

# Regresie Liniară, Regresie Logistică

Olimpiada de AI. Session 07

Nume și Prenume: \_\_\_\_\_

Data: \_\_\_\_\_

## 1. Regresia Liniară

### 1.1 Formalizarea Matematică a Ipotezei

În Machine Learning, obiectivul regresiei liniare este de a aproxima relația dintre variabilele de intrare ( $x$ ) și variabila țintă ( $y$ ). Definim **ipoteza**  $h_{\theta}(x)$  (notată adesea și cu  $\hat{y}$  pentru a indica o valoare estimată) sub următoarea formă:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Unde:

- $\theta_0$  reprezintă **bias-ul** (termenul liber).
- $\theta_1, \theta_2, \dots, \theta_n$  reprezintă **ponderile** (*weights*) asociate fiecărei trăsături (*feature*).

Folosind algebra liniară, putem exprima această relație compact ca produsul scalar dintre vectorul de parametri  $\theta$  și vectorul de trăsături  $x$ :

$$\hat{y} = \theta^T x$$

### 1.2 Evaluarea Modelului: Funcția de Cost (MSE)

Pentru a măsura performanța modelului, utilizăm o **Funcție de Cost** care cuantifică diferența dintre valorile prezise ( $\hat{y}$ ) și valorile reale din setul de date ( $y$ ). Cea mai utilizată metodă este **Mean Squared Error (MSE)**:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2$$

Proprietățile funcției de cost:

- **Penalizarea erorilor:** Prin ridicarea la pătrat, erorile mari sunt penalizate mult mai drastic decât erorile mici.
- **Convexitate:** Funcția are o formă convexă, garantând un **minim global**.
- **Factorul  $\frac{1}{2}$ :** Convenție matematică pentru simplificarea derivatei.

### 1.3 Derivarea Matematică a Soluției Analitice

Cum găsim setul optim de parametri care minimizează eroarea  $J$ ? Calculăm derivatele parțiale și le egalăm cu 0.

## A. Cazul Simplu 2D (Determinarea $\beta_0, \beta_1$ )

Pentru o singură variabilă  $x$ , avem  $\hat{y} = \beta_0 + \beta_1 x$ . Funcția de cost (ignând constanta  $\frac{1}{2m}$  care nu afectează minimul) este:

$$E = \sum_{i=1}^m (\beta_0 + \beta_1 x^{(i)} - y^{(i)})^2$$

**1. Derivăm în raport cu  $\beta_0$  (Intercept):**

$$\begin{aligned} \frac{\partial E}{\partial \beta_0} &= 2 \sum_{i=1}^m (\beta_0 + \beta_1 x^{(i)} - y^{(i)}) = 0 \\ \sum_{i=1}^m \beta_0 + \beta_1 \sum_{i=1}^m x^{(i)} - \sum_{i=1}^m y^{(i)} &= 0 \end{aligned}$$

Stim că  $\sum_{i=1}^m \beta_0 = m\beta_0$ . Împărțind totul la  $m$  (numărul de exemple), obținem mediile  $(\bar{x}, \bar{y})$ :

$$\beta_0 + \beta_1 \bar{x} - \bar{y} = 0 \implies \boxed{\beta_0 = \bar{y} - \beta_1 \bar{x}}$$

**2. Derivăm în raport cu  $\beta_1$  (Panta):**

Calculăm derivata parțială în raport cu  $\beta_1$ , ținând cont că aceasta este înmulțită cu  $x^{(i)}$  în interiorul parantezei, și o egalăm cu 0:

$$\frac{\partial E}{\partial \beta_1} = \sum_{i=1}^m 2(\beta_0 + \beta_1 x^{(i)} - y^{(i)}) \cdot x^{(i)} = 0$$

Împărțim toată relația la 2 și facem înlocuirea  $\beta_0 = \bar{y} - \beta_1 \bar{x}$  (obținută anterior):

$$\sum_{i=1}^m (\underbrace{\bar{y} - \beta_1 \bar{x}}_{\beta_0} + \beta_1 x^{(i)} - y^{(i)}) x^{(i)} = 0$$

