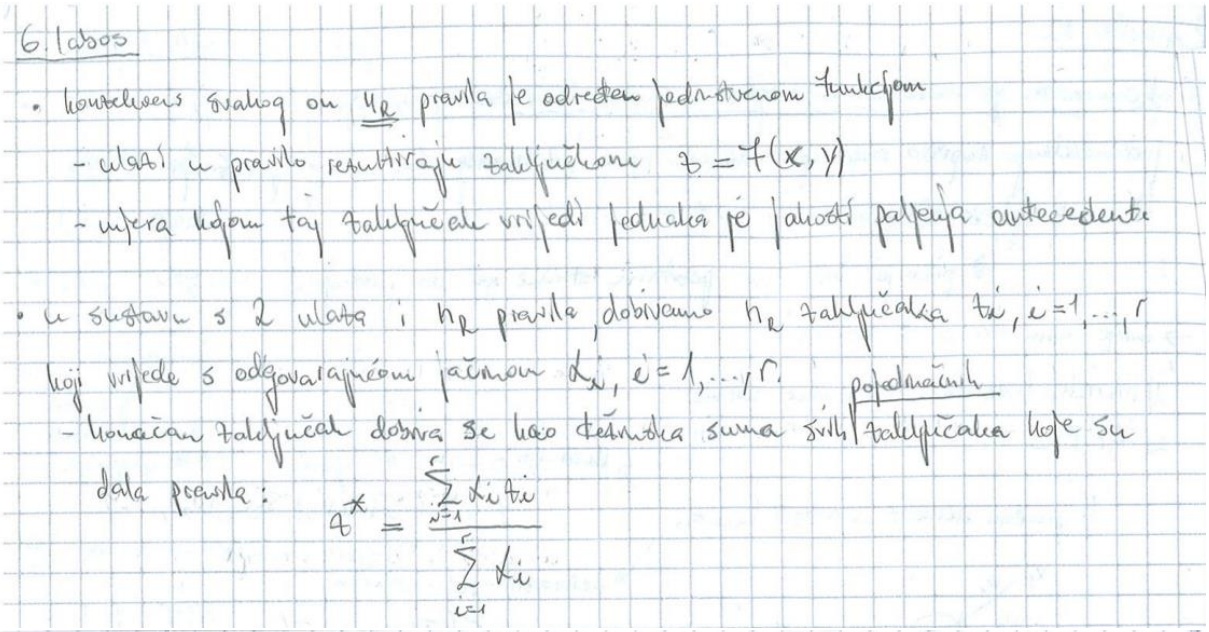
Neizrazito, evolucijsko i neuro-računarstvo

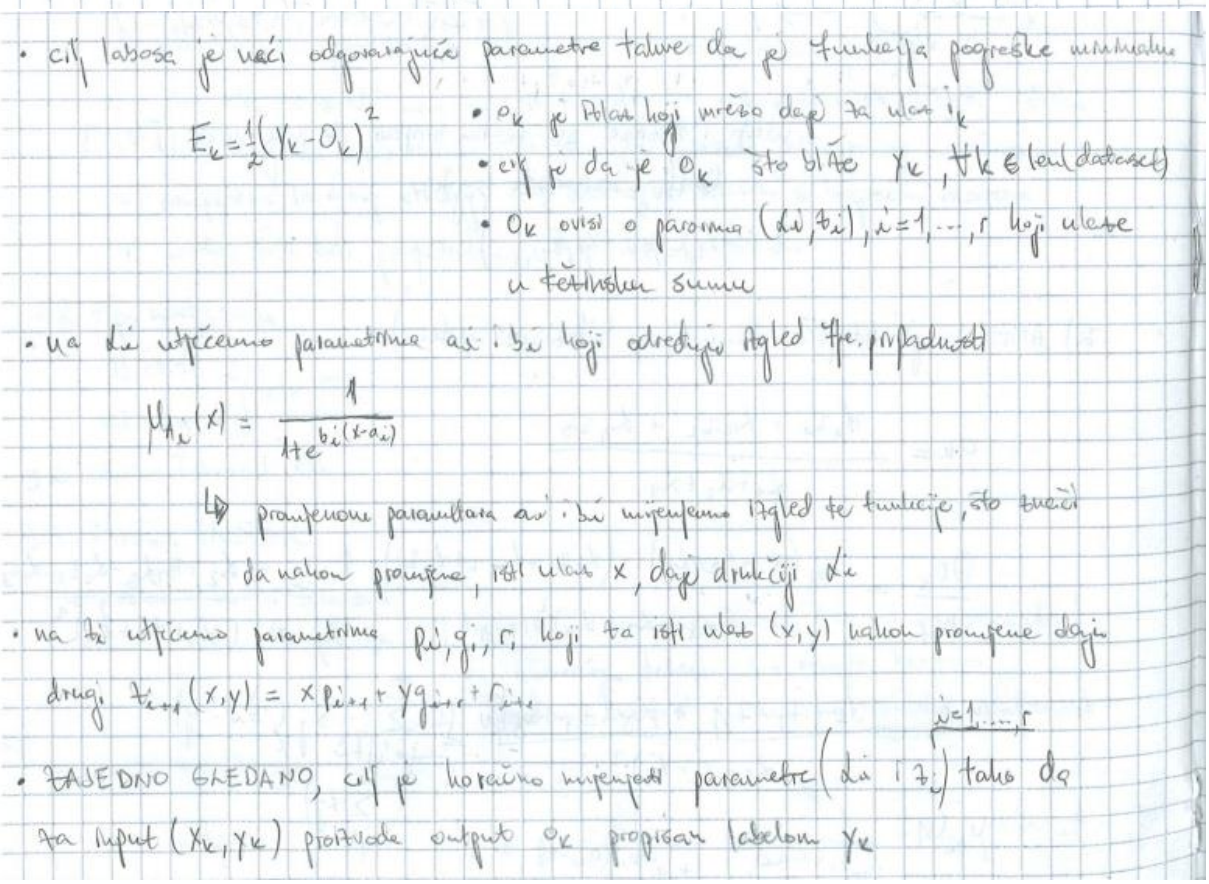
6. Domaća zadaća

1. zadatak



6. labos

- konačnom svakog on n_R pravila je određen jednostavnom funkcijom
  - ulazi u pravilo rezultiraju zaključkom $z = f(x, y)$
  - mjera kojom taj zaključak vrijedi jednaka je jakosti paljenja antecedenta
- u sustavu s 2 ulaza i n_R pravila, dobivamo n_R zaključaka z_i, $i = 1, \dots, r$ koji vrijede s odgovarajućom jačinom α_i, $i = 1, \dots, r$.
  - konačan zaključak dobiva se kao težinska suma svih pojedinačnih zaključaka koje su dala pravila: $z^* = \frac{\sum_{i=1}^{r} \alpha_i z_i}{\sum_{i=1}^{r} \alpha_i}$



- cilj labosa je naći odgovarajuće parametre tako da je funkcija pogreške minimalna

$$E_k = \frac{1}{2}(y_k - o_k)^2$$

  - o_k je izlaz koji mreža daje za ulaz i_k
  - cilj je da je o_k što bliže y_k, $\forall k \in$ (cijeli dataset)
  - o_k ovisi o parovima (α_i, z_i), $i = 1, \dots, r$ koji ulaze u težinsku sumu
- na što utječemo parametrima a_i i b_i koji određuju izgled μ_{A_i}, pripadnosti

$$\mu_{A_i}(x) = \frac{1}{1 + e^{b_i(x - a_i)}}$$

  ↳ promjenom parametara a_i i b_i mijenjamo izgled te funkcije, što znači da nakon promjene, isti ulaz x, daje drukčiji α_i
- na što utječemo parametrima p_i, q_i, r_i koji za isti ulaz (x, y) nakon promjene daju drugi, $z_{i+1}(x, y) = x p_{i+1} + y q_{i+1} + r_{i+1}$
- ZAJEDNO GLEDANO, cilj je koračno mijenjati parametre $(\alpha_i$ i $z_i)$, $i = 1, \dots, r$ tako da za input (x_k, y_k) proizvode output o_k propisan labelom y_k

R_1: Ako x je A_1 i y je B_1 tada $z_1 = p_1 x + q_1 y + r_1$

R_2: Ako x je A_2 i y je B_2 tada $z_2 = p_2 x + q_2 y + r_2$

⋮

R_r: Ako x je A_r i y je B_r tada $z_r = p_r x + q_r y + r_r$

x je ulaz u neiz. skupove $A_1, \dots, A_r$
y je ulaz u neiz. skupove $B_1, \dots, B_r$
} treba naučiti ukupno $2 \cdot r$ nepoznatih skupova odnosno $2 \cdot (2 \cdot r)$ parametara koji ih određuju + $3 \cdot r$ parametara koji određuju r funkcija u konsekvens-dijelovima pravila

Kako?

