Aplicații ale rețelelor neuro-fuzzy în estimarea costului dezvoltării software

Student: Madar Nicușor-Florin

Coordonator științific: Lect. dr. Șuter Florentina

Facultatea de Matematică și Informatică, Universitatea din București

16 septembrie 2017

Cuprins

Introducere

- Estimarea costului de dezvoltare al unui proiect software este o necesitate

Introducere

- Estimarea costului de dezvoltare al unui proiect software este o necesitate
- Estimările decid dacă un proiect primește finanțare, autorizație, echipa de dezvoltare, etc

Introducere

- Estimarea costului de dezvoltare al unui proiect software este o necesitate
- Estimările decid dacă un proiect primește finanțare, autorizație, echipa de dezvoltare, etc
- Apare necesitatea metodelor de a obține estimări cât mai apropiate de realitate

Introducere

- Folosim modelul COCOMO pentru a obține factorii de cost

Introducere

- Folosim modelul COCOMO pentru a obține factorii de cost
- Folosind date ale unor proiecte istorice, antrenăm o rețea neuro-fuzzy

Introducere

- Folosim modelul COCOMO pentru a obține factorii de cost
- Folosind date ale unor proiecte istorice, antrenăm o rețea neuro-fuzzy
- Analizăm rezultatele obținute peste datele de test

Cuprins

Modelul COCOMO

- Dezvoltat în 1981 de către Barry W. Boehm [1]

Modelul COCOMO

- Dezvoltat în 1981 de către Barry W. Boehm [1]
- Datele pentru care formulele sale de calcul au fost extrase din 63 de proiecte

Modelul COCOMO

- Dezvoltat în 1981 de către Barry W. Boehm [1]
- Datele pentru care formulele sale de calcul au fost extrase din 63 de proiecte
- Estimează costul de dezvoltare în funcție de 15 multiplicatori de efort și dimensiunea estimată a proiectului în mii de linii de cod

Modelul COCOMO

Caracteristică	Very Low	Low	Nominal	High	Very High	Extra High
Fiabilitatea softului necesară	0.75	0.88	1.00	1.15	1.40	
Dimensiunea bazei de date		0.94	1.00	1.08	1.16	
Complexitatea produslui	0.70	0.85	1.00	1.15	1.30	1.65
Constrângeri de performanță la rulare			1.00	1.11	1.30	1.66
Constrâgeri de memorie			1.00	1.06	1.21	1.56
Volatilitatea mediului de mașini virtuale		0.87	1.00	1.15	1.30	
Timpul necesar pentru schimbări		0.87	1.00	1.07	1.15	
Capabilitatea analiștilor	1.46	1.19	1.00	0.86	0.71	
Experiență în aplicații	1.29	1.13	1.00	0.91	0.82	
Capabilitatea inginerilor soft	1.42	1.17	1.00	0.86	0.70	
Experiență cu mașinile virtuale	1.21	1.10	1.00	0.90		
Experiență cu limbajul de programare	1.14	1.07	1.00	0.95		
Aplicarea metodelor de inginerie soft	1.24	1.10	1.00	0.91	0.82	
Folosirea uneltelor software	1.24	1.10	1.00	0.91	0.83	
Strictețea planificârii	1.23	1.08	1.00	1.04	1.10	

Cuprins

Logică fuzzy

- Mulțime fuzzy – apartenența unui element la o mulțime nu mai este stabilită de către o funcție binară

Logică fuzzy

- Mulțime fuzzy – apartenența unui element la o mulțime nu mai este stabilită de către o funcție binară
- Apare noțiunea de funcție de apartenență

Logică fuzzy

- Mulțime fuzzy – apartenența unui element la o mulțime nu mai este stabilită de către o funcție binară
- Apare noțiunea de funcție de apartenență

Definiție

$$\mu_A : X \to [0, 1] \tag{1}$$

Logică fuzzy

- Sistem fuzzy – sistem ce conține una sau mai multe variabile care primește valori peste stări care sunt mulțimi fuzzy

Logică fuzzy

- Sistem fuzzy – sistem ce conține una sau mai multe variabile care primește valori peste stări care sunt mulțimi fuzzy
- Pentru fiecare variabilă mulțimile fuzzy sunt definite peste o mulțime universală

Logică fuzzy

- Sistem fuzzy – sistem ce conține una sau mai multe variabile care primește valori peste stări care sunt mulțimi fuzzy
- Pentru fiecare variabilă mulțimile fuzzy sunt definite peste o mulțime universală
- Își gasesc utilitatea în a modela variabile din lumea reală, unde orice măsurătoare are precizie finită