Grupăm termenii care conțin  $\beta_1$  și pe cei care conțin  $y$ :

$$\sum_{i=1}^m [\beta_1(x^{(i)} - \bar{x}) - (y^{(i)} - \bar{y})] x^{(i)} = 0$$

Desfacem suma în două părți și mutăm termenii cu  $y$  în partea dreaptă a egalului:

$$\begin{aligned} \beta_1 \sum_{i=1}^m (x^{(i)} - \bar{x}) x^{(i)} &= \sum_{i=1}^m (y^{(i)} - \bar{y}) x^{(i)} \\ \boxed{\beta_1 = \frac{\sum_{i=1}^m (x^{(i)} - \bar{x})(y^{(i)} - \bar{y})}{\sum_{i=1}^m (x^{(i)} - \bar{x})^2}} \end{aligned}$$

## B. [Avansat] Cazul General Multidimensional (Forma Matriceală)

Pentru  $n$  trăsături, scriem eroarea sub formă vectorială unde  $\hat{y} = X\theta$ :

$$J(\theta) = \frac{1}{2m} \|X\theta - y\|^2 = \frac{1}{2m} (X\theta - y)^T (X\theta - y)$$

Calculăm gradientul  $\nabla_\theta J(\theta)$  și îl egalăm cu 0:

$$\begin{aligned} \nabla_\theta J(\theta) &= \frac{1}{2m} \cdot 2X^T(X\theta - y) \\ \nabla_\theta J(\theta) &= 2X^T X \theta - 2X^T y = 0 \end{aligned}$$

Rezultă **Ecuatia Normală** (generalizarea formulelor de mai sus):

$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

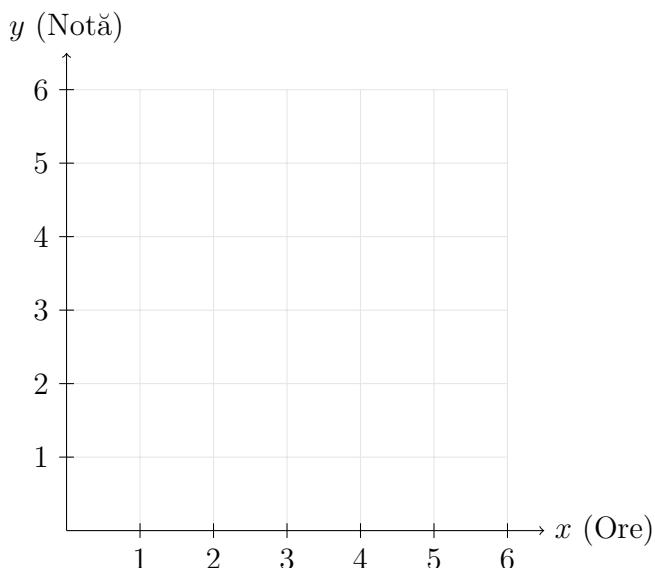
## 1.4 [Exercițiu] Vizualizare 2D

Se dă setul de date: (Ore studiu  $x$ , Notă  $y$ ):  $A(1, 2)$ ,  $B(2, 3)$  și  $C(3, 4)$ . Calculați panta  $\beta_1$  și interceptul  $\beta_0$  folosind:  $\beta_1 = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sum(x-\bar{x})^2}$ ,  $\beta_0 = \bar{y} - \beta_1 \bar{x}$ .

**Rezolvare:**

- $\bar{x} = \underline{\hspace{2cm}}$      $\bar{y} = \underline{\hspace{2cm}}$
- $\beta_1 = \underline{\hspace{2cm}}$
- $\beta_0 = \underline{\hspace{2cm}}$
- Ecuția dreptei:  $y = \underline{\hspace{2cm}}$

**Reprezentare Grafică:** Desenați punctele  $A, B, C$  și dreapta rezultată mai jos.



## 1.5 [Exercițiu] Regresia Liniară Multiplă (3D)

Trecerea la 3D presupune că avem două variabile de intrare  $(x_1, x_2)$ . Modelul nu mai este o dreaptă, ci un **plan de regresie** definit de ecuația:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

**Set de date:** Avem 3 observații de forma  $(x_1, x_2, y)$ :  $P_1(1, 1, 6)$ ,  $P_2(2, 1, 8)$ ,  $P_3(1, 2, 9)$ .

**Pasul 1: Construirea Matricelor**

Pentru a folosi **Ecuția Normală**  $\theta = (X^T X)^{-1} X^T y$ , trebuie să organizăm datele sub formă matricială.

