expresive" pentru a fi în concordanță cu setul de învățare -> altfel problemă de " underfit" (*underfitting*)
  - Clasificatoarele care au o complexitate prea mare pot duce la fenomenul de "overfit" (*overfitting*) = include zgomot sau șabloane de date nerelevante

Tehnici de clasificare

- Metode bazate pe arbori de decizie
  - Random Forests
- K-Nearest Neighbors
- Support Vector Machines
- Logistic Regression
- Rețele Bayesiene
- și multe altele...

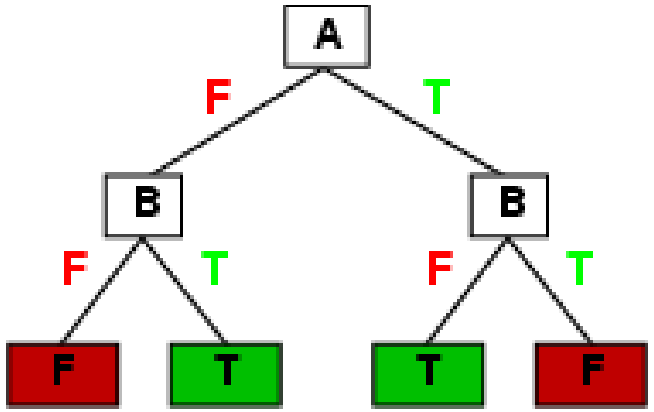
Arbori de decizie

- Principiu – algoritm Greedy
  - arborele este construit într-o manieră recursivă de sus în jos folosind tehnica *Divide et Impera*
- Iterații - la început, toate tuplurile sunt la rădăcină
  - tuplurile sunt partiționate recursiv
  - atributele testului sunt selectate pe baza unei măsuri euristice sau statistice (de exemplu – *Information Gain*)
- Condiții de oprire
  - toate mostrele pentru un nod dat aparțin aceleiași clase
  - nu există atribute rămase pentru partiționarea ulterioară
  - nu mai sunt eșantioane

Expresivitate

- Arborele de decizie poate exprima orice funcție a atributelor de intrare
  - de exemplu, pentru funcțiile booleene, tabelului de adevăr pt **A xor B**:

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



- există un arbore de decizie consistent pentru orice set de antrenament cu o cale către frunze pentru fiecare exemplu (cu excepția cazului în care f nedeterminist în x), dar probabil că nu se va generaliza la exemple noi

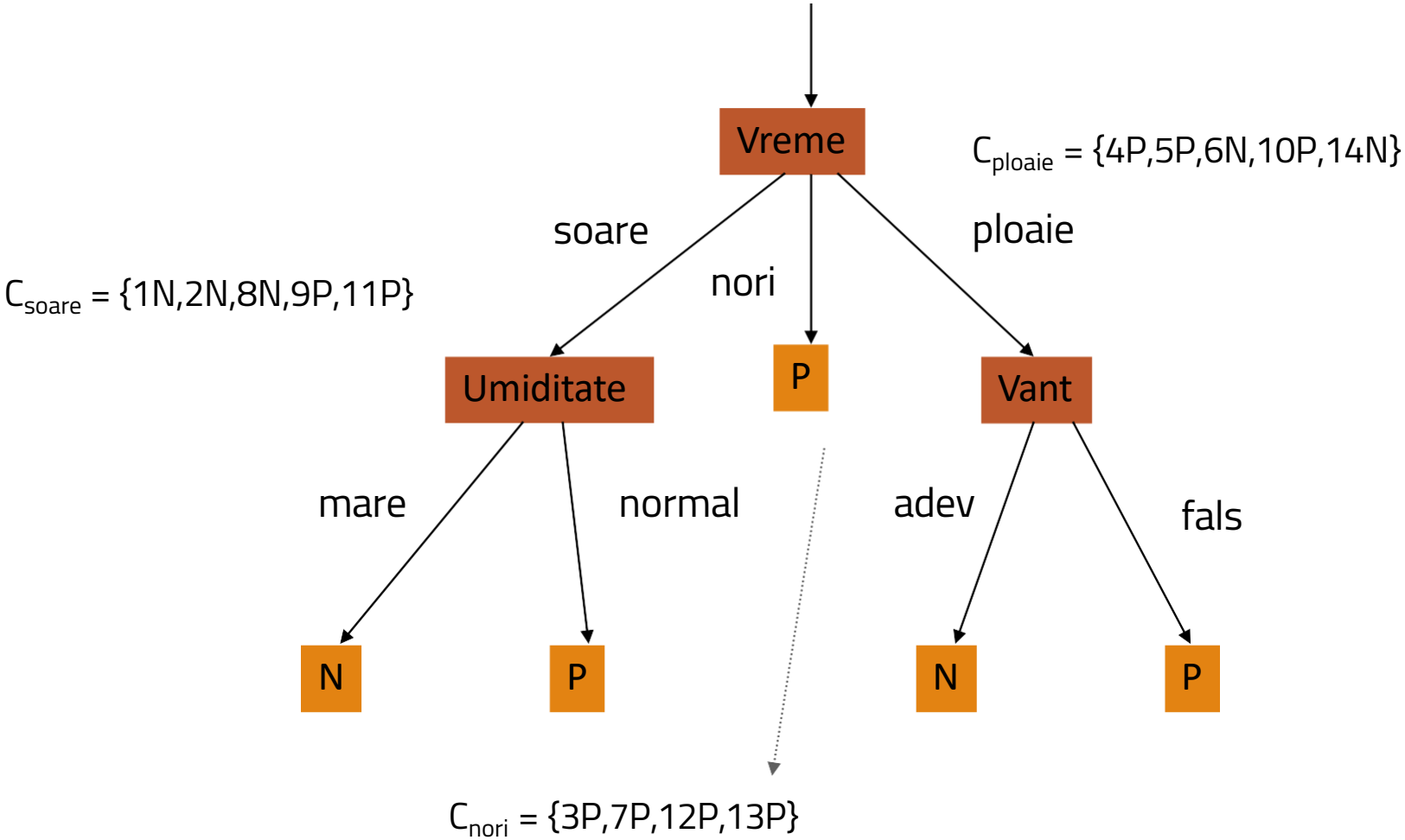
Partiționare optimă

- **Prima decizie:** Atributul rădăcină reprezintă primul test efectuat asupra datelor
- **Impact asupra purității nodurilor**: O alegere bună a atributului rădăcină poate duce la noduri mai pure (noduri care conțin în principal instanțe din aceeași clasă)
- Criterii pentru a alege cel mai bun atribut pentru rădăcină, dintre care cele mai comune sunt:
  - Entropia: Măsoară impuritatea unui set de date
  - Gain (Câștigul informațional)