Logică fuzzy

- Sistem fuzzy – sistem ce conține una sau mai multe variabile care primește valori peste stări care sunt mulțimi fuzzy
- Pentru fiecare variabilă mulțimile fuzzy sunt definite peste o mulțime universală
- Își gasesc utilitatea în a modela variabile din lumea reală, unde orice măsurătoare are precizie finită

Exemplu

Presupunem o regulă pentru un termostat:
IF (temperatura este "rece") THEN (încălzirea este "mare")

Rețele neuronale

- Sisteme conecționiste de calcul inspirate din modelul biologic al rețelelor neuronale

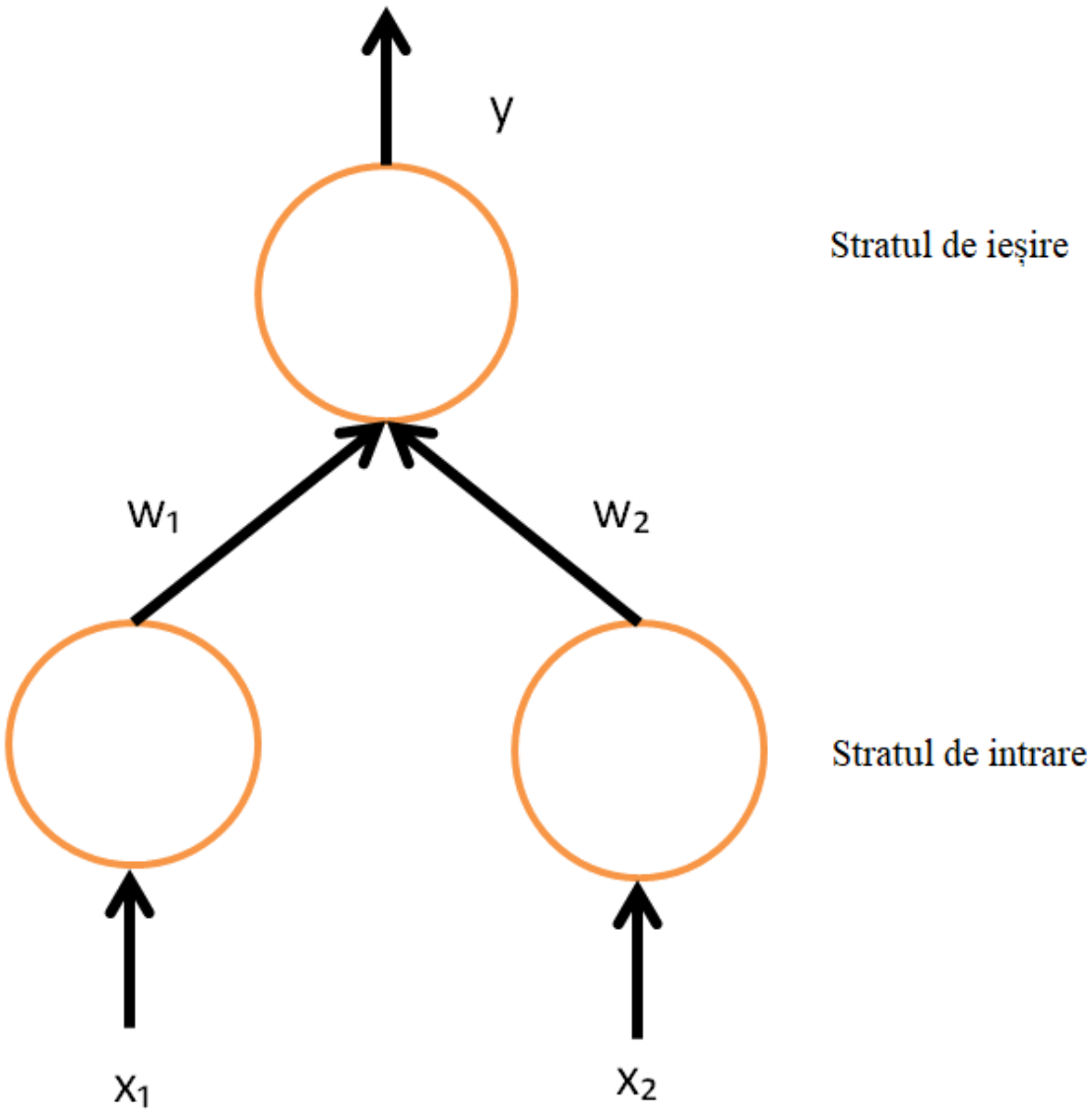
Rețele neuronale

- Sisteme conecționiste de calcul inspirate din modelul biologic al rețelelor neuronale
- Sunt alcătuite din grupări de perceptroni care transmit succesiv semnale între ele