$$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \quad y = \begin{bmatrix} 6 \\ 8 \\ 9 \end{bmatrix}$$

\**Observație: Prima coloană de 1 corespunde bias-ului ( $\theta_0$ ).*

## Pasul 2: Produsele Matriciale ( $X^T X$ și $X^T y$ )

Transpusa unei matrice ( $X^T$ ) se obține transformând liniile în coloane:

$$X = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \implies X^T = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

### A. Calculul $X^T X$ :

$$X^T X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}^T \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 4 & 4 \\ 4 & 6 & 5 \\ 4 & 5 & 6 \end{bmatrix}$$

### B. Calculul $X^T y$ :

$$X^T y = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}^T \begin{bmatrix} 6 \\ 8 \\ 9 \end{bmatrix} = \begin{bmatrix} 23 \\ 31 \\ 32 \end{bmatrix}$$

## Pasul 3: Calculul Inversei Matricei $A = X^T X$

Pentru a inversa matricea  $A$ , folosim relația:

$$A^{-1} = \frac{1}{\det(A)} \cdot \text{adj}(A)$$

Calculul Determinantului ( $\det A$ )

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \implies \det(A) = (aei + bfg + cdh) - (ceg + afh + bdi)$$

$$\det(A) = (3 \cdot 6 \cdot 6 + 4 \cdot 5 \cdot 4 + 4 \cdot 4 \cdot 5) - (4 \cdot 6 \cdot 4 + 4 \cdot 5 \cdot 3 + 6 \cdot 4 \cdot 4) = 268 - 267 = 1$$

Matricea adjunctă  $\text{adj}(A)$  se obține prin calcularea cofactorilor  $c_{ij} = (-1)^{i+j} \cdot \text{Minor}_{ij}$ . Pentru fiecare element, semnul este dat de  $(-1)^{i+j}$ , unde  $i$  este linia și  $j$  coloana.

$$\begin{bmatrix} (-1)^{1+1} & (-1)^{1+2} & (-1)^{1+3} \\ (-1)^{2+1} & (-1)^{2+2} & (-1)^{2+3} \\ (-1)^{3+1} & (-1)^{3+2} & (-1)^{3+3} \end{bmatrix} \rightarrow \begin{bmatrix} + & - & + \\ - & + & - \\ + & - & + \end{bmatrix}$$

Minorul unui element (notat cu  $\text{Minor}_{ij}$ ) este determinantul care rămâne după ce elimină complet linia  $i$  și coloana  $j$  pe care se află acel element.

**Exemple de calcul (primul rând):**

$$\bullet \quad c_{11} = + \begin{vmatrix} 6 & 5 \\ 5 & 6 \end{vmatrix} = (6 \cdot 6 - 5 \cdot 5) = 11$$

$$\bullet \quad c_{12} = - \begin{vmatrix} 4 & 5 \\ 4 & 6 \end{vmatrix} = -(24 - 20) = -4$$

$$\bullet \quad c_{13} = + \begin{vmatrix} 4 & 6 \\ 4 & 5 \end{vmatrix} = (20 - 24) = -4$$

După calcularea tuturor celor 9 elemente și transpunerea lor, obținem:

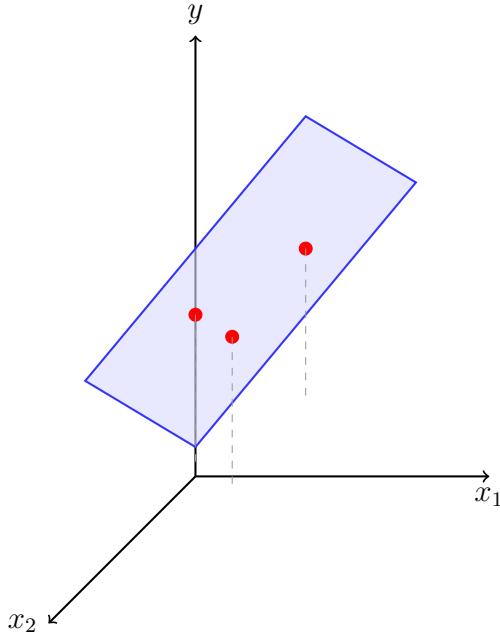
$$(X^T X)^{-1} = \begin{bmatrix} 11 & -4 & -4 \\ -4 & 2 & 1 \\ -4 & 1 & 2 \end{bmatrix}$$