Exemplul 1 – Clasificare Binară

No.	Atribute				Clasa
	Vreme	Temperatura	Umiditate	Vant	
1	soare	cald	mare	fals	N
2	soare	cald	mare	adev	N
3	nori	cald	mare	fals	P
4	ploaie	placut	mare	fals	P
5	ploaie	racoare	normal	fals	P
6	ploaie	racoare	normal	adev	N
7	nori	racoare	normal	adev	P
8	soare	placut	mare	fals	N
9	soare	racoare	normal	fals	P
10	ploaie	placut	normal	fals	P
11	soare	placut	normal	adev	P
12	nori	placut	mare	adev	P
13	nori	cald	normal	fals	P
14	ploaie	placut	mare	adev	N

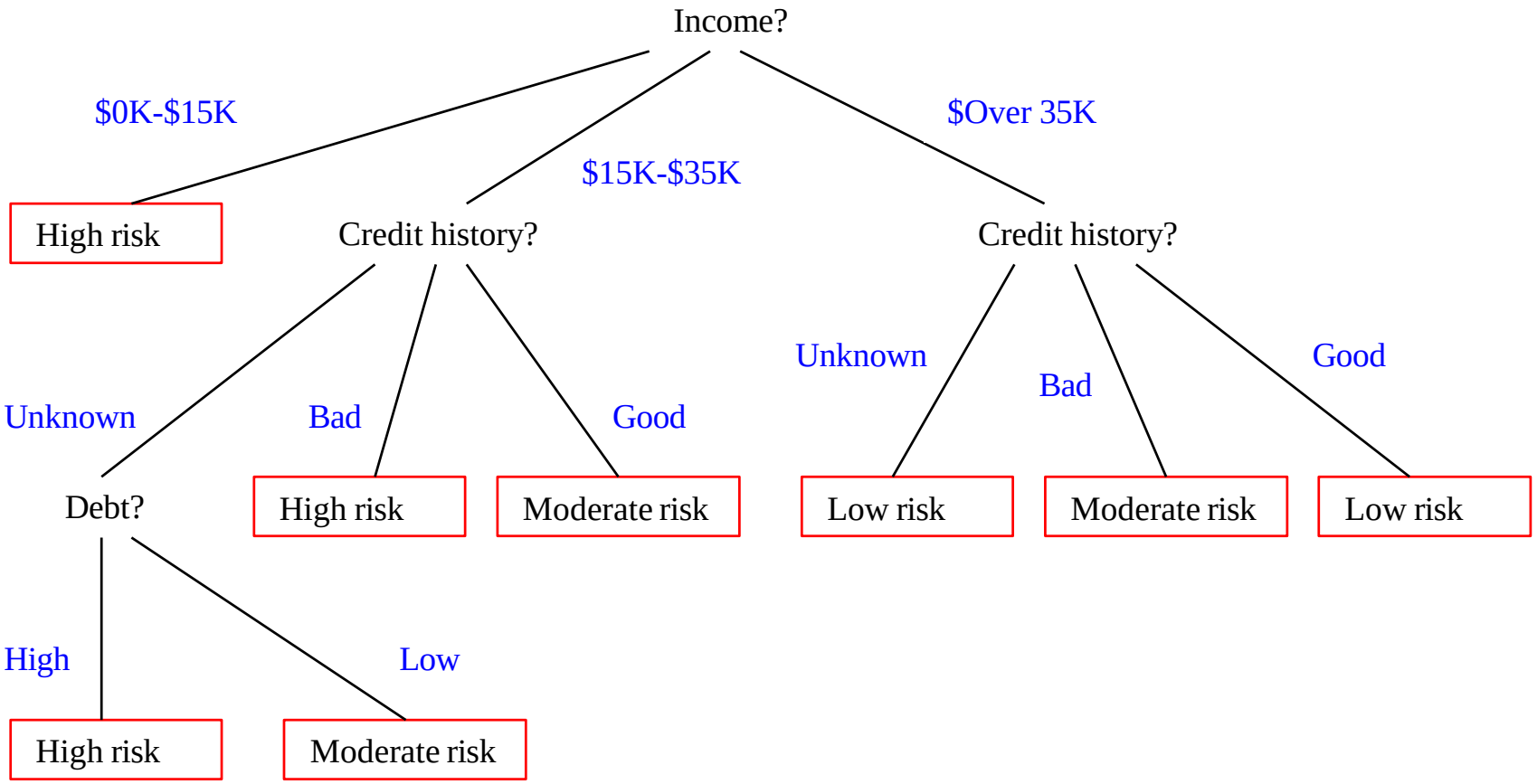
Exemplul 1 – Clasificare Binară



Exemplul 2 – Multi-clasă

No.	*Risk (Classification)*	*Credit History*	*Debt*	*Collateral*	**Income**
1	High	Bad	High	None	$0 to $15k
2	High	Unknown	High	None	$15 to $35k
3	Moderate	Unknown	Low	None	$15 to $35k
4	High	Unknown	Low	None	$0k to $15k
5	Low	Unknown	Low	None	Over $35k
6	Low	Unknown	Low	Adequate	Over $35k
7	High	Bad	Low	None	$0 to $15k
8	Moderate	Bad	Low	Adequate	Over $35k
9	Low	Good	Low	None	Over $35k
10	Low	Good	High	Adequate	Over $35k
11	High	Good	High	None	$0 to $15k
12	Moderate	Good	High	None	$15 to $35k
13	Low	Good	High	None	Over $35k
14	High	Bad	High	None	$15 to $35k

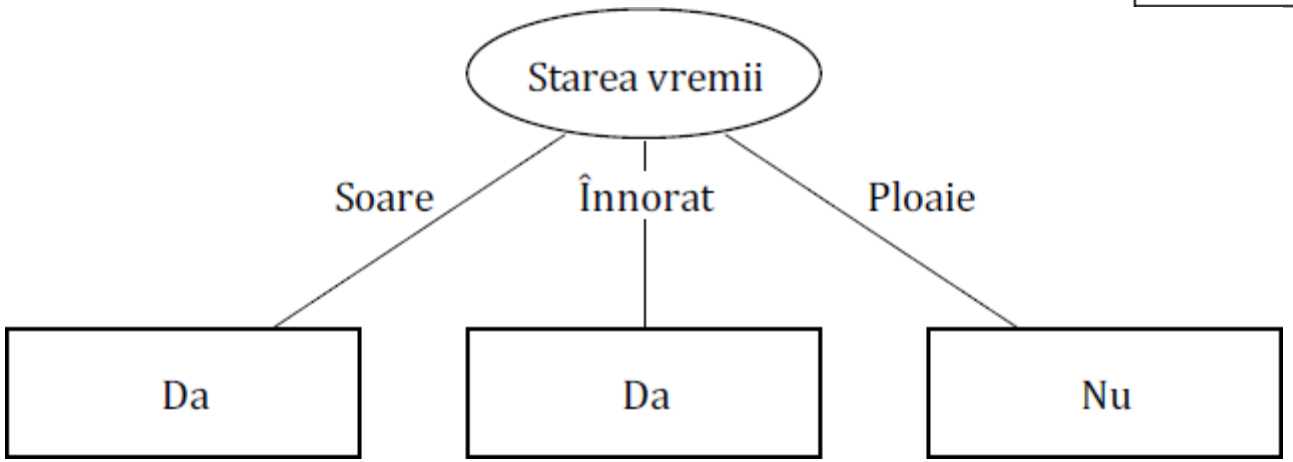
Exemplul 2 – Multi-clasă



Atribute nominale

- Partiționarea multiplă
  - Numărul de partiții = numărul de valori distincte
- Partiționarea binară
  - Se împart valorile în două submulțimi
- Trebuie descoperită partiționarea optimă

