Laborator 2

Lucian M. Sasu

1 martie 2020

1 Regresie liniară

1. Să se scrie în Python o funcţie care să primească la intrare o matrice $\mathbf{X}$ şi să returneze matricea scalată, astfel: dacă matricea $\mathbf{X}$ primită ca parametru este

$$\mathbf{X} = \begin{pmatrix} x_0^{(1)} & x_1^{(1)} & \dots & x_{n-1}^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \dots & x_{n-1}^{(2)} \\ \dots & \dots & \dots & \dots \\ x_0^{(m)} & x_1^{(m)} & \dots & x_{n-1}^{(m)} \end{pmatrix} \tag{1}$$

atunci:

   (a) pentru fiecare coloană $0 \leq j \leq n-1$ determinăm minimul şi maximul ei:

$$min_j = \min_{i=0,\dots,m-1} x_j^{(i)} \tag{2}$$

   respectiv

$$max_j = \max_{i=0,\dots,m-1} x_j^{(i)} \tag{3}$$

   (b) fiecare element al lui $\mathbf{X}$ se va împărţi la diferenţa dintre maximul şi minimul coloanei pe care se găseşte:

$$y_j^{(i)} = \frac{x_j^{(i)} - min_j}{max_j - min_j} \tag{4}$$

   Note:

      i. Coloanele constante vor fi inlocuite cu valoarea 1.
      ii. După aplicarea în ordine a celor două operaţii de mai sus pe o matrice de intrare $\mathbf{X}$, vom obţine o matrice având componentele în intervalul $[0, 1]$.

iii. Se cere implementarea folosind cod vectorizat, folosind numpy, vedeţi Prezentare NumPy.

Funcţia pe care o scrieţi va returna tripla: matricea cu elemente scalate şi vectorii $(min_0, \ldots, min_{n-1})$, $(max_1, \ldots, max_{n-1})$, conţinând respectiv minimele şi maximele coloanelor.

2. Să se scrie o a doua funcţie care să primească drept parametru o matrice $\mathbf{X}$ de forma:

$$\mathbf{X} = \begin{pmatrix} x_0^{(1)} & x_1^{(1)} & \ldots & x_{n-1}^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \ldots & x_{n-1}^{(2)} \\ \ldots & \ldots & \ldots & \ldots \\ x_0^{(m)} & x_1^{(m)} & \ldots & x_{n-1}^{(m)} \end{pmatrix} \tag{5}$$

si să returneze o matrice cu $n+1$ coloane, prima fiind coloana cu valoare 1 iar restul fiind cele din $\mathbf{X}$:

$$\mathbf{Z} = \begin{pmatrix} 1 & x_0^{(1)} & x_1^{(1)} & \ldots & x_{n-1}^{(1)} \\ 1 & x_0^{(2)} & x_1^{(2)} & \ldots & x_{n-1}^{(2)} \\ 1 & \ldots & \ldots & \ldots & \ldots \\ 1 & x_0^{(m)} & x_1^{(m)} & \ldots & x_{n-1}^{(m)} \end{pmatrix} \tag{6}$$

3. Să se folosească metoda de minimizare folosind căutarea bazată pe gradient, pentru a determina model de predicţie bazat pe regresie liniară. Se va folosi setul de date din secţiunea 2. Se va face în prealabil scalarea datelor, urmată de adăugarea unei coloane de valori 1, folosind funcţiile de la punctele precedente. Se va reprezenta grafic evoluţia funcţiei de eroare J, pentru a permite eventuale ajustări manuale ale ratei de învăţare α (vezi observaţiile din curs). Se vor scrie functii pentru:

   (a) calculul valorii estimate de model pentru un set de date de intrare, $h_\theta(\mathbf{X})$

   (b) calculul functiei de cost, $J(\theta)$

   (c) calculul gradientului funcţiei de cost

   (d) efectuarea pasilor din algoritmul de instruire

   (e) calculul erorii patratice medii

   Urmăriţi scheletul funcţiilor din notebook-ul Jupyter si completati codul lipsă. Aserţiunile trebuie să fie îndeplinite.