↳ pojednostavljenje: kako predstaviti jedno pravilo? ✓✓
kako ažurirati parametre?

$$E_k = \frac{1}{2}(y_k - o_k)^2$$

$$\boxed{\frac{\partial E_k}{\partial r_i}} = \frac{\partial E_k}{\partial o_k} \cdot \frac{\partial o_k}{\partial z_k} \cdot \frac{\partial z_k}{\partial r_i} = -(y_k - o_k) \cdot \frac{\alpha_k}{\sum_{j=1}^{r} \alpha_j} \cdot 1$$

(1) (2) (3)

1) $\frac{\partial E_k}{\partial o_k} = \frac{\partial}{\partial o_k}\left(\frac{1}{2}(y_k - o_k)^2\right) = -(y_k - o_k)$

2) $\frac{\partial o_k}{\partial z_k} = \frac{\partial}{\partial z_k}\left(\frac{\sum_{i=1}^{r} \alpha_i z_i}{\sum_{j=1}^{r} \alpha_j}\right) = \frac{\alpha_k}{\sum_{j=1}^{r} \alpha_j}$

3) $\frac{\partial z_k}{\partial r_i} = \frac{\partial}{\partial r_i}\left(p_i x + q_i y + r_i\right) = 1$

$$\boxed{\frac{\partial E_k}{\partial p_i}} = 1) \cdot 2) \cdot \frac{\partial z_k}{\partial p_i} = \frac{\partial}{\partial p_i}\left(p_i x + q_i y + r_i\right) = -(y_k - o_k) \cdot \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j} \cdot x$$

$$\boxed{\frac{\partial E_k}{\partial q_i}} = 1) \cdot 2) \cdot \frac{\partial z_k}{\partial q_i} = \frac{\partial}{\partial q_i}\left(p_i x + q_i y + r_i\right) = -(y_k - o_k) \cdot \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j} \cdot y$$

$$\frac{\partial E_k}{\partial a_i} = \frac{\partial E_k}{\partial o_k} \cdot \frac{\partial o_k}{\partial \alpha_i} \cdot \frac{\partial \alpha_i}{\partial a_i} = -(y_k - o_k) \cdot \frac{\sum_{j=1, j\neq i}^{r} \alpha_j (z_{ki} - z_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2} \cdot b_i \alpha_i (1-\alpha_i)$$

1) 2) 3)

1) $\frac{\partial E_k}{\partial o_k} = -(y_k - o_k)$ 2) $\frac{\partial o_k}{\partial \alpha_i} = \frac{\sum_{j=1, j\neq i}^{r} \alpha_j (z_{ki} - z_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2}$ 3) $\frac{\partial \alpha_i}{\partial a_i} = b_i \alpha_i (1-\alpha_i)$

$$\frac{\partial E_k}{\partial b_i} = 1) \cdot 2) \frac{\partial \alpha_i}{\partial b_i} = 1) \cdot 2) \cdot -(x - a_i) \alpha_i (1-\alpha_i)$$

$$= (y_k - o_k) \frac{\sum_{j=1, j\neq i}^{r} \alpha_j (z_{ki} - z_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2} \cdot (x - a_i) \alpha_i (1-\alpha_i)$$

Batch

$$E = \frac{1}{2N} \sum_{k=1}^{N} (y_k - o_k)^2$$

$$\left\{ \psi(t+1) = \psi(t) - \eta \frac{\partial E}{\partial \psi} \right\} \rightarrow \text{OPĆENITO VRIJEDI}$$

$$\frac{\partial E}{\partial r_i} = \frac{\partial}{\partial r_i} \left(\frac{1}{2N} \cdot \sum_{k=1}^{N} (y_k - o_k)^2 \right) = \frac{1}{2N} \cdot \sum_{k=1}^{N} 2(y_k - o_k) \cdot (-1) \cdot \frac{\partial o_k}{\partial r_i} =$$

$$= -\frac{1}{N} \sum_{k=1}^{N} (y_k - o_k) \cdot \frac{\partial o_k}{\partial r_i} = -\frac{1}{N} \sum_{k=1}^{N} (y_k - o_k) \cdot \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j}$$

1)

$$\frac{\partial o_k}{\partial r_i} = \frac{\partial o_k}{\partial z_{ki}} \cdot \frac{\partial z_{ki}}{\partial r_i} = \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j} \cdot 1$$

$$\frac{\partial E}{\partial r_i} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial z_{ki}} \cdot \frac{\partial z_{ki}}{\partial r_i}$$

$$\frac{\partial E}{\partial p_i} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial z_{ki}} \cdot \frac{\partial z_{ki}}{\partial p_i} = -\frac{1}{N} \sum_{k=1}^{N} (y_k - o_k) \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j} \cdot x_k$$

$$\frac{\partial E}{\partial q_i} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial \alpha_i} \cdot \frac{\partial \alpha_i}{\partial q_i} = -\frac{1}{N} \sum_{k=1}^{N} (y_k - o_k) \frac{\alpha_i}{\sum_{j=1}^{r} \alpha_j} \cdot y_k$$

$$\frac{\partial E}{\partial a_i} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial \alpha_i} \cdot \frac{\partial \alpha_i}{\partial a_i} = -\frac{1}{N} \sum_{k=1}^{N} (y_k - o_k) \cdot \frac{\sum_{j=1, j \neq i}^{r} \alpha_j (x_k - b_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2} \cdot b_i \alpha_i (1 - \alpha_i)$$

1) 2) 3)

1) $\Rightarrow \frac{\partial E}{\partial o_k} = \frac{1}{2N} \cdot \sum_{k=1}^{N} 2 (y_k - o_k) \cdot (-1) = -\frac{1}{N} \cdot \left(\sum_{k=1}^{N} (y_k - o_k) \right)$

2) $\frac{\partial o_k}{\partial \alpha_i} = \frac{\sum_{j=1, j \neq i}^{r} \alpha_j (x_k - b_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2}$ 3) $\frac{\partial \alpha_i}{\partial b_i} = b_i \alpha_i (1 - \alpha_i)$

$$\frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial \alpha_i} \cdot \frac{\partial \alpha_i}{\partial b_i} = 1) \cdot 2) \cdot 3) = +\frac{1}{N} \sum_{k=1}^{N} \frac{\sum_{j=1, j \neq i}^{r} \alpha_j (x_k - b_j)}{\left(\sum_{j=1}^{r} \alpha_j\right)^2} \cdot (y_k - o_k) (x_k - a_i) \alpha_i (1 - \alpha_i)$$

3) $\Rightarrow \frac{\partial \alpha_i}{\partial b_i} = -(x - a_i) \alpha_i (1 - \alpha_i)$

$$\frac{\partial \alpha_i}{\partial a_{i,alpha}} = \frac{\partial}{\partial a_{i,alpha}} \left(alpha(x) \cdot beta(x) \right) = beta(x) \left(alpha(x) \cdot (1 - alpha(x)) \cdot (+b_{i,alpha}) \right)$$

$$= +b_{i,alpha} \cdot beta(x) \cdot alpha(x) \cdot (1 - alpha(x))$$

$$alpha = \sigma\left(b_{i,alpha} \left(a_{i,alpha} - x \right) \right)$$

$$\frac{\partial alpha}{\partial a_i} = \sigma\left(b_{i,alpha} \left(a_{i,alpha} - x \right) \right) \cdot \left(1 - \sigma\left(b_{i,alpha} \left(a_{i,alpha} - x \right) \right) \right) \left(+b_{i,alpha} \right)$$

$$beta = \sigma\left(b_{i,beta} \left(a_{i,beta} - x \right) \right)$$

$$\frac{\partial beta}{\partial a_{i,alpha}} = 0 \Rightarrow const!$$

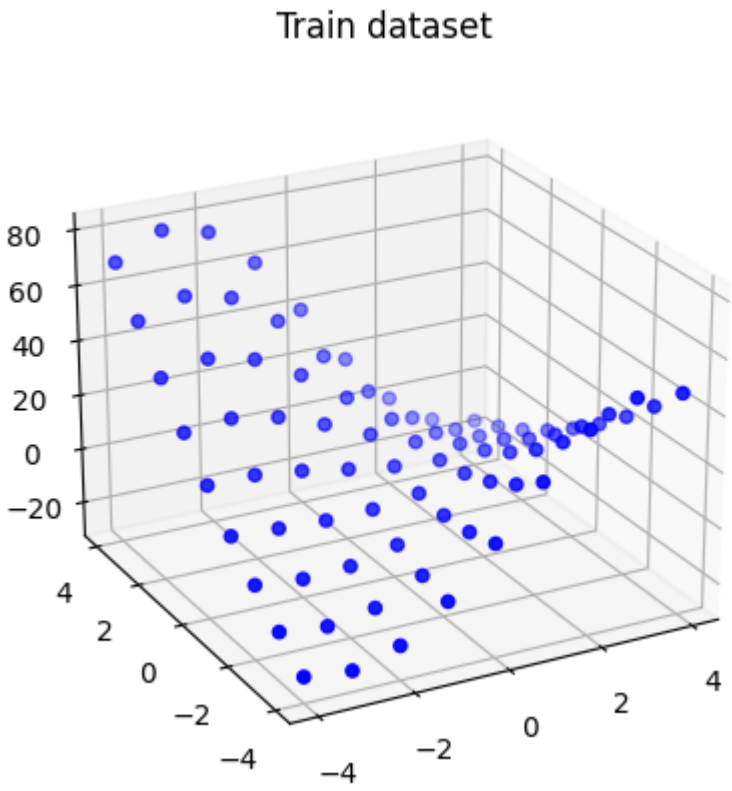
$$\frac{\partial \alpha_i}{\partial b_{i,alpha}} = \frac{\partial}{\partial b_{i,alpha}} \left(alpha(x) \cdot beta(x) \right) = beta(x) \cdot \left(alpha(x) \left(1 - alpha(x) \right) \left(a_{i,alpha} - x \right) \right)$$