Rețele neuronale

- Sisteme conecționiste de calcul inspirate din modelul biologic al rețelelor neuronale
- Sunt alcătuite din grupări de perceptroni care transmit succesiv semnale între ele
- Învățarea se bazează pe alegerea unei funcții de cost și propagarea înapoi a erorilor astfel încât fiecare perceptron să își actualizeze ponderile

Rețele neuronale



Cuprins

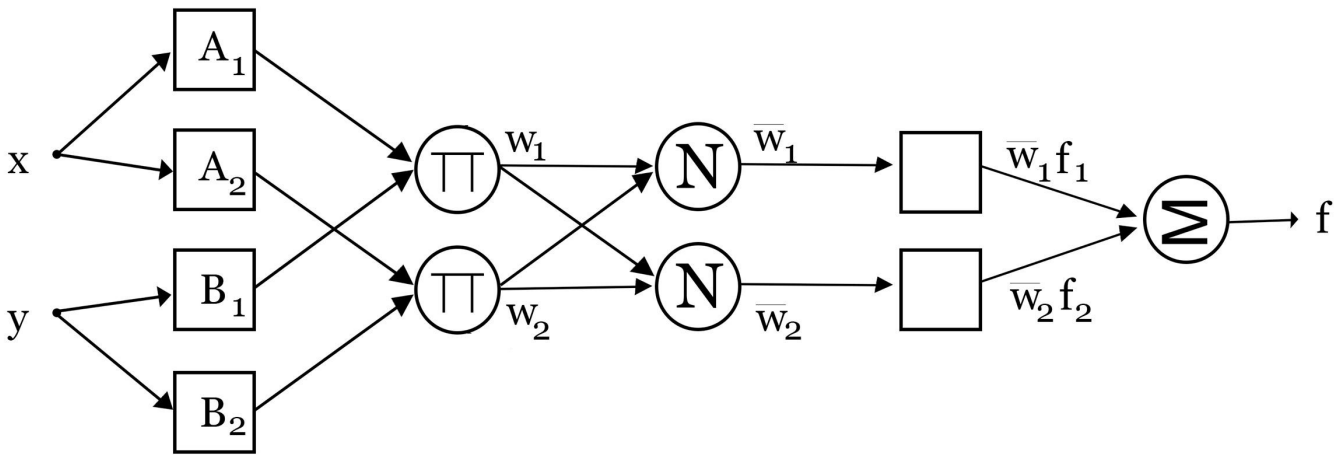
Rețele ANFIS

- Sunt un caz particular de rețele adaptive [2]

Rețele ANFIS

- Sunt un caz particular de rețele adaptive [2]
- Sunt construite peste un sistem de inferență fuzzy, unde regulile capătă puteri de "aprindere" (ponderi), astfel încât să aibă loc un proces de învățare

Rețele ANFIS



Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then

Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then
- Primul strat transformă variabilele de intrare x și y în corespondenții lor fuzzy.

Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then
- Primul strat transformă variabilele de intrare x și y în corespondenții lor fuzzy.
- Cel de-al doilea strat înmulțește semnalele primite de la primul strat și trimite rezultatul mai departe

Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then
- Primul strat transformă variabilele de intrare x și y în corespondenții lor fuzzy.
- Cel de-al doilea strat înmulțește semnalele primite de la primul strat și trimite rezultatul mai departe
- Al treilea strat calculează puterea de "aprindere" normalizată pentru fiecare intrare

Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then
- Primul strat transformă variabilele de intrare x și y în corespondenții lor fuzzy.
- Cel de-al doilea strat înmulțește semnalele primite de la primul strat și trimite rezultatul mai departe
- Al treilea strat calculează puterea de "aprindere" normalizată pentru fiecare intrare
- Stratul 4 calculează pentru fiecare putere de "aprindere" din stratul anterior rezultatul aplicării funcției, i.e.

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{2}$$

unde $\bar{w}_i$ este ieșirea stratului 3, p_i, q_i, r_i este mulțimea parametrilor

Rețele ANFIS

- Figura anterioară exemplifică o rețea care modelează doua reguli fuzzy if-then
- Primul strat transformă variabilele de intrare x și y în corespondenții lor fuzzy.
- Cel de-al doilea strat înmulțește semnalele primite de la primul strat și trimite rezultatul mai departe
- Al treilea strat calculează puterea de "aprindere" normalizată pentru fiecare intrare
- Stratul 4 calculează pentru fiecare putere de "aprindere" din stratul anterior rezultatul aplicării funcției, i.e.

