

Laborator 2

Lucian M. Sasu

5 martie 2019

1 Regresie liniară

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

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

atunci:

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

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

respectiv

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

- (b) fiecare element al lui \mathbf{X} se va împărți la diferența dintre maximul și minimul coloanei pe care se găsește:

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

Note:

- i. Coloanele constante nu se vor scala.

- ii. După aplicarea în ordine a celor două operații de mai sus pe o matrice de intrare \mathbf{X} , vom obține o matrice având componentele în intervalul $[0, 1]$, excepție fiind eventualele coloane constante.
- iii. Se recomandă implementarea folosind cod vectorizat, folosind numpy, vedeți [Prezentare NumPy](#).

Funcția pe care o scrieți va returna tripla: matricea cu elemente scalate și vectorii $(min_0, \dots, min_{n-1})$, $(max_1, \dots, max_{n-1})$, conținând respectiv minimele și maximele coloanelor.

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

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

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

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

3. Să se folosească metoda de minimizare folosind căutarea bazată pe gradient, pentru a determina model de predicție bazat pe regresie liniară. Se va folosi setul de date din secțiunea 2. Se va face în prealabil scalarea datelor, urmată de adăugarea unei coloane de valori 1, folosind funcțiile de la punctele precedente. Se va reprezenta grafic evoluția funcției de eroare J , pentru a permite eventuale ajustări manuale ale ratei de învățare α (vezi observațiile din curs).
4. Să se calculeze prin metoda pseudoinversei coeficienții regresiei liniare pentru setul de date din secțiunea 2.

Coeficienții funcției de regresie se calculează cu formula:

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

unde \mathbf{X} este matricea obținută după aplicarea funcției de la punctul 2. Pentru pseudoniversă se poate folosi funcția `numpy.linalg.pinv`. Funcția de regresie liniară este $h_{\theta} : \mathbf{R}^{n+1} \rightarrow \mathbf{R}$,

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

Pentru fiecare vector $\mathbf{x}^{(i)}$ din setul de instruire se va afișa valoarea prezisă de (8), împreună cu valoarea efectivă din setul de instruire. Se va calcula și afișa în final funcția totală de eroare:

$$J(\theta_0, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(\mathbf{x}^{(i)}) - y_i)^2 \quad (9)$$

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

5. De regulă, performanța unui model nu se măsoară pe datele de instruire, ci ne interesează puterea lui de generalizare, *i.e.* capacitatea de a face predicții pentru date similare cu setul de instruire, dar neutilizate în instruire. Una din metodele acceptate de măsurare a performanței este:

- se face o permutare aleatoare a setului de date, se poate folosi funcția `numpy.random.shuffle`.
- se împarte setul de la punctul precedent în două subseturi, primele 70% din date sunt folosite doar pentru antrenare, restul doar pentru testare;
- pe setul de antrenare se construiește model de regresie;
- se calculează eroarea J pentru modelul rezultat peste setul de date de testare.

Să se urmeze acești pași pentru cuantificarea erorii. Puteți explica de ce la rulări diferite se raportează rezultate diferite? cum se poate face ca testele să fie reproductibile (să dea de fiecare dată același rezultat)?¹

2 Setul de date

Setul de date este “Wine quality” de la adresa:

<http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv>.

¹Indicație: documentați-vă asupra semnificației funcției `numpy.random.seed`.

O descriere a setului de date se găsește la

<http://archive.ics.uci.edu/ml/datasets/Wine+Quality>.

Valoarea ce trebuie prezisă este **quality** - pe care modelul o va prezice ca pe o valoare reală, iar trăsăturile de intrare sunt celelalte 11.

3 Precizări

- Se vor scrie funcții care implementează pașii dați mai sus.
- Din punctajul acordat, 1 punct este pentru scrierea vectorizată a codului, fără folosirea instrucțiunilor **while**, **for**, **if**, exceptând iterarea peste setul de date în rezolvarea punctului 3 și afișarea valorilor prezise, conform ecuației (8). Valoarea funcției de eroare J din ec. (9) se va calcula fără a folosi instrucțiuni de ciclare.
- Studenții se pot consulta pentru rezolvarea temei, dar rezolvările vor fi individuale.
- Prezentarea temei se va face la laboratorul din săptămâna 18—22 martie 2019. 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.