$$\frac{\partial alpha}{\partial b_{i,alpha}} = alpha (1 - alpha) \cdot \left(a_{i,alpha} - x \right)$$

$$\frac{\partial beta}{\partial b_{i,alpha}} = 0 \Rightarrow const!$$

2. Za detalje implementacije zaviriti u kod.

3.



4.

a) samo jedno pravilo:

Ovako izgrađen ANFIS sustav prejednostavan je da bi naučio iole složenije preslikavanje. Ono što uočavamo kad pokrenemo program je da pogreška opada relativno sporo u odnosu na situacije kad imamo veći broj pravila.

```
C:\Users\PP\AppData\Local\Programs\Python\Python39\python.exe C:/CodeRepository/Python/PycharmProjects/NENR/zadó.py
Unesite broj pravila: 1
epoch = 1 error = 371.41164595394974
epoch = 2 error = 334.48033358202906
epoch = 3 error = 313.30505844386164
epoch = 4 error = 298.7639836021766
epoch = 5 error = 287.58611048262765
epoch = 6 error = 278.50017482687525
epoch = 7 error = 270.93363389729274
epoch = 8 error = 264.5708225152855
epoch = 9 error = 259.20095050887585
epoch = 10 error = 254.6641784380394
epoch = 11 error = 250.8311299649657
epoch = 12 error = 247.59401132771944
epoch = 13 error = 244.8619341700259
epoch = 14 error = 242.55790421784403
epoch = 15 error = 240.6165839834131
epoch = 16 error = 238.98249760193048
epoch = 17 error = 237.6085392128879
epoch = 18 error = 236.45471537536346
epoch = 19 error = 235.4870786336308
epoch = 20 error = 234.67682117709742
epoch = 21 error = 233.99950396144266
epoch = 22 error = 233.43440090455695
epoch = 23 error = 232.9639409814366
epoch = 24 error = 232.57323364446864
epoch = 25 error = 232.24966517087563
```

Važno je uočiti i da kako epohe odmiču pogreška počinje rasti što je dokaz prevelike jednostavnosti sustava.

```
epoch = 2941 error = 231.12586591020002
epoch = 2942 error = 231.12586592851122
epoch = 2943 error = 231.12586594680863
epoch = 2944 error = 231.12586596509232
epoch = 2945 error = 231.12586598336233
epoch = 2946 error = 231.12586600161862
epoch = 2947 error = 231.12586601986115
epoch = 2948 error = 231.12586603809
epoch = 2949 error = 231.12586605630514
epoch = 2950 error = 231.12586607450677
epoch = 2951 error = 231.12586609269465
epoch = 2952 error = 231.12586611086894
epoch = 2953 error = 231.12586612902956
epoch = 2954 error = 231.12586614717674
epoch = 2955 error = 231.12586616531013
epoch = 2956 error = 231.12586618343016

Process finished with exit code -1
```

b) dva pravila:

Ovakav ANFIS je također prejednostavan, iako se ponaša bolje od sustava sa samo jednim pravilom. Kod njega uočavamo brže smanjenje pogreške, no nakon određenog broja epoha pogreška također počinje rasti.

```
C:\Users\PP\AppData\Local\Programs\Python\Python39\python.exe C:/CodeRepository/Python/PycharmProjects/NENR/zad6.py
Unesite broj pravila: 2
epoch = 1 error = 370.3575790055284
epoch = 2 error = 261.245942476195
epoch = 3 error = 168.79914541397167
epoch = 4 error = 108.06413882775854
epoch = 5 error = 71.38654147894108
epoch = 6 error = 49.30887783583404
epoch = 7 error = 35.9802345281368
epoch = 8 error = 27.863207039761733
epoch = 9 error = 22.775586142676094
epoch = 10 error = 19.425237499709322
epoch = 11 error = 17.064806885333844
epoch = 12 error = 15.241256456232069
epoch = 13 error = 13.840770281317859
epoch = 14 error = 12.815243550166405
epoch = 15 error = 11.963850672522543
epoch = 16 error = 11.213687711389062
epoch = 17 error = 10.569972070232414
epoch = 18 error = 10.038062972255151
epoch = 19 error = 9.614979927559888
epoch = 20 error = 9.289848781636579
epoch = 21 error = 9.048404670496696
epoch = 22 error = 8.875998267254792
epoch = 23 error = 8.760852129506022
epoch = 24 error = 8.695560912158331
```

```
epoch = 1561 error = 3.7993883031147964
epoch = 1562 error = 3.7994730411379107
epoch = 1563 error = 3.7995577889348926
epoch = 1564 error = 3.799642546272421
epoch = 1565 error = 3.799727312917667
epoch = 1566 error = 3.7998120886382853
epoch = 1567 error = 3.7998968732024663
epoch = 1568 error = 3.7999816663788453
epoch = 1569 error = 3.800066467936624
epoch = 1570 error = 3.800151277645416
epoch = 1571 error = 3.8002360952754075
epoch = 1572 error = 3.8003209205972563
epoch = 1573 error = 3.8004057533821145
epoch = 1574 error = 3.8004905934016326
epoch = 1575 error = 3.8005754404279775
epoch = 1576 error = 3.800660294233793
epoch = 1577 error = 3.8007451545922364
epoch = 1578 error = 3.8008300212769703
epoch = 1579 error = 3.80091489406216
epoch = 1580 error = 3.8009997727224416
epoch = 1581 error = 3.8010846657032994
epoch = 1582 error = 3.8011695467694646
```

c) prikladan broj pravila:

Eksperimentiranjem i bilježenjem pogreške nakon 1000 epoha treniranja zaključujem da je prikladan broj pravila upravo 20. Iako je proces učenja stohastički te bi postupak učenja trebalo provesti više puta pa usporediti srednje vrijednosti konačnih pogrešaka, radi uštede vremena odlučujem se za ovaj heuristički pristup koji mi opravdava logičku pretpostavku da bi takav sustav morao biti dovoljno složen.

Uz to prilažem slike pogrešaka sustava sa varijabilnim brojem pravila nakon 1000 epoha učenja kao nekakav dokaz ispravnosti moje pretpostavke.

Broj pravila = 5:

```
epoch = 1000 error = 0.4169607421492397
```

Broj pravila = 10:

```
epoch = 1000 error = 0.2780248340089124
```

Broj pravila = 15:

```
epoch = 1000 error = 0.26289871285014543
```

Broj pravila = 20:

```
epoch = 1000 error = 0.18560391421375183
```

Broj pravila = 25:

```
epoch = 1000 error = 0.48580109077402495
```

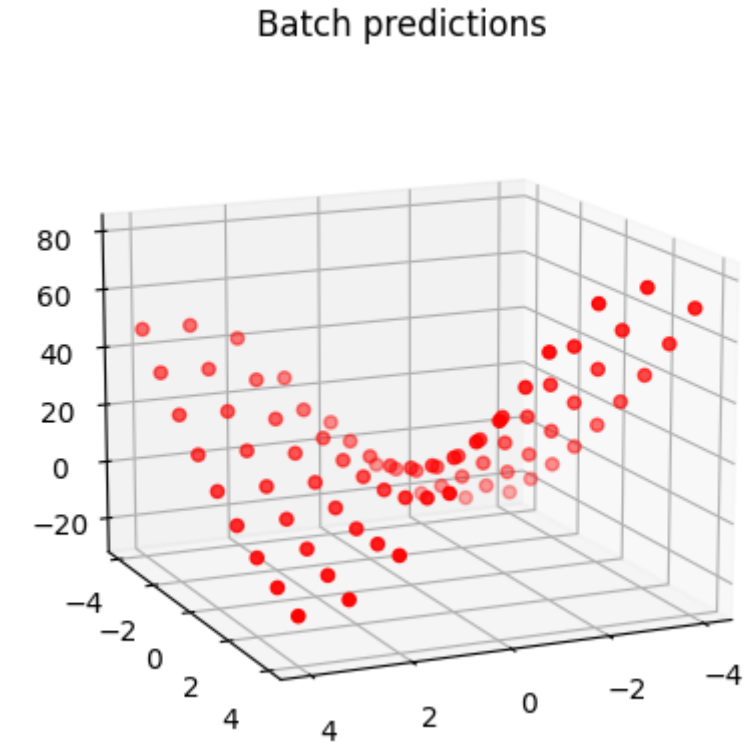
Broj pravila = 30:

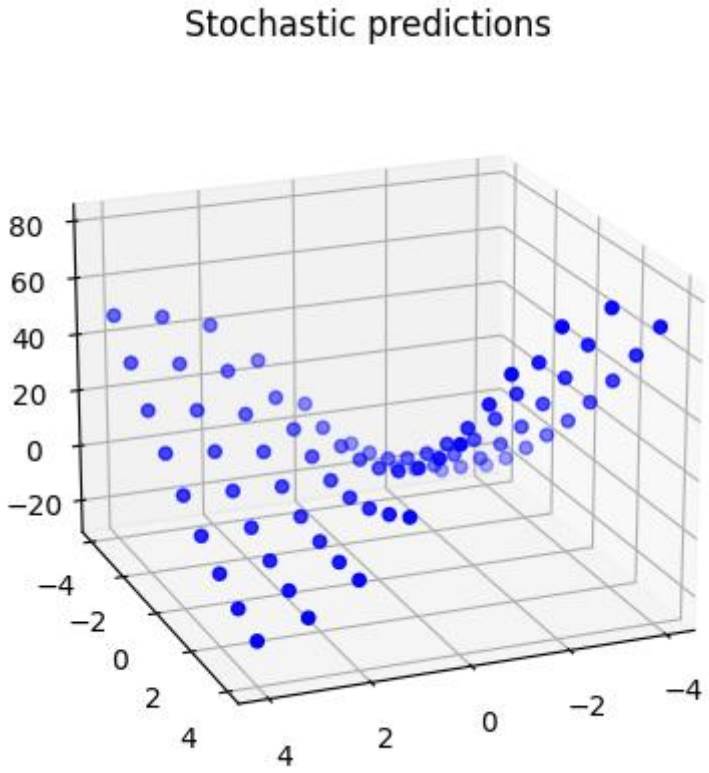
```
epoch = 1000 error = 0.3233332353325057
```

Sada kad smo odredili prikladan broj pravila vidimo da pogreška opada i relativno brzo.

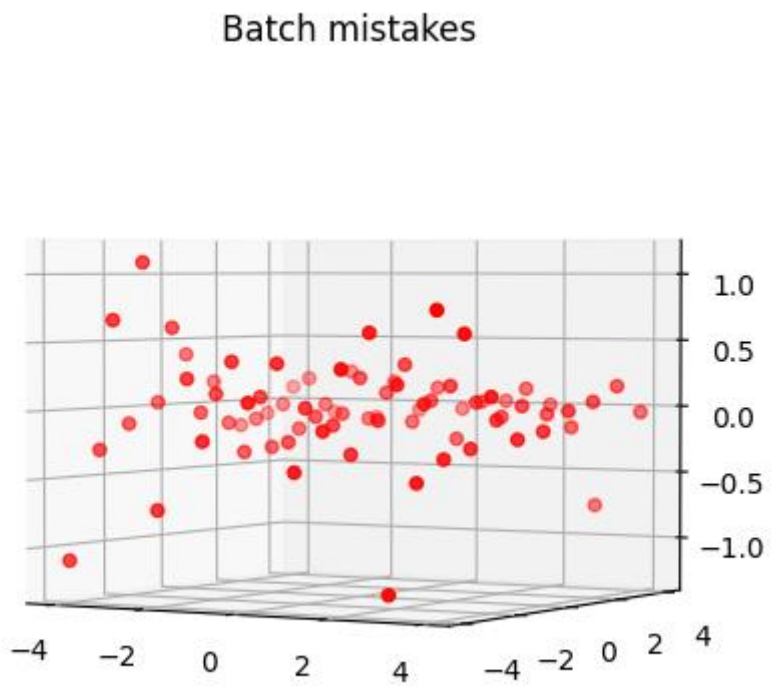
Za slučaj a) i b) zaključujemo da algoritam neće proizvesti smisleno rješenje stoga bi naučene funkcije i odstupanja ($y_k - o_k$) bila velika za svaki ulazni primjer. Jedino u slučaju c) gdje imamo prikladan broj pravila dobivamo ono što smo tražili:

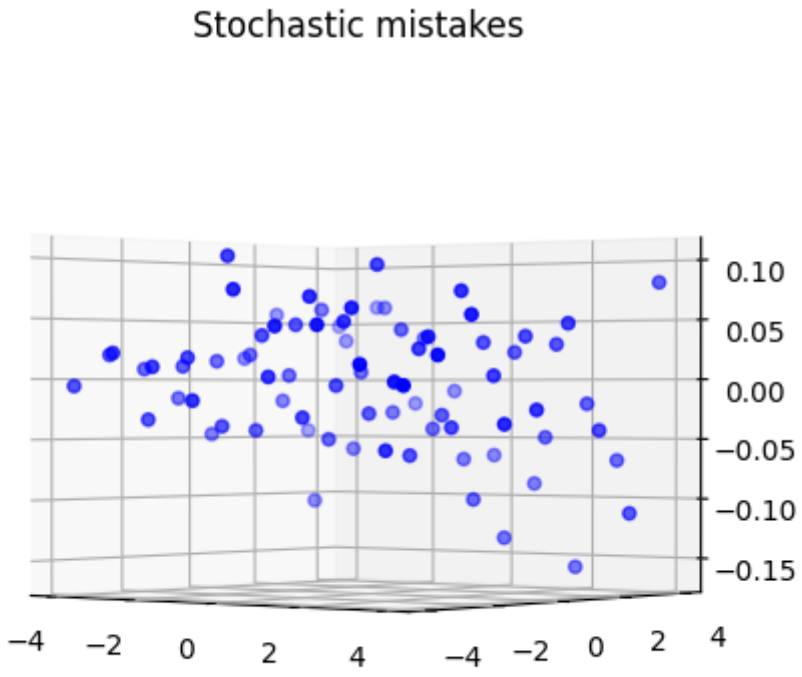
a) vjerno preslikavanje ulazne funkcije



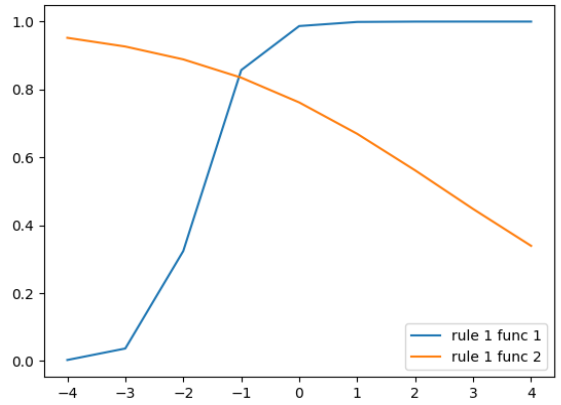


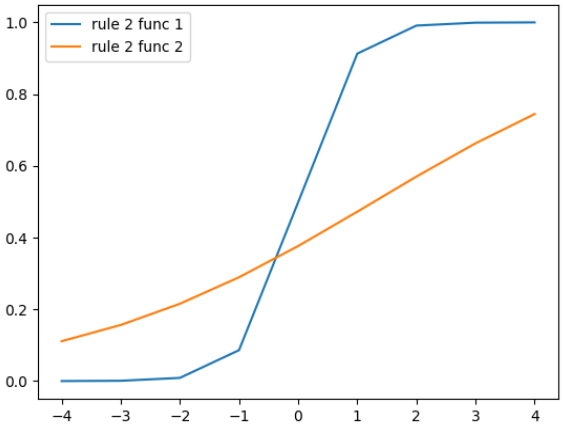
b) mala odstupanja oznaka predviđenih ANFIS-om od onih u skupu za učenje