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{2}$$

unde $\bar{w}_i$ este ieșirea stratului 3, p_i, q_i, r_i este mulțimea parametrilor

- Stratul 5 însumează toate semnalele pe care le primește și oferă rezultatul

Cuprins

Rezultate

- Împărțind datele în 85% date de antrenare și 15% date de test am obținut

Rezultate

- Împărțind datele în 85% date de antrenare și 15% date de test am obținut
- Eroare RMSE de 0.059

Rezultate

- Împărțind datele în 85% date de antrenare și 15% date de test am obținut
- Eroare RMSE de 0.059
- Eroare MMRE de 0.036

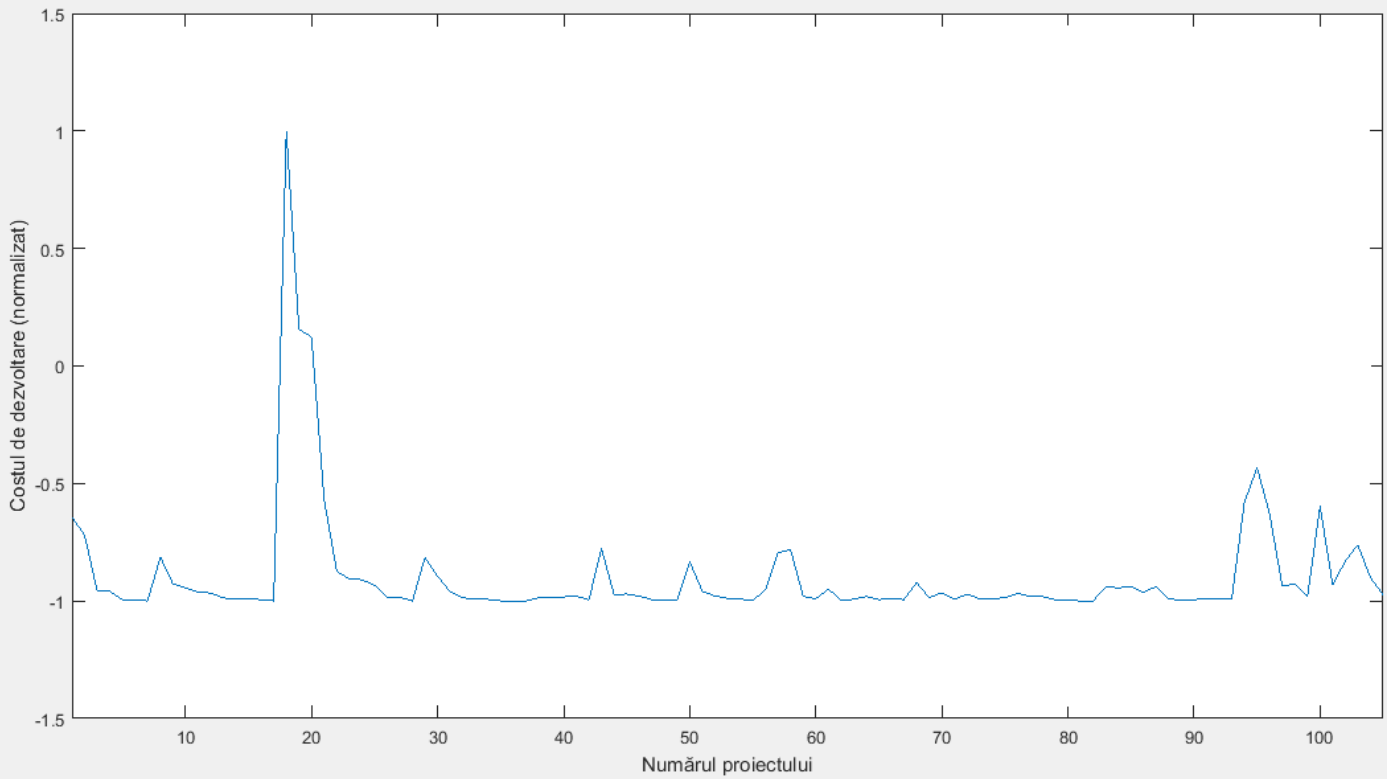
Rezultate

- Împărțind datele în 85% date de antrenare și 15% date de test am obținut
- Eroare RMSE de 0.059
- Eroare MMRE de 0.036
- Estimare PRED(25) de 1