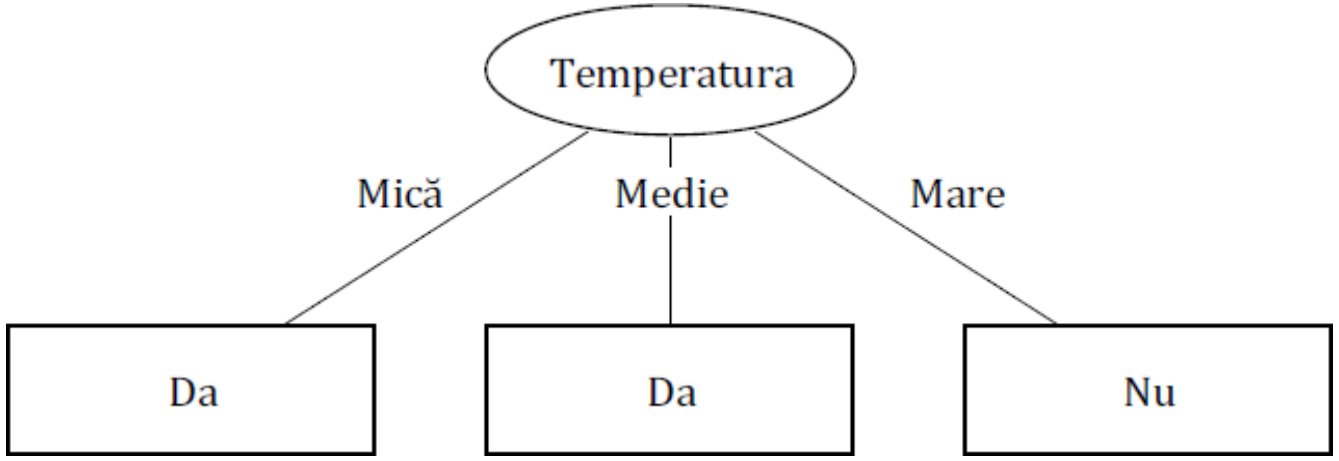
Starea vremii	Joc
Soare	Da
Înnorat	Da
Ploaie	Nu



Atribute ordinale

- Partiționarea multiplă
  - Numărul de partiții = numărul de valori distincte
- Partiționarea binară
  - Se împart valorile în două submulțimi
- Trebuie descoperită partiționarea optimă

Temperatură	Joc
Mică	Da
Medie	Da
Mare	Nu



Atribute continue

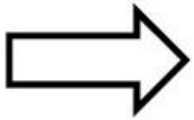
- Se discretizează datele pentru a le transforma în atribute ordinale:
  - Cu interval egal (histograma)
  - Cu frecvență egală (mulțimi cu numere egale de instanțe)
  - Grupare (clustering)
- Decizie binară: (A_i < v) sau (A_i > v)
  - Trebuie considerate toate partiționările posibile
  - Necesită un efort de calcul mai mare

Discretizarea

- Cu intervale egale – de exemplu, 3 intervale

[65, 75], (75, 85], (85, 95]

Umiditate	Joc
65	Da
70	Da
72	Da
75	Da
80	Da
85	Da
86	Nu
90	Nu
90	Nu
91	Nu
93	Nu
95	Nu

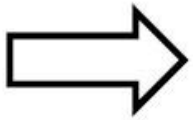


Umiditate-Dis1	Joc
Mică	Da
Mică	Da
Mică	Da
Mică	Da
Medie	Da
Medie	Da
Mare	Nu
Mare	Nu
Mare	Nu
Mare	Nu
Mare	Nu
Mare	Nu

Discretizarea

- Binară – de exemplu, 85

Umiditate	Joc
65	Da
70	Da
72	Da
75	Da
80	Da
85	Da
86	Nu
90	Nu
90	Nu
91	Nu
93	Nu
95	Nu



$(A_i \leq 85)?$

Umiditate	Umiditate-Bin	Joc
65	Da	Da
70	Da	Da
72	Da	Da
75	Da	Da
80	Da	Da
85	Da	Da
86	Nu	Nu
90	Nu	Nu
90	Nu	Nu
91	Nu	Nu
93	Nu	Nu
95	Nu	Nu

Avantaje/Dezavantje

- Avantaje
  - ușor de construit/implementat
  - extrem de rapid la clasificarea înregistrărilor necunoscute
  - acuratețea este comparabilă cu alte tehnici de clasificare pentru multe seturi de date simple
- Dezavantaje
  - costisitor din punct de vedere computațional de antrenat
  - unii arbori de decizie pot fi excesiv de complexi și nu generalizează bine datele
  - mai puțină expresivitate

Clasificare bazată pe instanțe

- *Lazy* learners

Instanțele memorate

Atr1	……	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

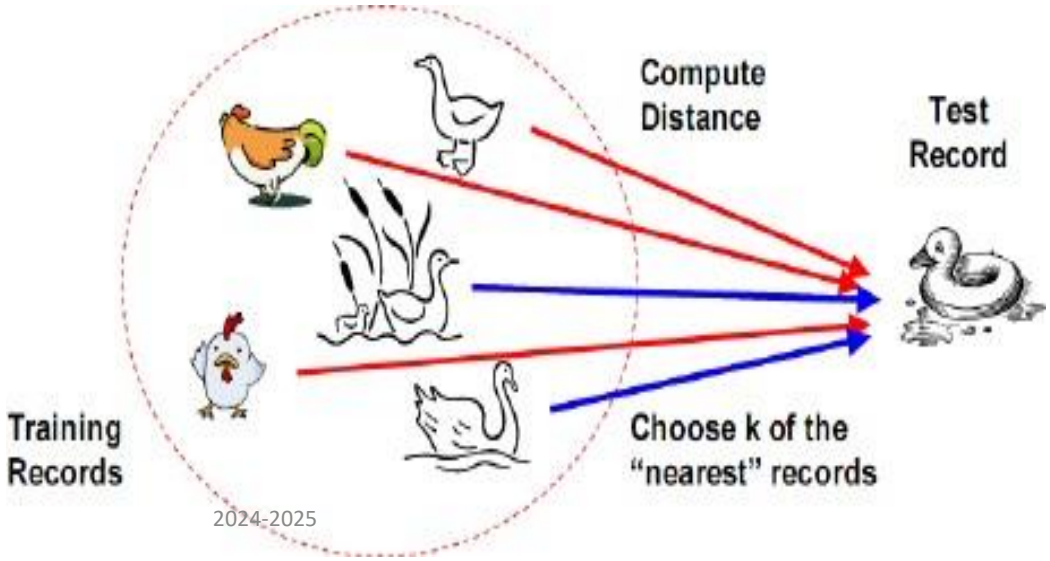
- Se memorează efectiv instanțele de antrenare și se folosesc pentru a prezice clasele instanțelor noi