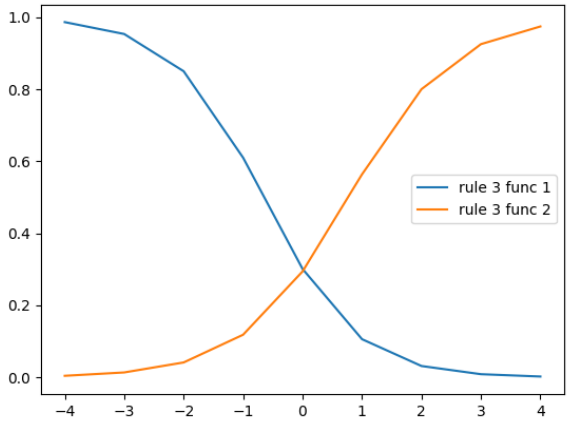


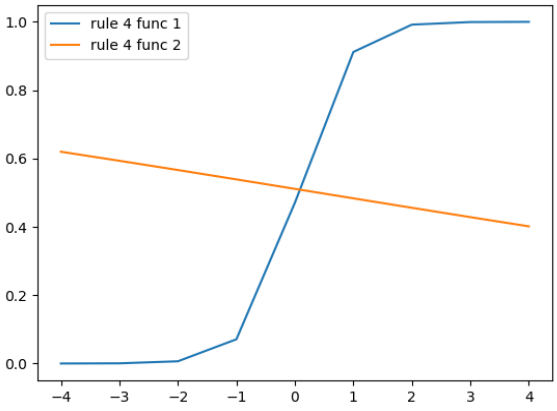


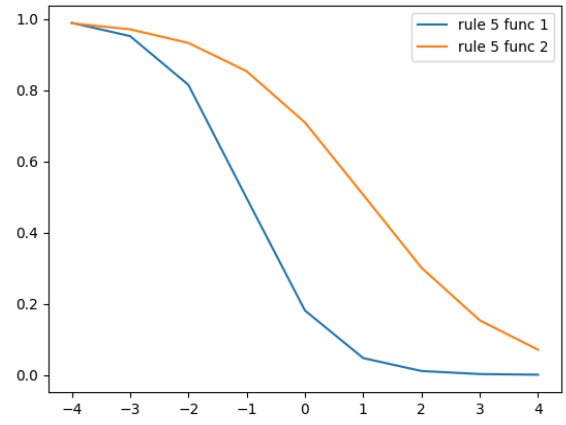
5. Naučena pravila izgledaju ovako. Ovisno o izgledu funkcije mogli bismo smisliti imena. Npr. Ime funkcije 1 pravila 1 neka je: „skoro nenegativan broj“. Imena funkcija pravila 8 neka je: „velik broj“. Obratno vrijedi za funkcije pravila 9: „mali broj“. Tako bi se dalje mogli igrati da funkcijama pridajemo imena. Dakako, to vrijedi samo za one funkcije koje bismo mogli opisati riječima, to jest više ili manje predstavljaju neki jezični izraz.