4. Să se calculeze prin metoda pseudoinversei coeficienţii regresiei liniare pentru setul de date din secţiunea 2.

   Coeficienţii funcţiei de regresie se calculează cu formula:

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{pmatrix} = \left(\mathbf{X}^t\mathbf{X}\right)^{-1}\mathbf{X}^t y \tag{7}$$

   unde $\mathbf{X}$ este matricea obţinută după aplicarea funcţiei de la punctul 2. Pentru pseudoniversă se poate folosi funcţia numpy.linalg.pinv. Funcţia de regresie liniară este $h_\theta : \mathbf{R}^{n+1} \to \mathbf{R}$,

$$h_\theta(\mathbf{x}) = \boldsymbol{\theta}^t \cdot \mathbf{x} \tag{8}$$

   Pentru fiecare vector $\mathbf{x}^{(i)}$ din setul de instruire se va afişa valoarea prezisă de (8), împreună cu valoarea efectivă din setul de instruire. Se va calcula şi afişa în final funcţia de eroare pătratică medie.

   Observaţie: pentru determinarea vectorului $\boldsymbol{\theta}$ nu e obligatoriu a se face scalarea de la punctul 1 – a se vedea cursul.

   Întrebare: de ce ponderile ponderilor returnate de metoda pseudoinversei diferă de valorile calculate prin metoda bazată pe gradient?

5. De regulă, performanţa unui model nu se măsoară pe datele de instruire. Ne interesează mai mult puterea lui de generalizare, *i.e.* capacitatea de a face predicţii pentru date similare cu setul de instruire, dar neutilizate în instruire. Una din metodele acceptate de măsurare a performanţei este:

   - se face o permutare aleatoare a setului de date, de exemplu folosind funcţia np.random.permutation.
   - se împarte setul de la punctul precedent în două subseturi, primele 70% din date sunt folosite doar pentru antrenare, restul doar pentru testare;
   - se determină parametrii de scalare de pe setul de antrenare şi se aplică pe setul de testare; în urma acestui pas, setul de antrenare va avea în mod cert valori în $[0, 1]$; posibil ca şi setul de testare să capete aceeaşi proprietate în urma scalării, dar nu e garantat (explicaţi de ce);

- pe setul de antrenare se construieşte model de regresie;
- folosind ponderile w obţinte pentru setul de antrenare, se calculează valorile prezise de model pentru setul de testare şi eroarea pătratică medie pentru setul de date de testare.

Să se urmeze aceşti paşi pentru cuantificarea erorii. Puteţi explica de ce la rulări diferite se raportează rezultate diferite? cum se poate face ca testele să fie reproductibile (să dea de fiecare dată acelaşi rezultat)[1]?

Încercaţi diferite valori pentru parametrul rată de învăţare α, pentru un număr maxim de iteraţii fixat (de exemplu 100000). Există diferenţe de performanţă pentru valori diferite ale lui α? Care este valoarea cea mai bună pe care aţi găsit-o?

2 Setul de date

Setul de date este "Condition Based Maintenance of Naval Propulsion Plants" de la adresa:
http://archive.ics.uci.edu/ml/machine-learning-databases/00316/. Mai multe detalii şi o descriere a setului de date se găsesc la
http://archive.ics.uci.edu/ml/datasets/Condition+Based+Maintenance+of+Naval+Propulsi
Valoarea ce trebuie prezisă este cea din ultima coloană, `GT Turbine decay state coefficient` - pe care modelul o va prezice ca pe o valoare reală, iar trăsăturile de intrare sunt primele 16, de la `Lever position` la `Fuel flow`.

3 Precizări

- Se vor scrie funcţii care implementează paşii daţi mai sus. Urmariti scheletele de funcţii din notebook si completaţi codul în mod corespunzător.
- Din punctajul acordat, 1 punct este pentru scrierea vectorizată a codului, fără folosirea instrucţiunilor `while`, `for`, `if`, exceptând iterarea peste setul de date în rezolvarea punctului 3 şi afişarea valorilor prezise, conform ecuaţiei (8).
- Studenţii se pot consulta pentru rezolvarea temei, dar rezolvările vor fi individuale.

[1] Indicaţie: documentaţi–vă asupra semnificaţiei funcţiei numpy.random.seed.

- Prezentarea temei se va face la laboratorul din săptămâna 16—20 martie 2020. Pentru o întârziere de cel mult o săptămână se vor scădea 2 puncte din nota cuvenită. Temele predate cu o întârziere mai mare de o săptămână nu vor mai fi luate în considerare.