#### Pasul 4: Soluția Finală (Calculul $\theta$ )

Înmulțim inversa  $(X^T X)^{-1}$  cu vectorul  $X^T y$  pentru a obține ponderea fiecărei trăsături:

$$\theta = \begin{bmatrix} 11 & -4 & -4 \\ -4 & 2 & 1 \\ -4 & 1 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 23 \\ 31 \\ 32 \end{bmatrix} = \begin{bmatrix} 11 \cdot 23 + (-4) \cdot 31 + (-4) \cdot 32 \\ -4 \cdot 23 + 2 \cdot 31 + 1 \cdot 32 \\ -4 \cdot 23 + 1 \cdot 31 + 2 \cdot 32 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Rezultă parametrii:  $\theta_0 = 1$  (bias),  $\theta_1 = 2$ ,  $\theta_2 = 3$ . Ecuația:  $\hat{y} = 1 + 2x_1 + 3x_2$ .



## 2. Regularizarea: Evitarea Overfitting-ului

O problemă frecventă în antrenarea modelelor este **Overfitting-ul**: modelul învăță ”pe de rost” datele de antrenament (inclusiv zgromotul), dar nu performează bine pe date noi. Soluția este **Regularizarea**: adăugăm un termen de penalizare în Funcția de Cost pentru a păstra parametrii  $\theta$  la valori mici.

### 2.1 Regresia Ridge (Regularizare L2)

Adăugăm suma pătratelor parametrilor la funcția de cost. Aceasta forțează ponderile să fie mici, dar nenule.

$$J(\theta)_{Ridge} = \underbrace{\text{MSE}}_{\text{Eroare}} + \lambda \underbrace{\sum_{j=1}^n \theta_j^2}_{\text{Penalizare L2}}$$

**Ecuația Normală cu Regularizare L2:** Pentru a rezolva direct, modificăm matricea  $X^T X$  adăugând o matrice identitate ponderată (Ex. avansat: Calculați derivata, egalati cu 0 și vedeti dacă obțineti Ecuația Normală de mai jos):

$$\theta = (X^T X + \lambda I)^{-1} X^T y$$

Notă:  $\lambda$  este un hiperparametru care controlează cât de mult penalizăm modelul.

## 2.2 Regresia Lasso (Regularizare L1)

Adăugăm suma modulelor parametrilor. Această metodă are proprietatea unică de a putea reduce unii parametri exact la 0, realizând astfel și o **selectie de trăsături** (elimină variabilele inutile).

$$J(\theta)_{Lasso} = \text{MSE} + \lambda \sum_{j=1}^n |\theta_j|$$

**Observație:** Spre deosebire de Ridge, Lasso **nu are o formulă de calcul direct** (o Ecuație Normală). Motivul este că funcția modul  $|\theta|$  **nu este derivabilă** în punctul 0. Nu putem aplica metoda egalării gradientului cu 0, ci trebuie să folosim metode iterative (Gradient Descent).

| Tip        | Penalizare      | Efect Principal                       |
|------------|-----------------|---------------------------------------|
| Ridge (L2) | $\sum \theta^2$ | Ponderi mici (previne valori extreme) |
| Lasso (L1) | $\sum  \theta $ | Sparsitate (unele $\theta_j$ devin 0) |

## 3. Optimizarea prin Gradient Descent (GD)

Înainte de a trece la Regresia Logistică, trebuie să înțelegem cum „învăță” un model AI. Procesul de învățare este, în esență, o problemă de optimizare: căutăm valorile parametrilor care minimizează eroarea. Cel mai utilizat algoritm pentru aceasta este **Gradient Descent**.

### 3.1 Formalizarea Algoritmului

Fie  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  o funcție derivabilă (funcția de cost). Algoritmul pornește de la un punct inițial  $x^{(0)} \in \mathbb{R}^d$  și actualizează iterativ poziția pentru a coborî spre minimul funcției.