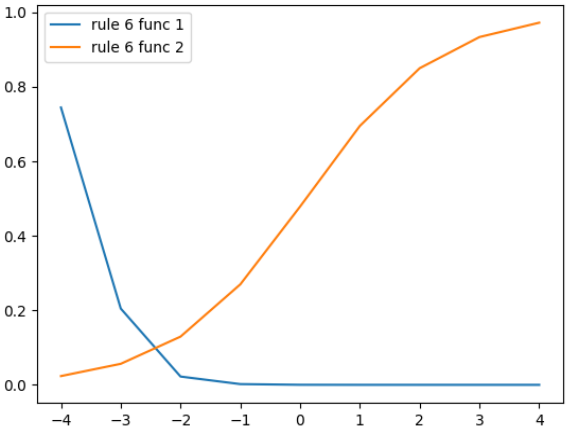


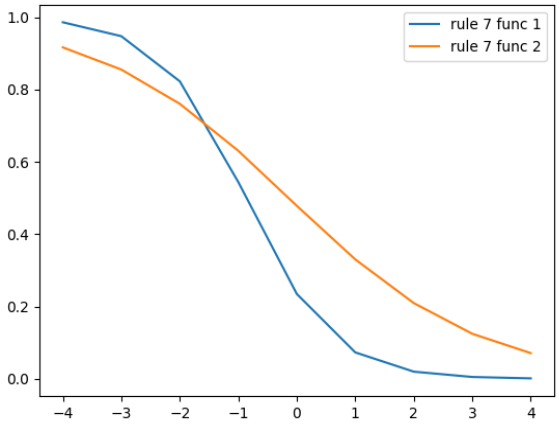


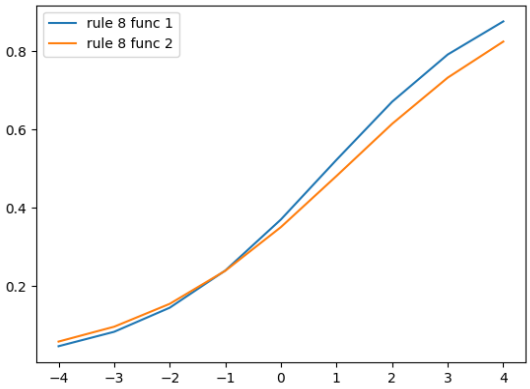


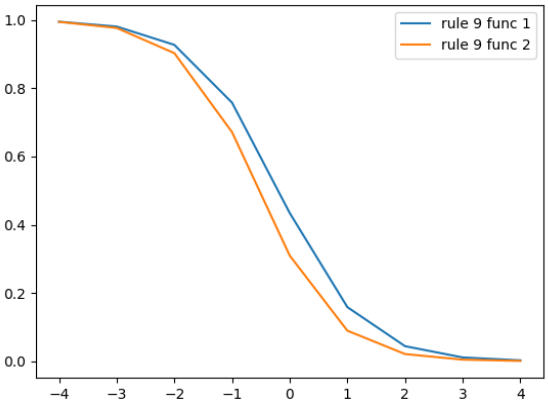


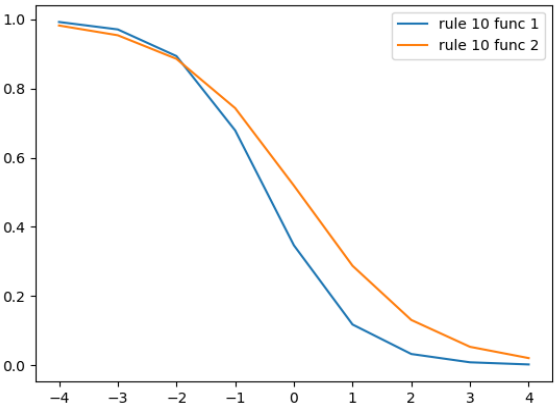


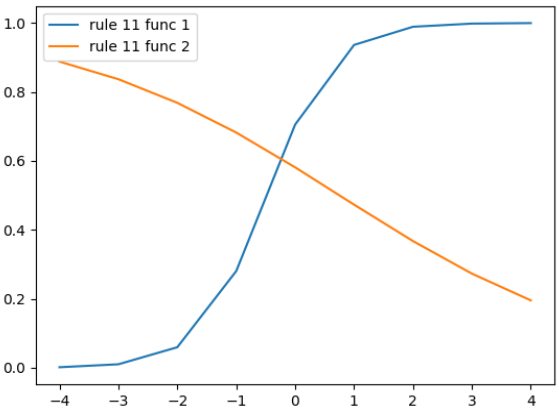


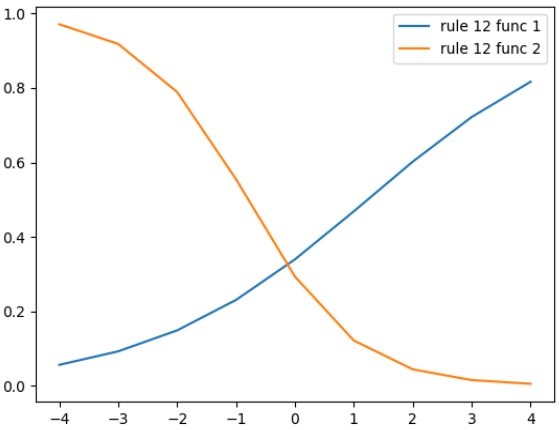

rule 7 func 1
rule 7 func 2

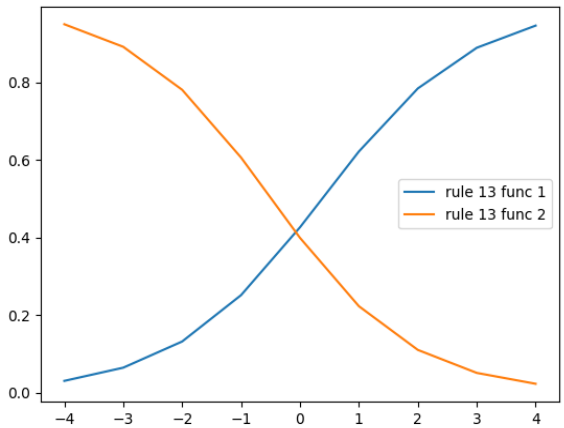

rule 8 func 1
rule 8 func 2

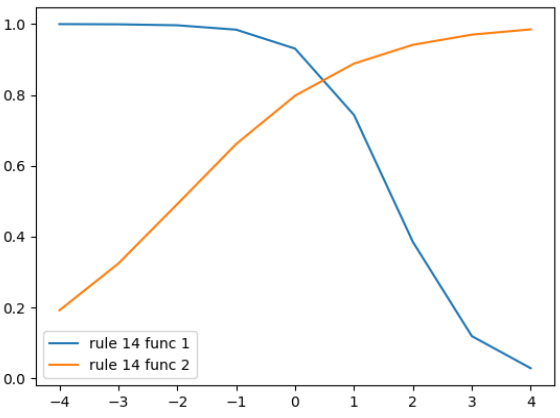

rule 9 func 1
rule 9 func 2

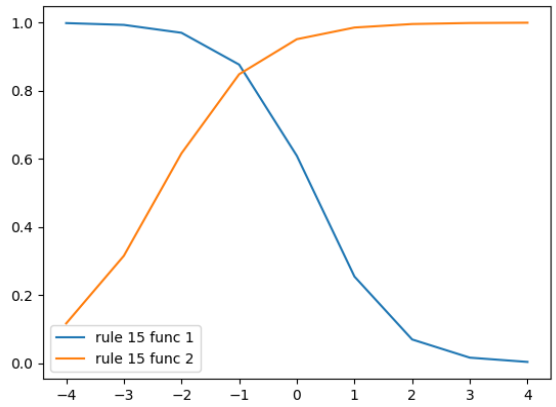

rule 10 func 1
rule 10 func 2

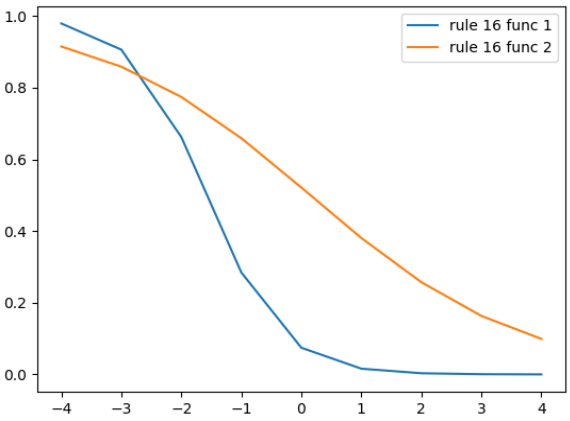

rule 11 func 1
rule 11 func 2

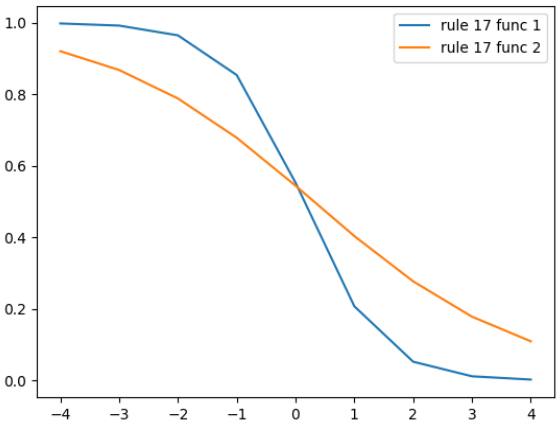

rule 12 func 1
rule 12 func 2

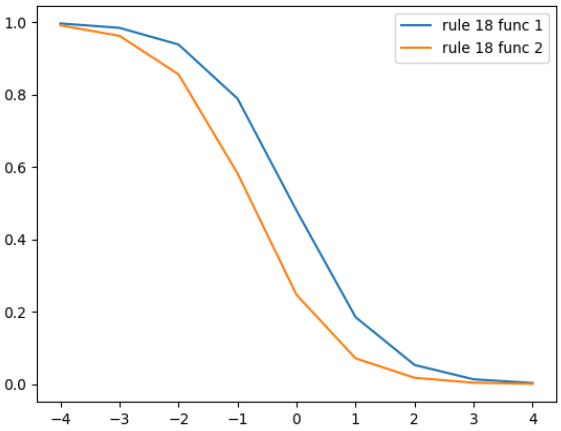

rule 13 func 1
rule 13 func 2

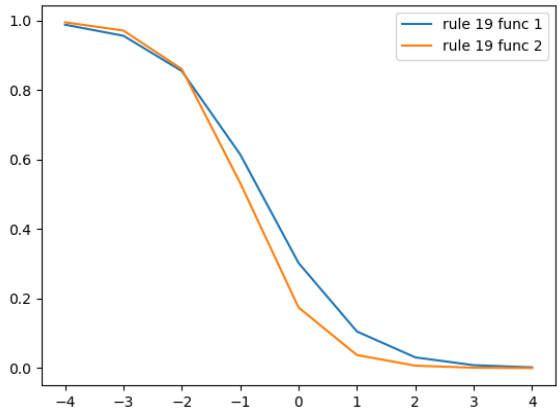

rule 14 func 1
rule 14 func 2

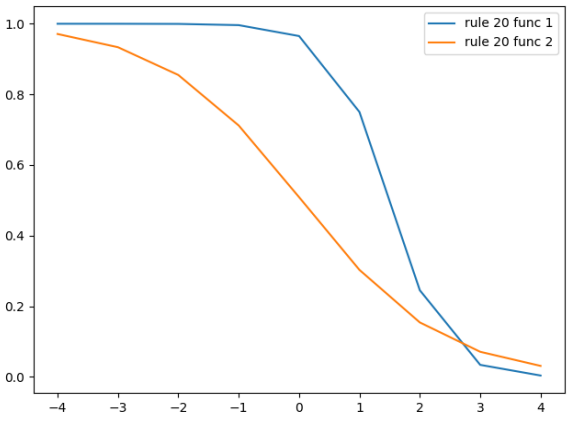

rule 15 func 1
rule 15 func 2


rule 16 func 1