Instanță nouă

Atr1	……	AtrN

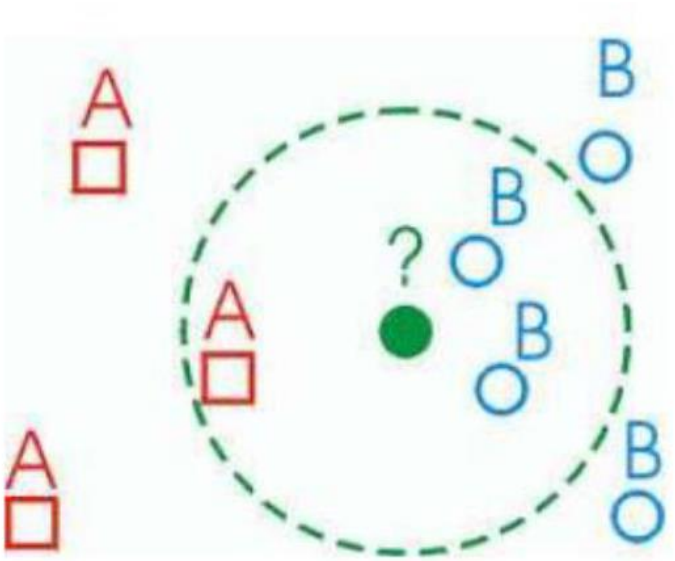
K-Nearest Neighbor (KNN)

- simplu, dar un algoritm de clasificare foarte puternic
- clasifică pe baza unei măsuri de similitudine
- neparametric
- Învățare "leneşă" (*lazy learning*)
  - Se folosesc cele mai apropiate k instanțe pentru a realiza clasificarea



KNN: Clasificare

- clasificat prin „MAJORITY VOTES” pentru clasele vecine
  - atribuit celei mai comune clase dintre cei K vecini cei mai apropiați ai săi (prin măsurarea „distanței” între date)



Pașii KNN

Step 1: Determine parameter K = number of nearest neighbors

Step 2: Calculate the distance between the query-instance and all the training examples.

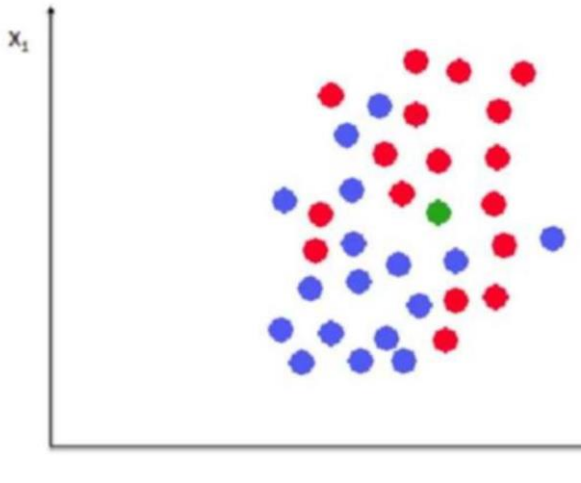
Step 3: Sort the distance and determine nearest neighbors based on the k-th minimum distance.

Step 4:Gather the category Y of the nearest neighbors.

Step 5: Use simple majority of the category of nearest neighbors as the prediction value of the query instance.

Distanță euclidiană:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$



Scalarea

- Problemă legată de ordinul de mărime al datelor:
  - avem două atribute: înălțimea şi greutatea unor persoane
  - Înălțimea e măsurată în metri; intervalul poate fi de ex [1.50 m, 2.00 m], deci cu o diferență de maxim 0.5
  - greutatea se măsoară în kilograme - interval [50 kg, 130 kg]
  - Diferențele de greutate domină pe cele în înălțime; o diferență de 1 kg este mai mare decât orice diferență de înălțime, **influențând prea mult la calculul distanței**
  - Soluție: scalarea mărimilor

Scalarea

- Se recomandă scalarea atributelor pentru a preveni dominarea măsurii de distanță de către un anumit atribut
- De exemplu:
  - Înălțimea unei persoane – interval [1.5, 2.0] m
  - Greutatea unei persoane – interval [50, 130] kg
  - Venitul unei persoane – interval [20.000, 1.000.000] lei/an
- Valorile atributelor sunt normalizate:

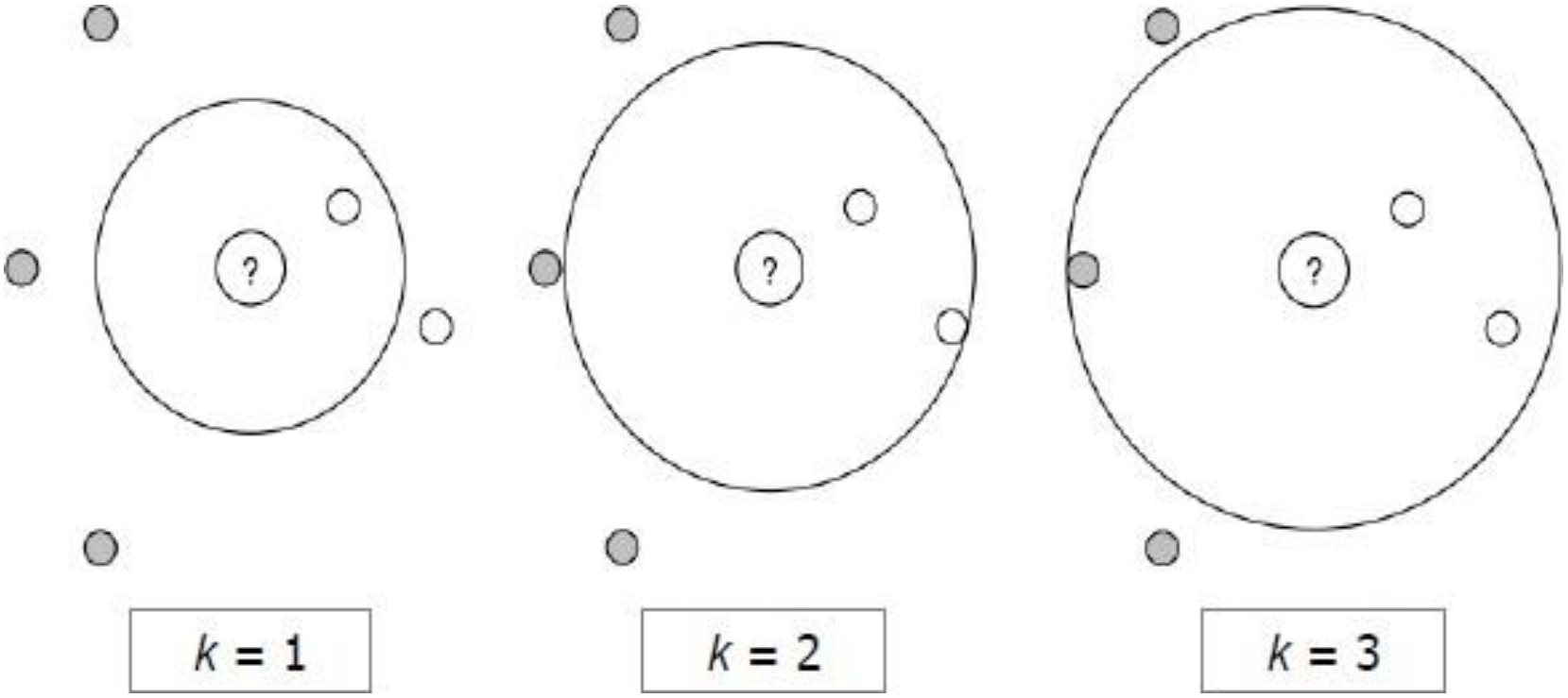
$$x_i' = \frac{x_i - min_i}{max_i - min_i} \in [0, 1]$$

Atribute nominale

- Este necesară gasirea unei “distanțe” între valorile diferite ale atributelor nominale
  - Ex: distanța dintre valorile: roşu, galben şi verde
- De obicei, se consideră distan a zero pentru valori identice și unu în caz contrar
  - Ex. având mai multe culori se poate utiliza o măsură metrică a nuanțelor din spațiul culorilor, punând galbenul mai apropiat de portocaliu decât verde
- Unele atribute au o importanță diferită care ese reflectată în distanța metrică cu ajutorul anumitor ponderi

Alegerea parametrului k

Cei mai apropiați k vecini ai unei instanțe x sunt punctele cu distanțele cele mai mici față de x



Numărul de vecini

- Valoarea lui k este importantă:
  - dacă e prea mic, atunci clasificatorul poate fi suspectat de *overfitting*, pentru că devine prea senzitiv la zgomotul din datele de intrare (zgomot ⇒ date eronate)
    - clasificarea poate fi afectată de zgomot
  - dacă e prea mare, atunci s–ar putea ca prea mulți dintre cei k vecini considerați să fie depărtați de punctul curent şi deci irelevanți pentru clasificarea curentă
    - vecinătatea poate include puncte din alte clase

Argumente pro/contra

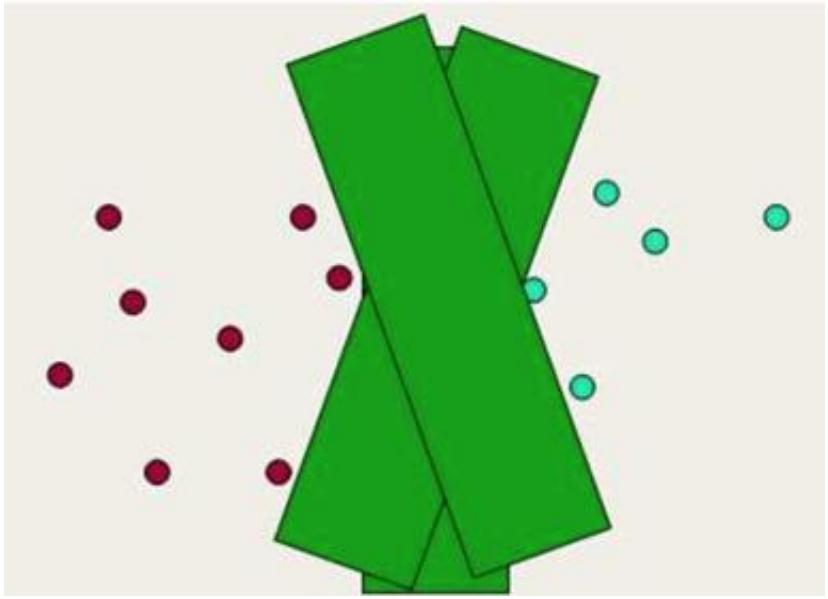
- Argumente pro
  - învățarea și implementarea sunt extrem de simple și intuitive
  - limite de decizie flexibile
- Argumente contra
  - caracteristicile irelevante sau corelate au un impact mare și trebuie eliminate
  - dimensionalitate mare este dificil de manevrat
  - costuri de calcul: calculul memoriei și al timpului de clasificare

Support Vectors Machine (SVM)

- Algoritm utilizat pentru sarcini de clasificare
- Obiectiv: identificarea hiperplanul optim într-un spațiu N-dimensional care separă punctele în diferite clase
  - Maximizarea marjei dintre cele mai apropiate puncte ale diferitelor clase
- Diminesiunea hiperplanului depinde de numărul de caracteristici (dimensionalitate)
  - Pentru 2 caracteristici – o linie
  - Pentru 3 caracteristici – plan 2D