**Regula de actualizare scalară** (pentru fiecare coordonată  $i$ ):

$$x_i^{(k+1)} \leftarrow x_i^{(k)} - \eta \frac{\partial}{\partial x_i} f(x^{(k)})$$

**Regula de actualizare vectorială:**

$$x^{(k+1)} \leftarrow x^{(k)} - \eta \nabla_x f(x^{(k)})$$

Unde:

- $\eta > 0$  este **rata de învățare** (*learning rate*).
- $\nabla_x f(x^{(k)})$  este **vectorul gradient**, format din toate derivatele parțiale calculate în punctul curent:

$$\nabla_x f(x^{(k)}) = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_d} \right)^\top$$

### 3.2 Derivate Necessare

Pentru a implementa Gradient Descent, trebuie să știm să calculăm derivatele funcțiilor obiectiv.

| Funcția   | Derivata   | Funcția   | Derivata  |
|---|------------|-----------|-----------|
| $c$   | 0          | $e^x$     | $e^x$     |
| $x^n$   | $nx^{n-1}$ | $\sin(x)$ | $\cos(x)$ |
| <b>Regula lanțului:</b> $[f(g(x))]' = f'(g(x)) \cdot g'(x)$ |            |           |           |

### 3.3 [Exercițiu] Aplicații Practice

#### A. Aplicare în $\mathbb{R}$ (Cazul 1D)

Fie funcția  $f(x) = 4x^2 - 2x + 1$  și rata de învățare  $\eta = 0.1$ . Derivata funcției este  $f'(x) = 8 \dots \dots \dots$

**Pasul 0:** Pornim de la  $x^{(0)} = 1$ .

- Panta:  $f'(1) = 8(1) - 2 = 6$
- Actualizare:  $x^{(1)} = 1 - (0.1 \cdot 6) = 0.4$

**Pasul 1:** Pornim de la  $x^{(1)} = 0.4$ .

- Panta:  $f'(0.4) = 8 \cdot (\dots \dots \dots) - 2 = \dots \dots \dots$
- Actualizare:  $x^{(2)} = 0.4 - (0.1 \cdot \dots \dots \dots) = \dots \dots \dots$

Calculati  $f(x^{(0)})$ ,  $f(x^{(1)})$ ,  $f(x^{(2)})$ . Ce observati?

---

---

---

---

#### B. Aplicare în $\mathbb{R}^2$ (Cazul 2D)

Fie funcția  $f(x_1, x_2) = x_1^2 + \sin(x_1 + x_2) + x_2^2$  și rata de învățare  $\eta = 0.4$ .

**Derivatele parțiale:**  $\frac{\partial f}{\partial x_1} = 2x_1 + \cos(x_1 + x_2)$ ,  $\frac{\partial f}{\partial x_2} = 2x_2 + \cos(x_1 + x_2)$

**Iterația 1:** Pornim de la punctul  $(x_1 = 3, x_2 = -3)$ .

- $\frac{\partial f}{\partial x_1} = 2(3) + \cos(3 - 3) = 6 + 1 = 7$
- $\frac{\partial f}{\partial x_2} = 2(-3) + \cos(3 - 3) = -6 + 1 = -5$
- $x_1^{(1)} = 3 - 0.4 \cdot (7) = 0.2$
- $x_2^{(1)} = -3 - 0.4 \cdot (-5) = -1$

**Iterația 2:** Pornim de la punctul  $(0.2, -1)$ .

Folositi  $\cos(0.2 - 1) = \cos(-0.8) \approx 0.7$

1. Noile pante:

- $\frac{\partial f}{\partial x_1} = \dots \dots \dots$
- $\frac{\partial f}{\partial x_2} = \dots \dots \dots$

2. Noile coordonate:

- $x_1^{(2)} = 0.2 - 0.4 \cdot (\dots \dots \dots) = \dots \dots \dots$
- $x_2^{(2)} = -1 - 0.4 \cdot (\dots \dots \dots) = \dots \dots \dots$

**Observații despre Hyperparametri:** Dacă rata de învățare este prea mare (ex:  $\eta = 0.8$ ), pașii pot depăși punctul de minim, fenomen cunoscut sub numele de **overshooting**. În acest caz, algoritmul poate oscila sau chiar diverge.

## 4. Regresia Logistică și Clasificarea

### 4.1 Diferențe fundamentale: Regresie Liniară vs. Regresie Logistică

Deși partajează denumirea de „regresie”, cele două modele servesc scopuri complet diferite:

- **Scopul modelării:** În timp ce Regresia Liniară încearcă să aproximeze o dreaptă care să treacă cât mai aproape de toate punctele de date, Regresia Logistică învață o dreaptă (sau un plan) care **separă** cel mai bine două clase de puncte.
- **Valorile de ieșire:** Regresia Liniară returnează valori continue, putând produce rezultate precum 1.5 sau -0.2, care nu au sens într-o problemă de clasificare. Regresia Logistică transformă orice rezultat într-o probabilitate situată strict între 0 și 1.