rule 16 func 2


rule 17 func 1
rule 17 func 2

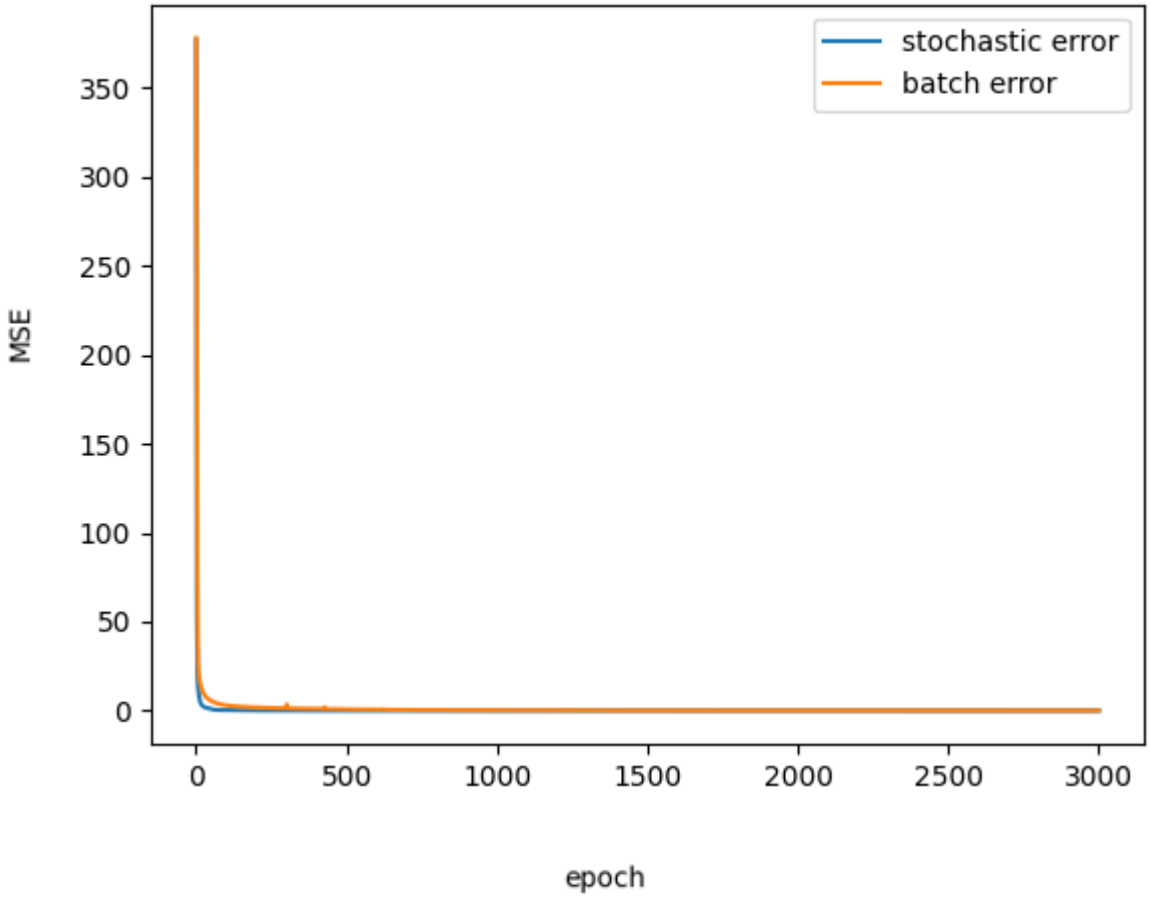

rule 18 func 1
rule 18 func 2


rule 19 func 1
rule 19 func 2


rule 20 func 1
rule 20 func 2

6. Prikazano u 4. c) pod b)

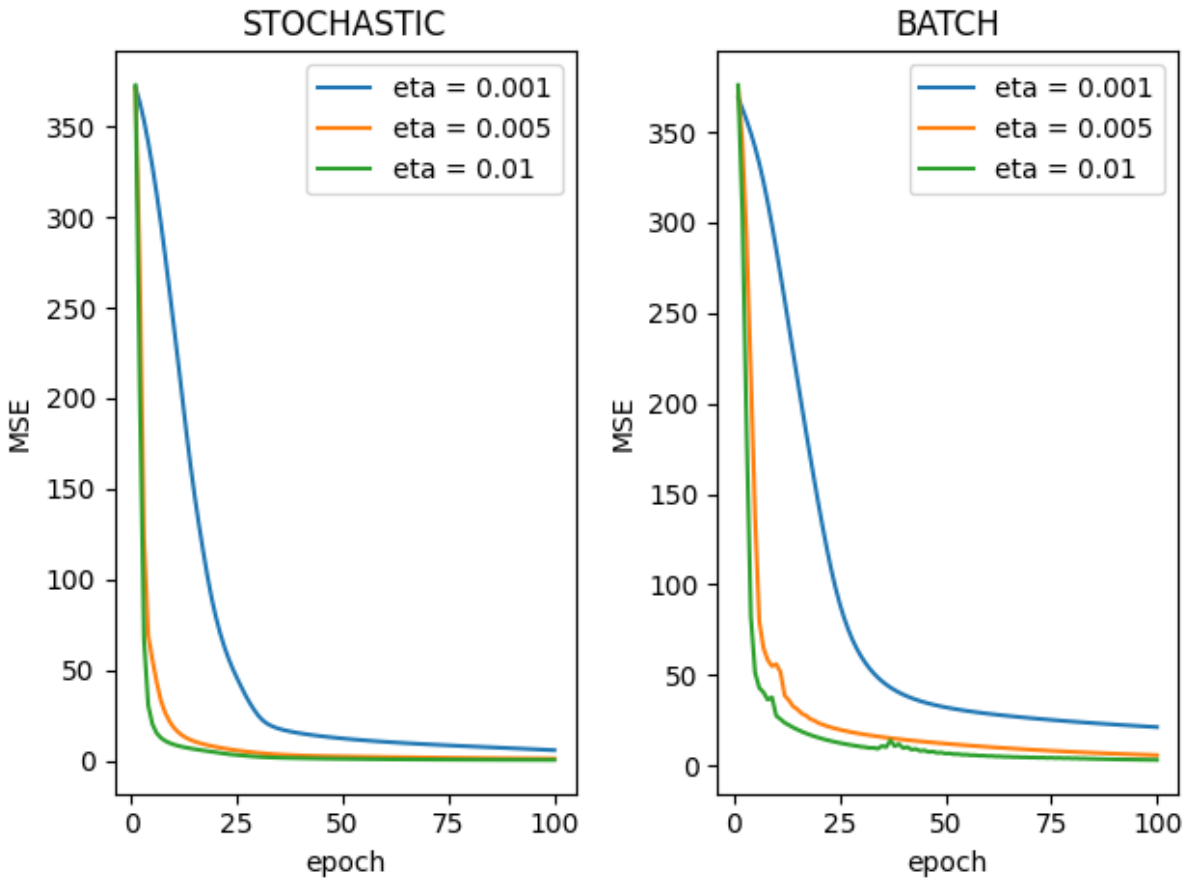
7. Pogreška malo brže prilazi nuli u stohastičkoj varijanti algoritma za istu stopu učenja. Bolje ponašanje stohastičke nad batch varijantom algoritma uz iste uvjete (eta = 0.01 i num_epochs = 3000) dokazuje i slika 4. c) pod b) gdje je vidljivo da stohastička varijanta puno manje griješi.



8. Analizu stopa učenja napravio sam za različit broj epoha kako bi bilo lakše uočiti razliku:

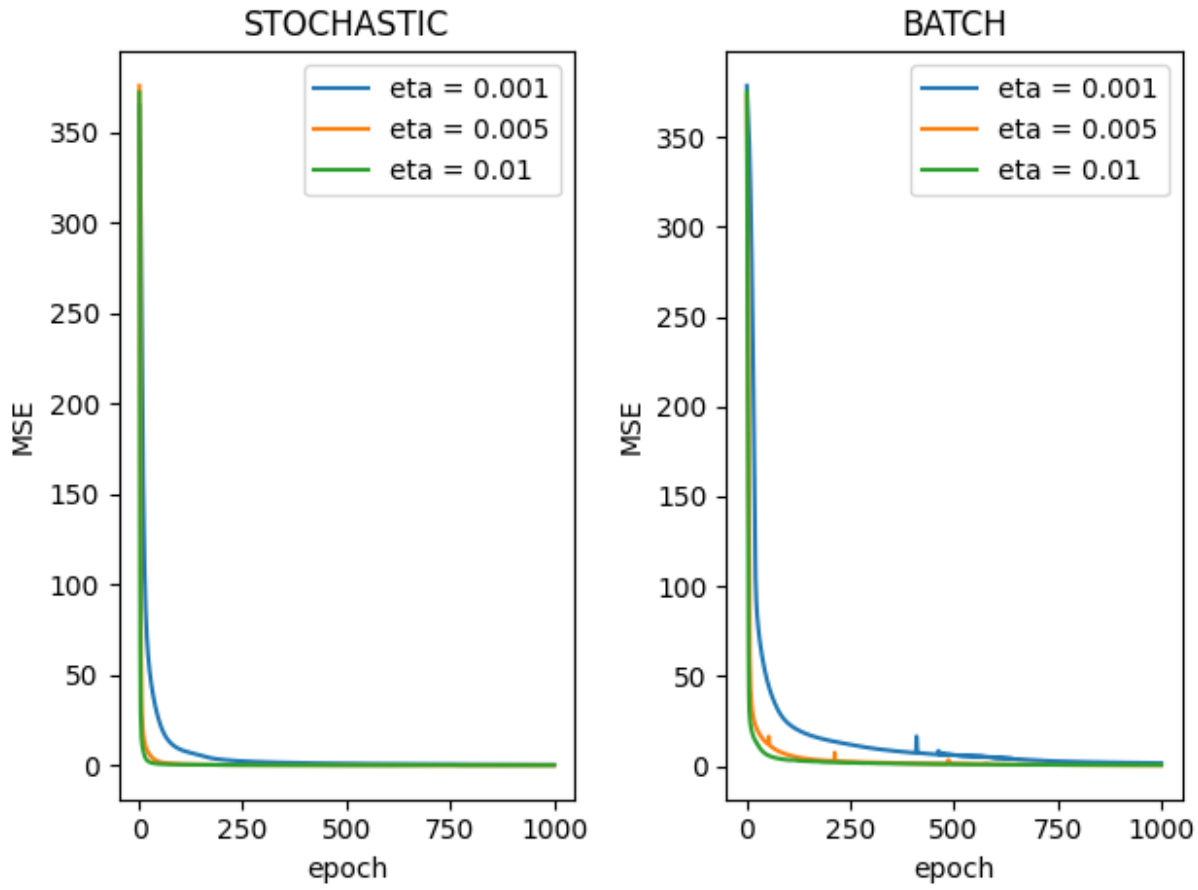
a) num_epochs = 100

Iz grafa je vidljivo da pogreška brže opada za veće stope učenja. U batch varijanti kod stopa učenja eta = 0.005 i eta = 0.01 nemamo kontinuiran pad greške kao u slučaju eta = 0.001. No bez obzira na to greška se smanjuje brže.