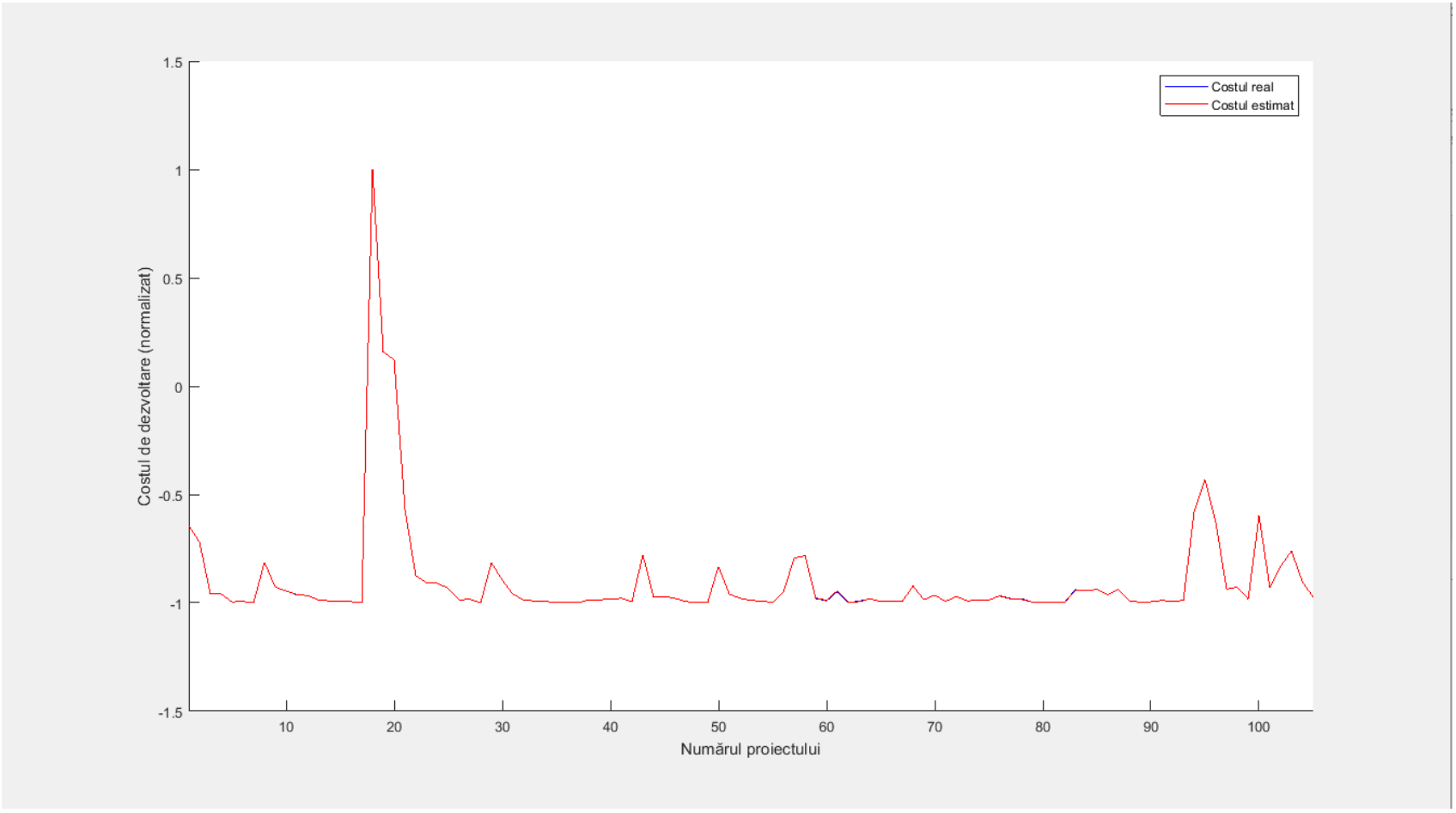
Rezultate

- Împărțind datele în 85% date de antrenare și 15% date de test am obținut
- Eroare RMSE de 0.059
- Eroare MMRE de 0.036
- Estimare PRED(25) de 1