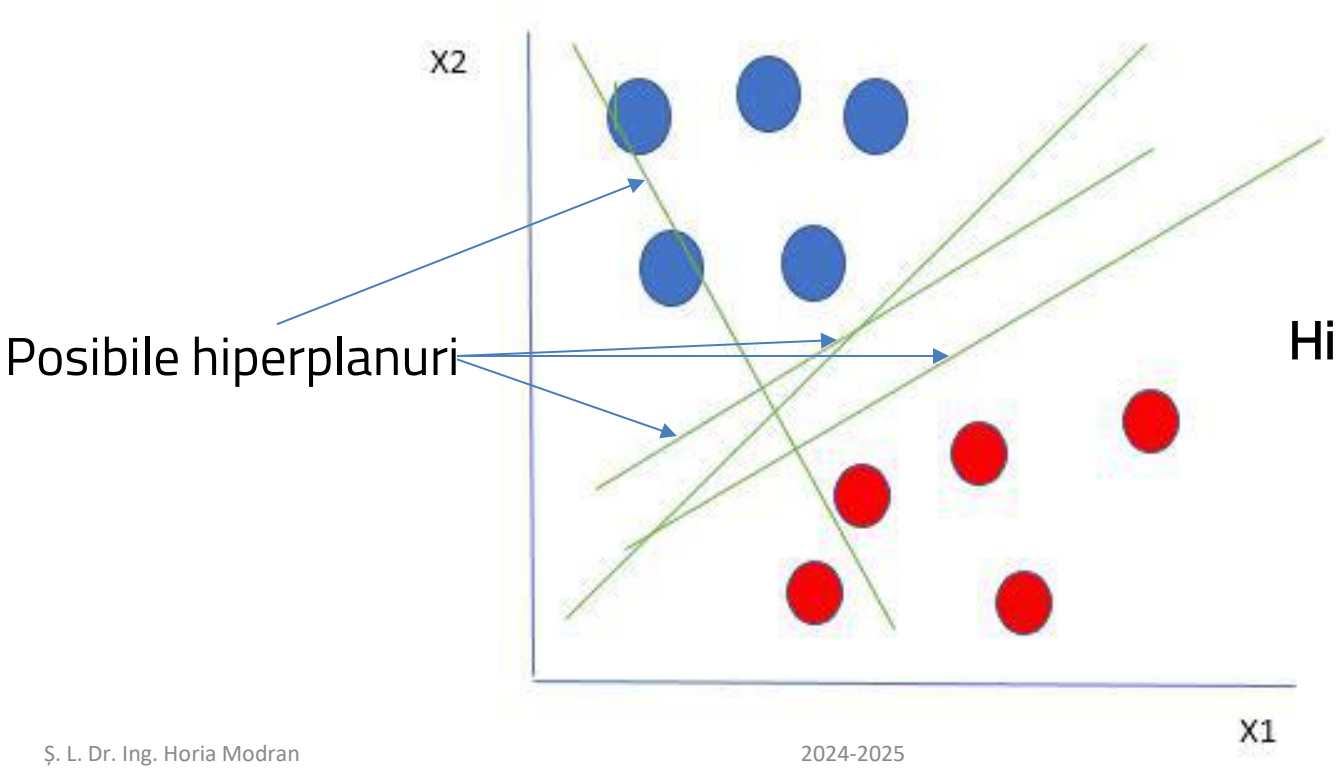
Support Vectors Machine (SVM)

- SVM separă clasele calculând o suprafață de decizie aflată la distanță maximă de punctele clasificate
- Exemplu ilustrat de suprafețe (benzi) de decizie posibile, colorate verde:

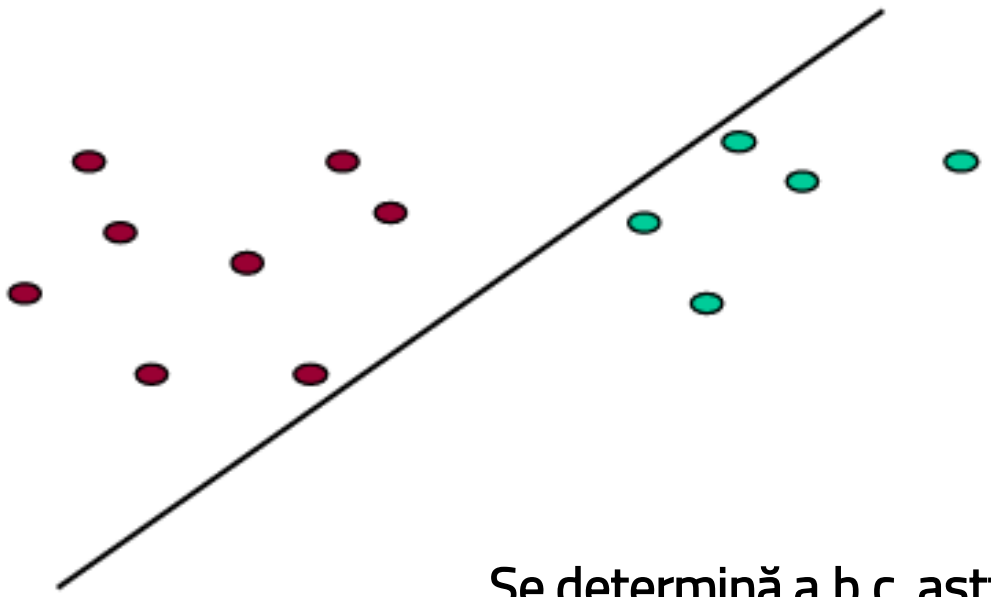


Support Vectors Machine (SVM)

- Considerând două variabile independente, x1 și x2, și o variabilă dependentă (cerc albastru/cerc roșu)



Caz 2D



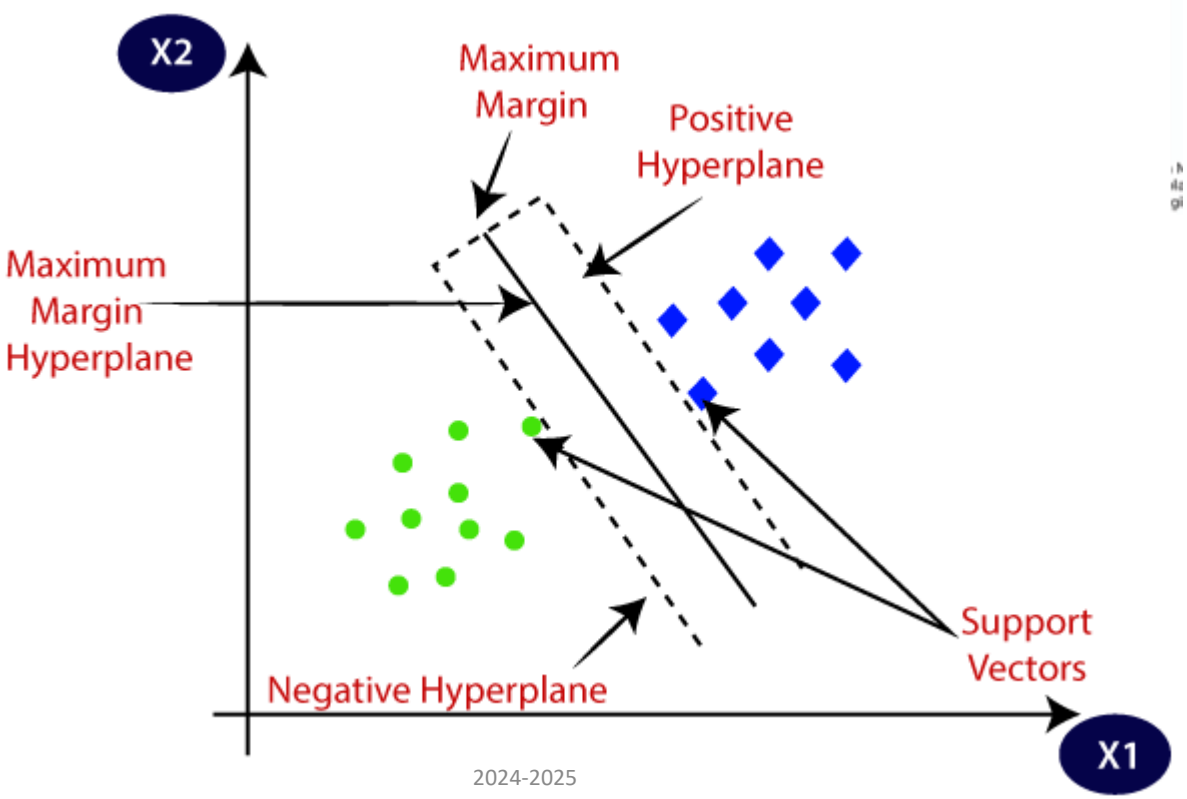
Se determină a,b,c, astfel încât:

$ax + by \geq c$ pentru punctele roșii

$ax + by \leq$ (sau $<$) c pentru punctele verzi

Termeni importanți

- **Support Vectors**: punctele cele mai apropiate de hiperplan.
- **Margin**: este distanța dintre hiperplan și observațiile cele mai apropiate de hiperplan



Algoritmul SVM

- Considerând o problemă de clasificare binară (+1 și -1)
- Avem un set de date de antrenament format din vectori caracteristici de intrare X și etichetele lor Y
- Ecuația hiperplanului este următoarea:

$$w^T X + b = 0$$

, unde W – vectorul normal la hiperplan (perpendiculara) și b - decalajul

- Pentru un clasificator SVM liniar

$$\hat{y} = \begin{cases} 1 & : \ w^T x + b \geq 0 \\ 0 & : \ w^T x + b < 0 \end{cases}$$

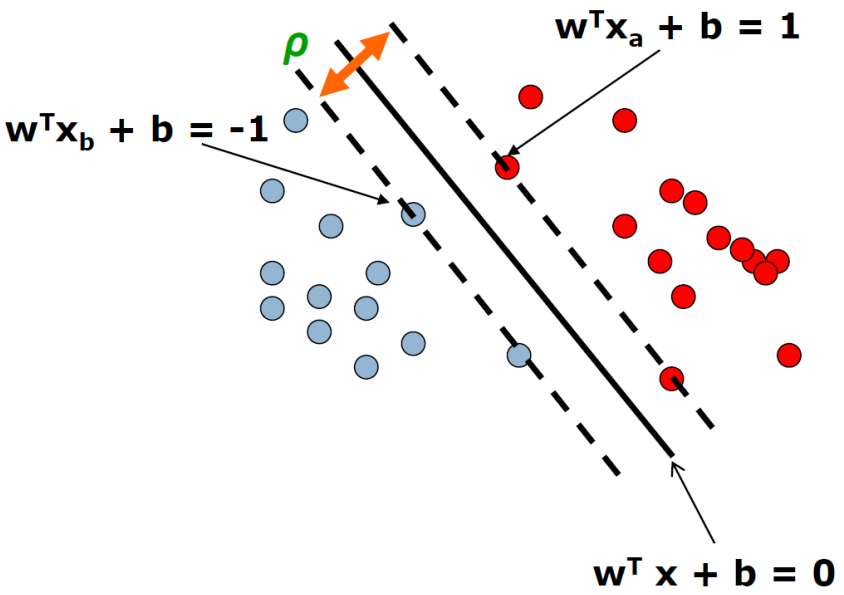
SVM Liniar

- presupunem că marja funcțională a fiecărui element de date este de cel puțin 1, apoi urmează două constrângeri pentru un set de antrenament $\{(x_i , y_i)\}$:

$$\mathbf{w}^T\mathbf{x_i} + b \geq 1 \quad \text{if } y_i = 1$$

$$\mathbf{w}^T\mathbf{x_i} + b \leq -1 \quad \text{if } y_i = -1$$

- pentru vectorii suport, inegalitatea devine o egalitate



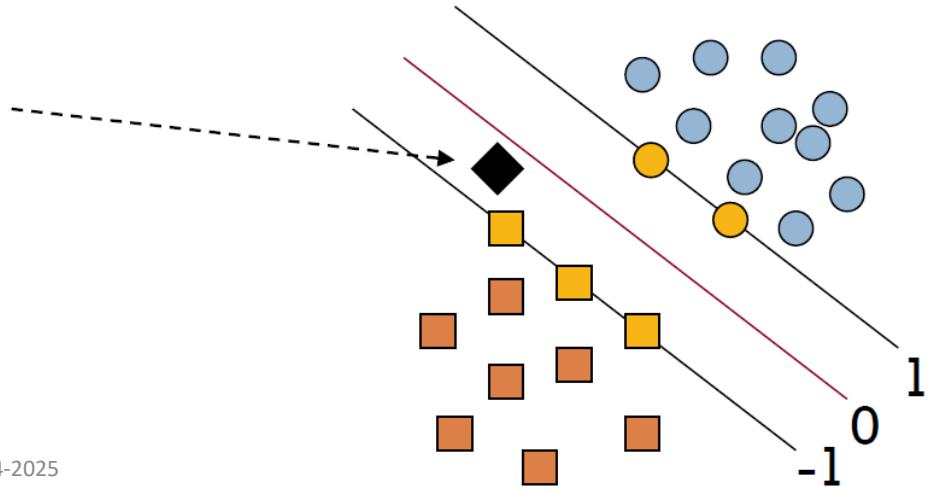
Hiperplan
$\mathbf{w}^T\mathbf{x} + b = 0$