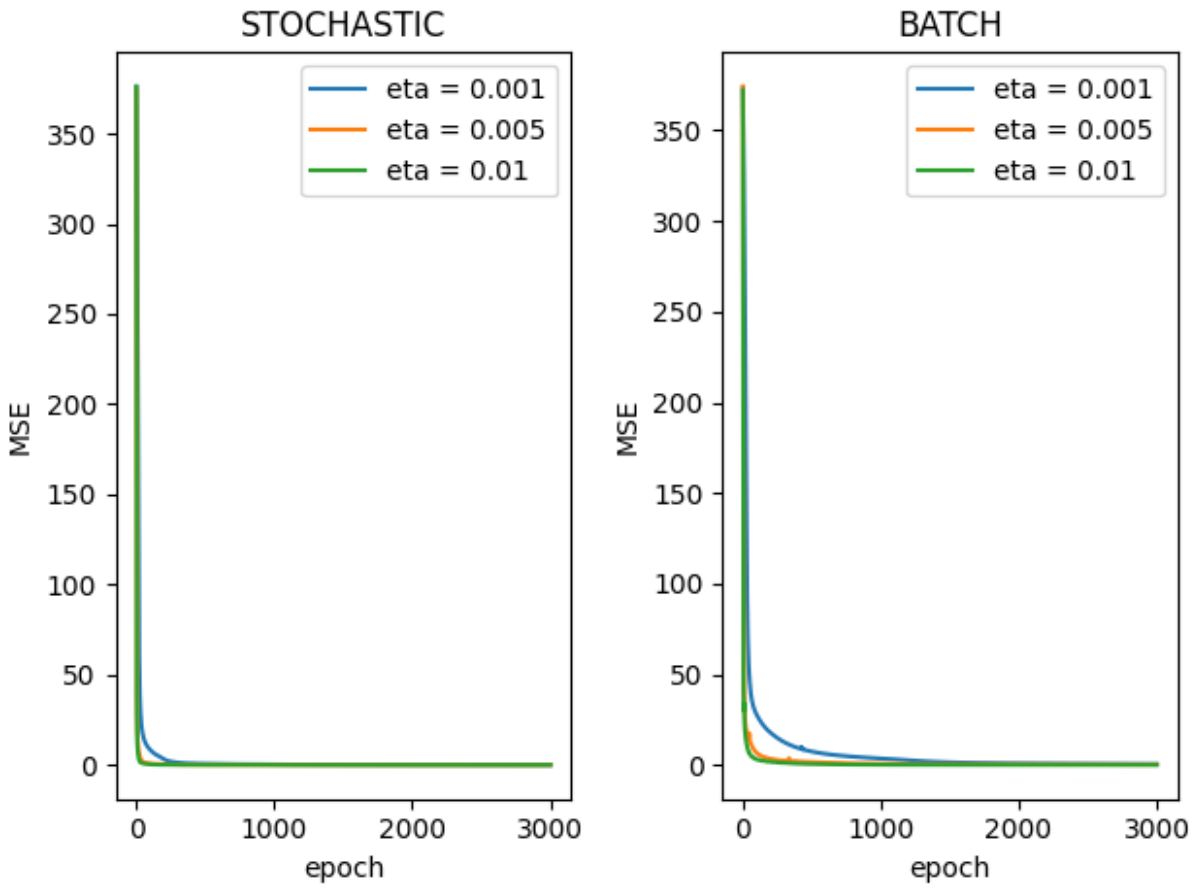


b) num_epochs = 1000

Slično razmatranje vrijedi i kad povećamo broj epoha, no sada u batch varijanti imamo mali skok pogreške čak i u slučaju najmanje stope učenja. Nebitno, konačno se i dalje krećemo u smjeru globalnog optimuma bez značajnijih „skretanja“.

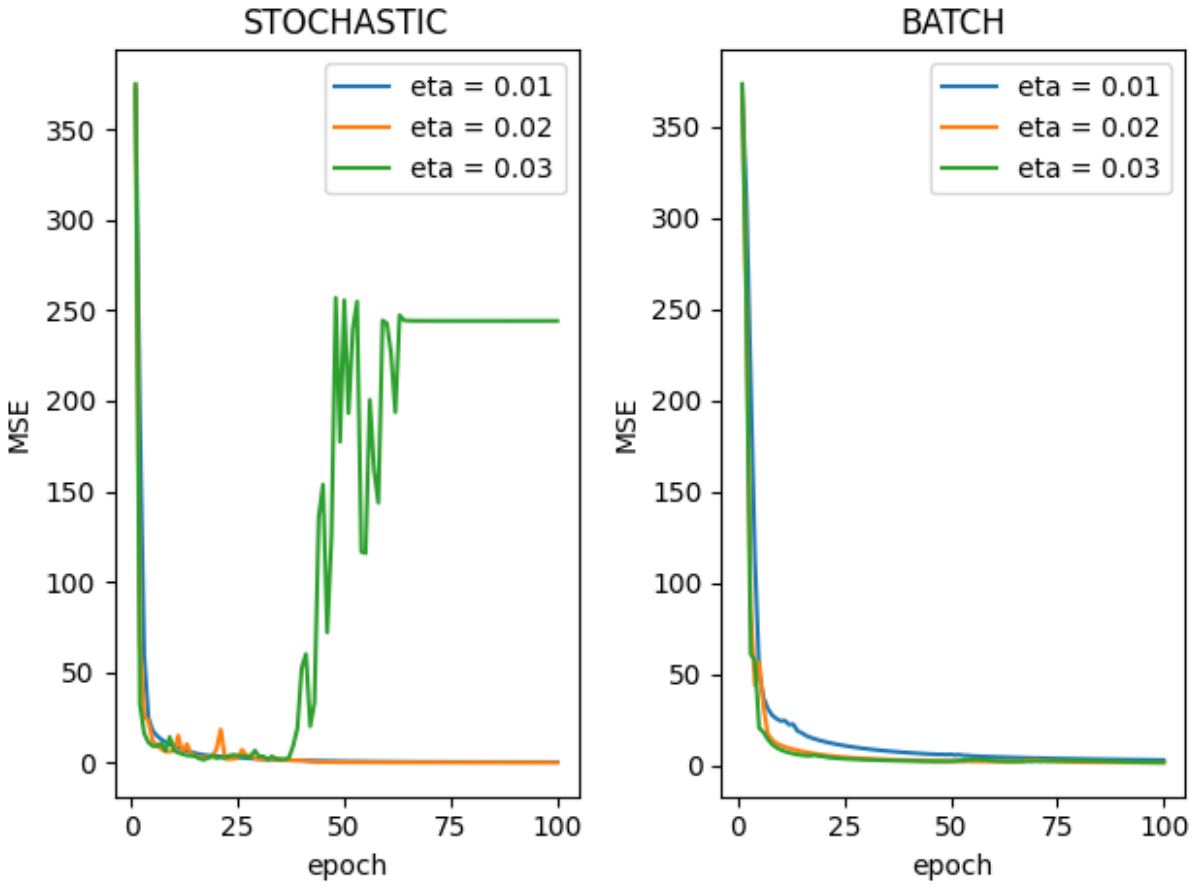


c) num_epochs = 3000

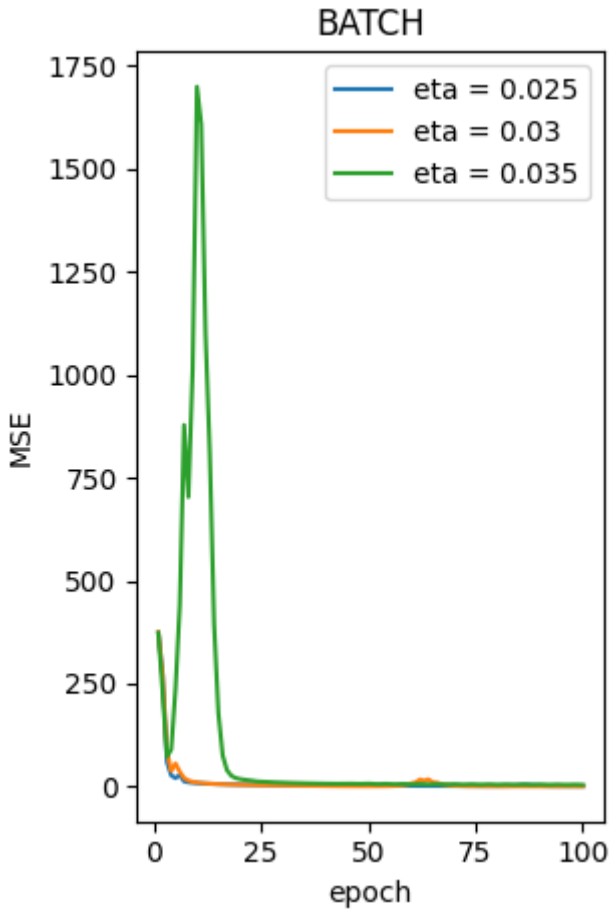


Postavlja se pitanje bi li imalo smisla i dalje povećavati stopu učenja obzirom da bez unatoč minimalnog postojanja alternacije greške konačno ipak opadaju.

Sljedeći graf prikazuje kako je stohastička varijanta na granici s divergencijom već za eta = 0.03, dok batch varijanta i dalje „preživljava“.



Pokušajmo povećati stopu učenja za batch varijantu da utvrdimo koja je njena gornja granica za eta.



Nakon par pokušaja algoritam ipak nije divergirao i „eksplodirao“, odnosno „overflowao“ za eta = 0.035 te smo dobili ovaj prikaz. Labele (vrlo mala, baš prikladna te vrlo velika) za stohastičku varijantu bih odredio kao: (0.001, 0.01, 0.03), a za batch varijantu kao: (0.005, 0.02, 0.035).