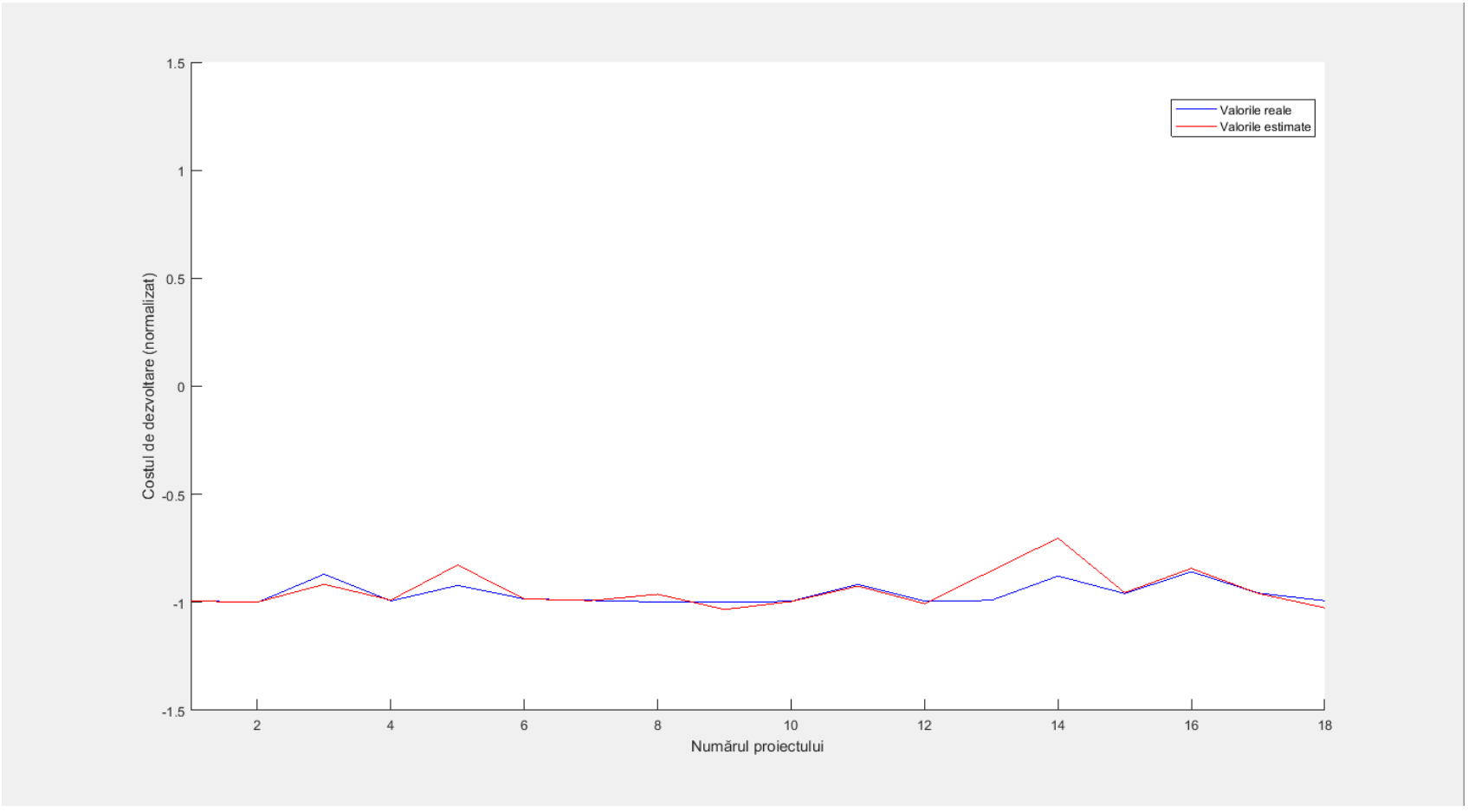
Rezultate


1.5
1
0.5
0
-0.5
-1
-1.5
Costul de dezvoltare (normalizat)
10
20
30
40
50
60
70
80
90
100
Numărul proiectului

Rezultate (antrenare)



Rezultate (test)



Rezultate

- În cazul rețelelor neuronale clasice, folosind aceleași date, rezultatele au fost

Rezultate

- În cazul rețelelor neuronale clasice, folosind aceleași date, rezultatele au fost
- Eroare RMSE de 0.227