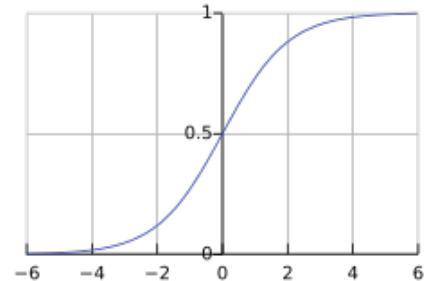
### 4.2 Funcția Sigmoid (Logistică)

Pentru a măpa orice valoare reală în intervalul  $(0, 1)$ , aplicăm funcția **Sigmoid**:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Unde  $z = \theta^T x$ . Această funcție mapează orice valoare în  $(0, 1)$  și transformă distanța față de graniță de separație într-o probabilitate:

- Dacă  $z \rightarrow \infty$ , atunci  $\sigma(z) \rightarrow 1$ .
- Dacă  $z \rightarrow -\infty$ , atunci  $\sigma(z) \rightarrow 0$ .
- Dacă  $z = 0$ , atunci  $\sigma(z) = 0.5$  (punctul de incertitudine).



### 4.3 Granița de Decizie (Decision Boundary)

Granița de decizie reprezintă locul geometric unde modelul trece de la o clasificare la alta, adică unde probabilitatea este 0.5 ( $z = 0$ ).

#### Cazul 2D: Dreapta de Decizie

Pentru două trăsături  $(x_1, x_2)$ , granița este definită de ecuația:

$$\theta_1 x_1 + \theta_2 x_2 + \theta_0 = 0$$

Aceasta reprezintă o **dreaptă** care separă setul de date în două clase.

#### Cazul 3D: Planul de Decizie

Pentru trei trăsături  $(x_1, x_2, x_3)$ , granița devine un **plan** în spațiu:

$$\theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_0 = 0$$

*Analogie:* Într-un spațiu 3D, această graniță acționează ca un plan care separă două grupuri de puncte (precum un perete separă obiectele din bucătărie de obiectele din dormitor).

#### Generalizarea: Hiperplanul

În spații cu  $n$  dimensiuni, unde nu mai putem vizualiza geometria, granița rămâne un obiect liniar numit **hiperplan**, definit algebric prin produsul scalar:

$$\theta^T x = 0$$

## 5. Optimizarea Regresiei Logistice prin Metodele Gradientului și Newton-Raphson

Considerăm un set de date de antrenament  $\{(x^{(i)}, y^{(i)})\}_{i=1,2} \in \mathbb{R}$ , unde vectorul de trăsături include termenul de bias:  $x_1^{(i)} = (1, x_1^{(i)})^\top$ . Eticheta  $y^{(i)} \in \{0, 1\}$ . Obiectivul este determinarea vectorului de ponderi  $w = (w_0, w_1)^\top$  care maximizează verosimilitatea datelor.

| $i$ | $x_1^{(i)}$ | $y^{(i)}$ |
|-----|-------------|-----------|
| 1   | -1          | 0         |
| 2   | +1          | 1         |

### 5.1 Formalizarea Funcției Obiectiv

Modelul probabilistic este definit de probabilitatea condiționată:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \rightarrow p(y = 1|x; w) = \sigma(w \cdot x) = \frac{1}{1 + \exp(-w_0 - w_1 x_1)}$$

[Exercitiu 5.1.1] Scrieți funcția de log-verosimilitate  $\ell(w)$  pentru acest set de date.

$$\ell(w) = \sum_{i=1}^n [y^{(i)} \ln \sigma(w \cdot x^{(i)}) + (1 - y^{(i)}) \ln(1 - \sigma(w \cdot x^{(i)}))]$$

Folosind proprietatea  $\ln(1 - \sigma(z)) = -\ln(1 + \exp(z))$ , funcția devine urmatoarea:

(Tip: încercați să desfaceti paranteza  $(1 - y^{(i)})$  de mai sus.)