Constrângere:
$\min_{i=1,\ldots,n} |\mathbf{w}^T\mathbf{x}_i + b| = 1$

Clasificare cu SVM

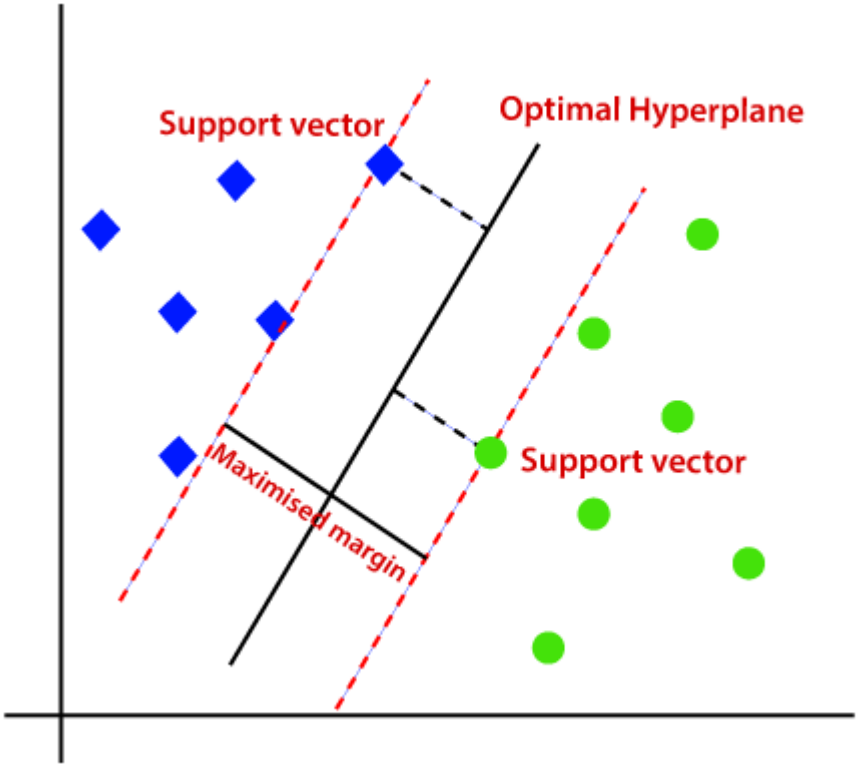
- Fiind dat un nou punct x , putem nota proiecția acestuia pe normala hiperplanului:
  - Se calculează după: $\mathbf{w}^T\mathbf{x} + b = \Sigma \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$
    - decide clasa în funcție de dacă este < sau > decat 0
- poate stabili un prag de încredere

Score > t: **da**
Score < $-t$: **nu**
Altfel: **?**



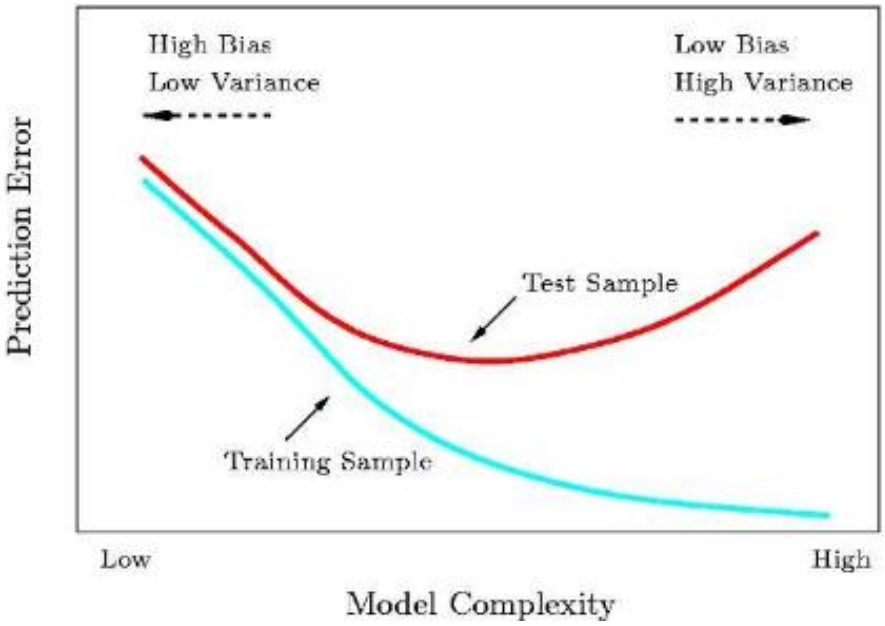
Exemplu SVM

- Să presupunem că avem un set de date care are două clase (verde și albastru). Vrem să clasificăm noile date ca fiind fie albastre, fie verzi.



Eroare antrenare vs testare

- Low bias/high variance – **overfitting** (“memorarea” exemplelor din setul de testare)
- High bias/low variance – **underfitting** (modelul nu poate învăța structura datelor)



Evaluarea modelelor

- Măsuri statistice
  - Acuratețea (*Accuracy*)
  - Precizia (*Precision)*
  - Recall
  - Scorul F1 (*F1-score*)
- **Eficiență**
  - În dezvoltarea modelului
  - În testarea modelului

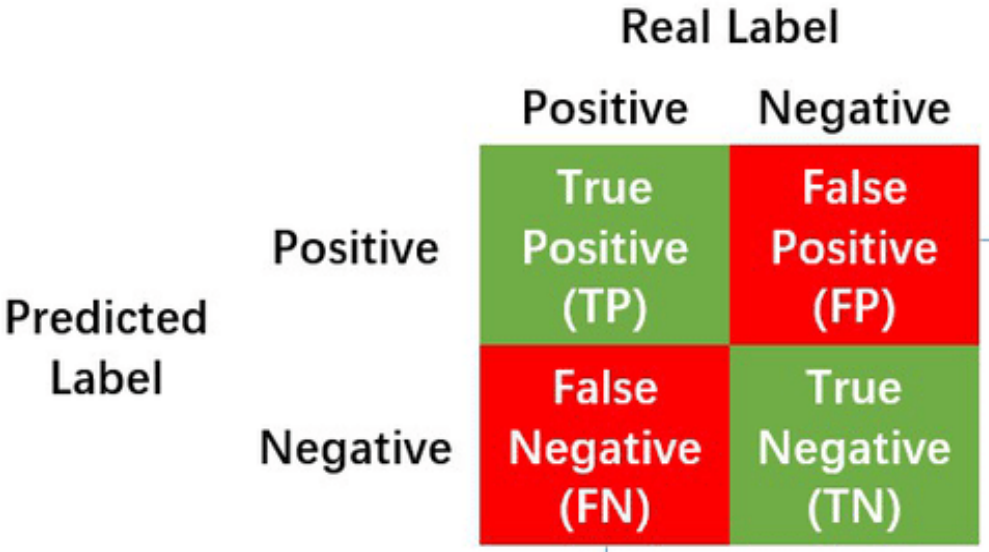
Măsuri statistice

- Acuratețea
  - Nr de exemple corect clasificate / nr total de exemple
  - Opusul erorii
  - Calculată pe: setul de validare, setul de testare
- Precizia (P)
  - nr. de exemple pozitive corect clasificate / nr. total de exemple clasificate ca pozitive
  - probabilitatea ca un exemplu clasificat pozitiv să fie relevant

Măsuri statistice

- Recall (R)
  - Nr. de exemple pozitive corect clasificate / nr. total de exemple pozitive
  - Probabilitatea ca un exemplu pozitiv să fie identificat corect de către clasificator
- Scorul F1
  - Combină precizia și recall-ul, facilitând compararea a 2 algoritmi
  - Media armonică dintre precizie și recall = 2PR/(P+R)

Metrici de performanță



$$\text{Precision} = \frac{\sum \text{TP}}{\sum \text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\sum \text{TP}}{\sum \text{TP} + \text{FN}}$$

$$\text{Accuracy} = \frac{\sum \text{TP} + \text{TN}}{\sum \text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

ÎNTREBĂRI ?