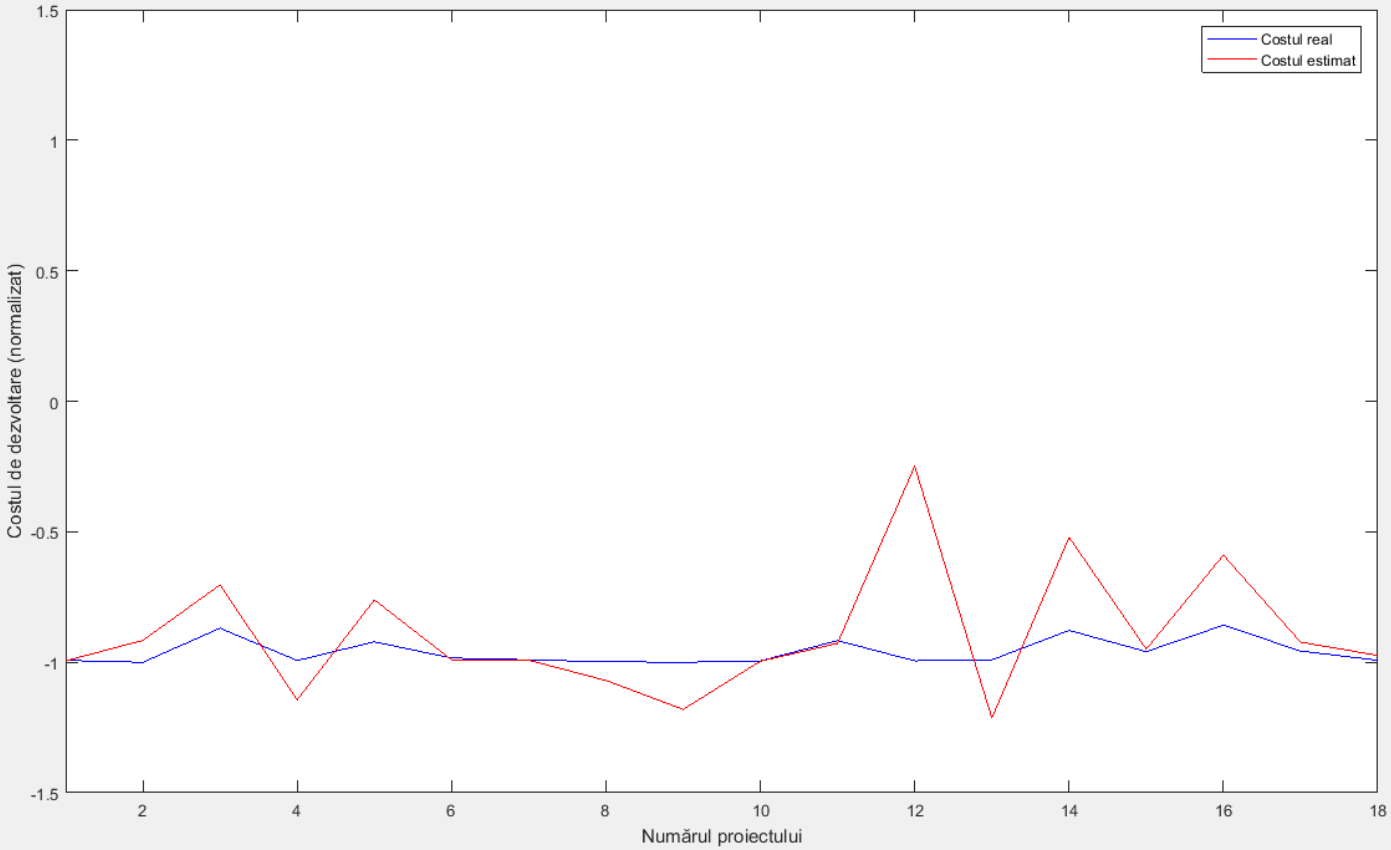
Rezultate

- În cazul rețelelor neuronale clasice, folosind aceleași date, rezultatele au fost
- Eroare RMSE de 0.227
- Eroare MMRE de 0.146