$$\ell(w) = \sum_{i=1}^n [y^{(i)}(w_0 + w_1 x_1^{(i)}) - \ln(1 + \exp(w_0 + w_1 x_1^{(i)}))]$$

[Exercitiu 5.1.2] Completați pentru setul nostru de date:

$$\ell(w) = \underbrace{0 \cdot (w_0 - w_1) - \ln(1 + \exp(w_0 - w_1))}_{\text{Punctul } i=1} + \underbrace{1 \cdot (\dots) - \ln(1 + \exp(\dots))}_{\text{Punctul } i=2}$$

### 5.2 Calculul Vectorului Gradient

Gradientul funcției  $\ell(w)$  în raport cu ponderile  $w_j$ . Puteți vedea cum se ajunge la această formulă în Anexă.

$$\frac{\partial \ell(w)}{\partial w_j} = \sum_{i=1}^n (y^{(i)} - \sigma(w \cdot x^{(i)})) x_j^{(i)}$$

[Exercitiu 5.2.1] Calculați vectorul gradient  $\nabla_w \ell(w)$  pornind de la inițializarea  $w^{(0)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ .

$$1. \quad \sigma(w^{(0)} \cdot x^{(1)}) = \sigma(0 - 0) = 0.5$$

$$2. \quad \sigma(w^{(0)} \cdot x^{(2)}) = \sigma(0 + 0) = 0.5$$

$$3. \quad \nabla_w \ell(w^{(0)}) = [0 - 0.5] \begin{pmatrix} 1 \\ -1 \end{pmatrix} + [1 - 0.5] \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \dots \\ \dots \end{pmatrix} + \begin{pmatrix} \dots \\ \dots \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

### 5.3 Gradient Ascendent vs. Newton-Raphson

[Exercitiu 5.3.1] Gradient Ascendent ( $\eta = 0.1$ ).

Actualizați ponderile:  $w^{(1)} = w^{(0)} + \eta \nabla_w \ell(w^{(0)})$ .

$$w^{(1)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0.1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0.1 \end{pmatrix}$$

[Exercitiu 5.3.2] Metoda Newton-Raphson Aceasta folosește matricea Hessiană  $H_w$ .

Dacă notăm  $\sigma_i = \sigma(w \cdot x^{(i)})$ , formula generală este:

$$H_w = - \sum_{i=1}^n \sigma_i (1 - \sigma_i) x^{(i)} (x^{(i)})^\top$$

$$H_{w^{(0)}} = -\frac{1}{4} \left[ \begin{pmatrix} 1 \\ -1 \end{pmatrix} (1, -1) + \begin{pmatrix} 1 \\ 1 \end{pmatrix} (1, 1) \right] = -\frac{1}{4} \left[ \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \right] = -\frac{1}{2} I_2$$

Actualizați ponderile:  $w^{(1)} = w^{(0)} - (H_{w^{(0)}})^{-1} \nabla_w \ell(w^{(0)})$ .

$$w_{Newton}^{(1)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - (-2I_2) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

### 5.4 Analiza rezultatelor

Comparați valorile funcției  $\ell(w)$  după prima iterare. Observați cum Metoda Newton realizează un progres mult mai mare către maxim.

- $\ell(w_{Gradient}^{(1)}) = 2 \ln \sigma(0.1) \approx -1.28$
- $\ell(w_{Newton}^{(1)}) = 2 \ln \sigma(2) \approx -0.25$

[Exercitiu 5.4.1] Rulați încă câte două iterații pentru fiecare dintre cele două metode: Gradient Ascendent și Newton-Raphson.

## 6. Temă

Vom extinde modelul pentru a clasifica obiecte bazate pe două caracteristici,  $x_1$  și  $x_2$ . Vectorul de trăsături va fi  $x = (1, x_1, x_2)^\top$ , iar ponderea va fi  $w = (w_0, w_1, w_2)^\top$ .

| $i$ | $x_1$ (Mărime) | $x_2$ (Culoare) | $y$ (Etichetă) |
|-----|----------------|-----------------|----------------|
| 1   | -1             | 0               | 0              |
| 2   | 0              | -1              | 0              |
| 3   | 1              | 0               | 1              |
| 4   | 0              | 1               | 1              |

Cerințe:

- Inițializare:** Considerați  $w^{(0)} = [0, 0, 0]^\top$ . Calculați probabilitatea  $\sigma(w^{(0)} \cdot x^{(i)})$  pentru fiecare din cele 4 puncte.
- Vectorul Gradient:** Folosind formula  $\nabla_w \ell(w) = \sum_{i=1}^4 (y^{(i)} - \sigma(w \cdot x^{(i)}))x^{(i)}$ , calculați componentele gradientului:

$$\nabla_w \ell(w^{(0)}) = \begin{bmatrix} \dots \\ \dots \\ \dots \end{bmatrix}$$

- Actualizare:** Aplicați o iterație de Gradient Ascendent cu  $\eta = 0.2$  pentru a găsi  $w^{(1)}$ .

$$w^{(1)} = w^{(0)} + 0.2 \cdot \nabla_w \ell(w^{(0)})$$

- Predictie:** Verificați dacă un punct nou  $P_{test}$  cu trăsăturile  $(x_1 = 1, x_2 = 1)$  este clasificat corect (Verificați dacă  $w^{(1)} \cdot x > 0$ ).
- Repetați pașii 2-3 de cel puțin 3 ori. Ce observați? Dar dacă în loc de  $\eta = 0.2$ , ati folosi  $\eta = 0.02$ ?

*Sugestie pentru calcul:* Observați simetria punctelor față de origine. La inițializarea cu zero, toate valorile  $\sigma$  vor fi 0.5, ceea ce va simplifica calculul erorii  $(y^{(i)} - 0.5)$ .

## Anexă: Derivarea Gradientului pentru Regresia Logistică

Calculul gradientului se bazează pe aplicarea regulii de derivare a funcțiilor compuse și pe o proprietate a funcției Sigmoid.

### A. Derivata funcției Sigmoid

Fie  $\sigma(z) = \frac{1}{1+e^{-z}}$ . Derivata sa în raport cu  $z$  este:

$$\frac{d\sigma}{dz} = \frac{d}{dz}(1 + e^{-z})^{-1} = -(1 + e^{-z})^{-2} \cdot (-e^{-z}) = \frac{e^{-z}}{(1 + e^{-z})^2}$$

Putem rescrie expresia pentru a obține o formă recursivă:

$$\begin{aligned} \frac{e^{-z}}{(1 + e^{-z})^2} &= \frac{1}{1 + e^{-z}} \cdot \left( \frac{e^{-z}}{1 + e^{-z}} \right) = \sigma(z) \cdot \left( \frac{1 + e^{-z} - 1}{1 + e^{-z}} \right) \\ \implies \sigma'(z) &= \sigma(z)(1 - \sigma(z)) \end{aligned}$$

### B. Derivarea Log-Verosimilității

Considerăm funcția de cost pentru un singur exemplu, unde  $z = w^T x$ :

$$\ell(w) = y \ln \sigma(z) + (1 - y) \ln(1 - \sigma(z))$$

Aplicăm regula lanțului pentru a deriva în raport cu o pondere  $w_j$ :

$$\frac{\partial \ell}{\partial w_j} = \frac{\partial \ell}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial z} \cdot \frac{\partial z}{\partial w_j}$$

Calculând componentele individuale:

$$\begin{aligned} \frac{\partial \ell}{\partial \sigma} &= \frac{y}{\sigma} - \frac{1-y}{1-\sigma} = \frac{y(1-\sigma) - \sigma(1-y)}{\sigma(1-\sigma)} = \frac{y-\sigma}{\sigma(1-\sigma)} \\ \frac{\partial z}{\partial w_j} &= x_j \end{aligned}$$

Substituind în expresia regulii lanțului și folosind derivata sigmoidului calculată la punctul A:

$$\frac{\partial \ell}{\partial w_j} = \left[ \frac{y-\sigma}{\sigma(1-\sigma)} \right] \cdot [\sigma(1-\sigma)] \cdot x_j$$

Termenul  $\sigma(1 - \sigma)$  se simplifică, rezultând gradientul pentru un singur exemplu:

$$\frac{\partial \ell}{\partial w_j} = (y - \sigma)x_j$$

Prin însumarea peste toate cele  $n$  exemple, obținem formula finală vectorială:

$$\nabla_w \ell(w) = \sum_{i=1}^n (y^{(i)} - \sigma(w \cdot x^{(i)})) x^{(i)}$$