Rezultate

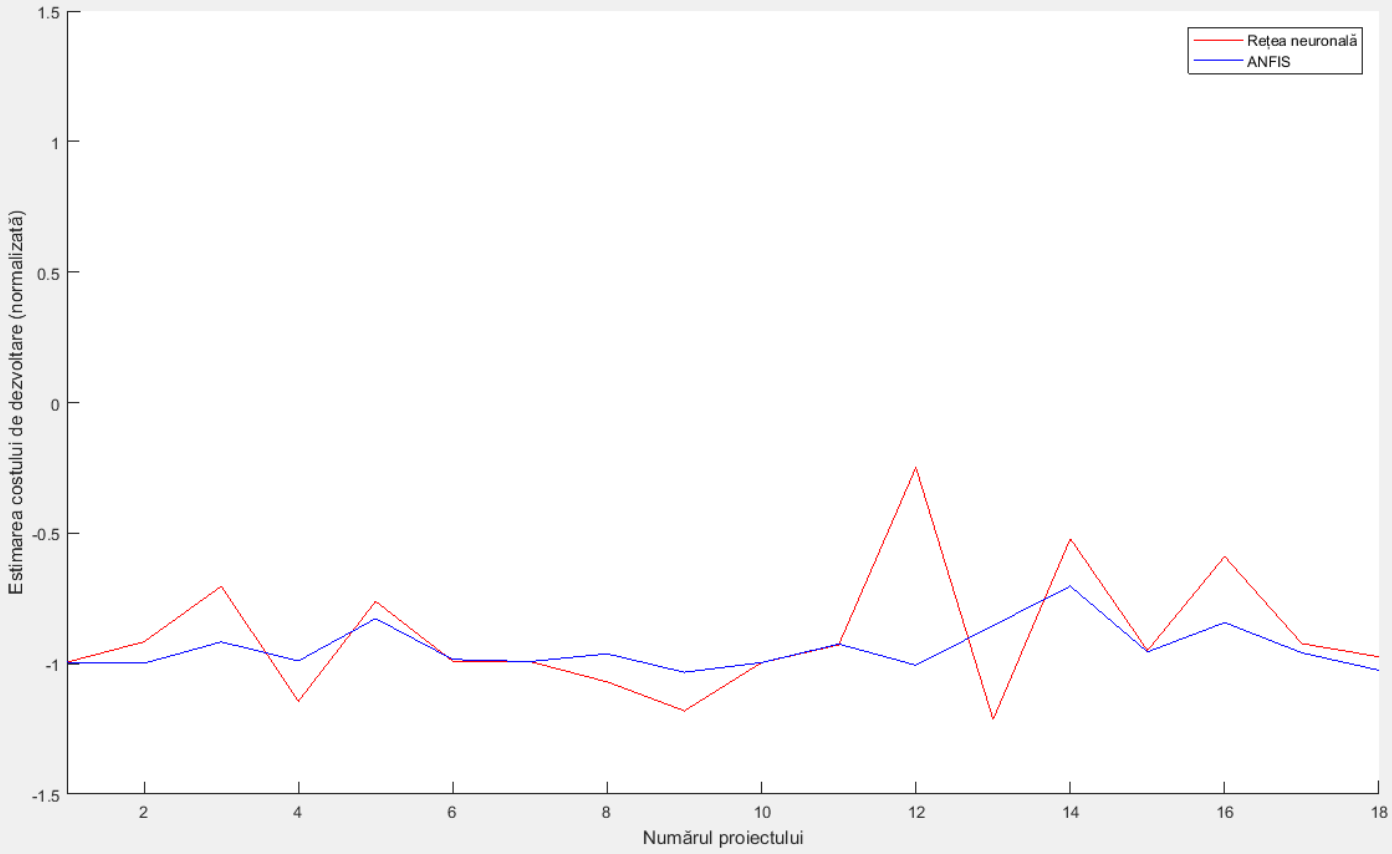
- În cazul rețelelor neuronale clasice, folosind aceleași date, rezultatele au fost
- Eroare RMSE de 0.227
- Eroare MMRE de 0.146
- Estimare PRED(25) de 0.83

Rezultate (test)



Cuprins

Concluzii (comparatie)



Concluzii

- Rețelele ANFIS obțin rezultate mult mai bune pentru acest set de date decât rețelele neuronale

Concluzii

- Rețelele ANFIS obțin rezultate mult mai bune pentru acest set de date decât rețelele neuronale
- PRED(25) de 1 vs PRED(25) de 0.83

Concluzii

- Rețelele ANFIS obțin rezultate mult mai bune pentru acest set de date decât rețelele neuronale
- PRED(25) de 1 vs PRED(25) de 0.83
- Considerăm că ANFIS este o soluție bună pentru estimările de cost

Concluzii

- Rețelele ANFIS obțin rezultate mult mai bune pentru acest set de date decât rețelele neuronale
- PRED(25) de 1 vs PRED(25) de 0.83
- Considerăm că ANFIS este o soluție bună pentru estimările de cost
- Rețelele ANFIS pot fi, în plus, ajustate și de cunoștinte expert

Cuprins

Bibliografie

[1] Boehm, Barry W. (1981). *Software Engineering Economics.* Prentice-Hall.

[2] Jang, J.-S.R. (1993) ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions On Systems, Man, And Cybernetics*, 23(3), pp. 665-685.