Umelé neurónové siete dokumentácia

Ladislav Rajcsányi a Daniel Malenka

Obsah

Rozdelenie prác

Prvú úlohu vypracoval Daniel Malenka, druhú Ladislav Rajcsányi , tretiu a bonusovú sme vypracovali spoločne.

Riešenia sú priložené vo forme spustiteľných programov v **Matlabe**, v priečinku **Figures** sa nachádzajú grafy, ktoré sme vytvorili pomocou programu.

1. Úloha

Úlohou bolo vytvoriť MLP sieť na rozpoznanie 5 skupín bodov , pričom ich bolo treba nájsť na základe troch parametrov (**x** , **y** , **z**) . Dáta sú uložené v súbore **databody2.mat** a nájsť čo najmenší počet neurónov v skrytej vrstve , tak aby neurónová sieť správne klasifikovala čo najviac bodov.

Vypracovanie :

Naše súradnice x , y , z , tvorili vstupy do neurónovej siete na základe ktorých sme určili do ktorej skupiny bude bod patriť . Pre výstup sme si vytvorili maticu ktorá mala riadky pretože sme mali výstupov a teda ich bolo treba zatriediť k jednej z daných skupín (výstupov) .

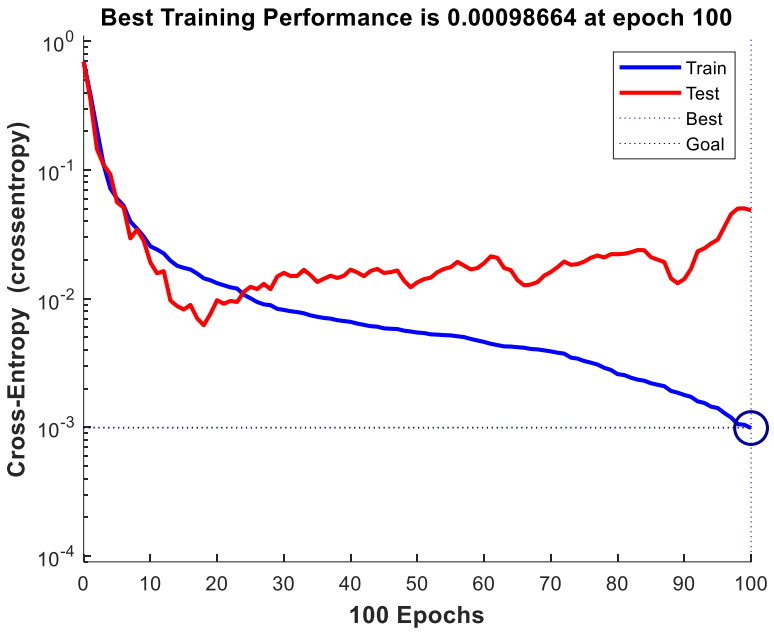
Štruktúra MLP siete a parametre trénovania :

Na vytvorenie MLP siete na klasifikáciu sme použili štruktúru **patternet**. Mala 1 skrytú vrstvu v ktorej sa nachádzalo 20 neurónov, ktoré mali aktivačnú funkciu **tansig** (**hyperbolický tangens**). Počet neurónov sme hľadali experimentálne. Výstupné neuróny, 5 neurónov, mali aktivačnú funkciu **softmax**. Sieť používala kriteriálnu funkciu **cross-entropy**. Pre ukončovacie podmienky sme stanovili tak že prvú , ktorá slúži na dosiahnutie minimálnej chyby sme nastavili na **net.trainParam.goal = 0. 001; . resp.= 1*10^-3**. Druhá slúži na ukončenie trénovania ak zmena chyby bude malá – **net.trainParam.min_grad=1e-6; . resp . =1*10^-6**

Ďalej sme nastavili ešte max. počet trénovacích cyklov na hodnotu = **100** , čo slúži najmä na to ak by sme nevedeli dosiahnuť minimálnu chybu. Na trénovanie sme vybrali náhodný výber dát (**net.divideFcn='dividerand'**) a to 80% z celkového počtu našich dát. Zvyšných 20% sme použili na trénovanie.

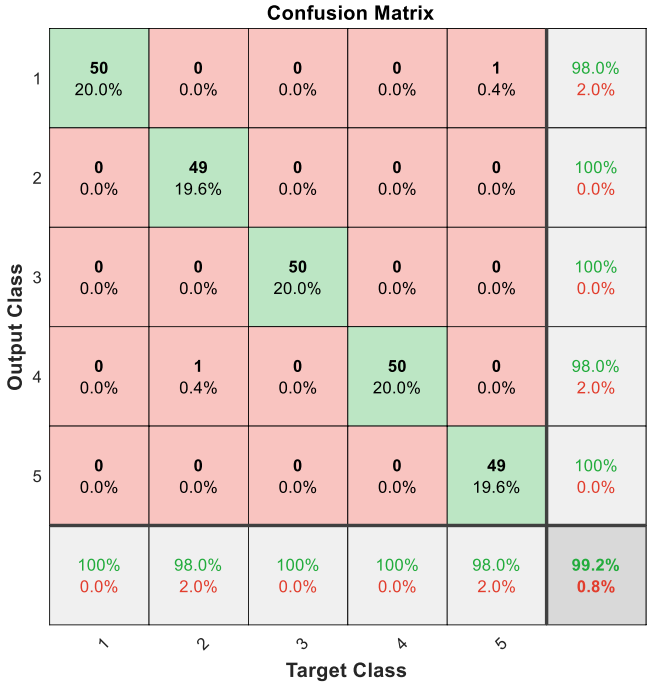
Trénovanie a výsledky :

Požadovanú chybu sme dosiahli už po 100 trénovacích cykloch.



Figures – Uloha1 – Uloha1_Performance

Ďalej sme mohli použiť kontingenčnú maticu , kde si môžeme všimnúť že neboli rozpoznané 2 vzorky ktoré mali byť : jedna by mala byť klasifikovaná v druhej triede a druhá v piatej. Preto je naša úspešnosť 99.2 %.



Figures – Uloha1 – Uloha1_All_Confusion_Matrix

Testovanie 5 bodov a ich triedenie :

Bod	x	y	z
1.	0.9000	0.1000	0.9000
2.	0.5000	0.7000	0.1000
3.	0.2000	0.8000	0.6000
4.	0.9000	0.2000	0.3000
5.	1.0000	1.0000	1.0000

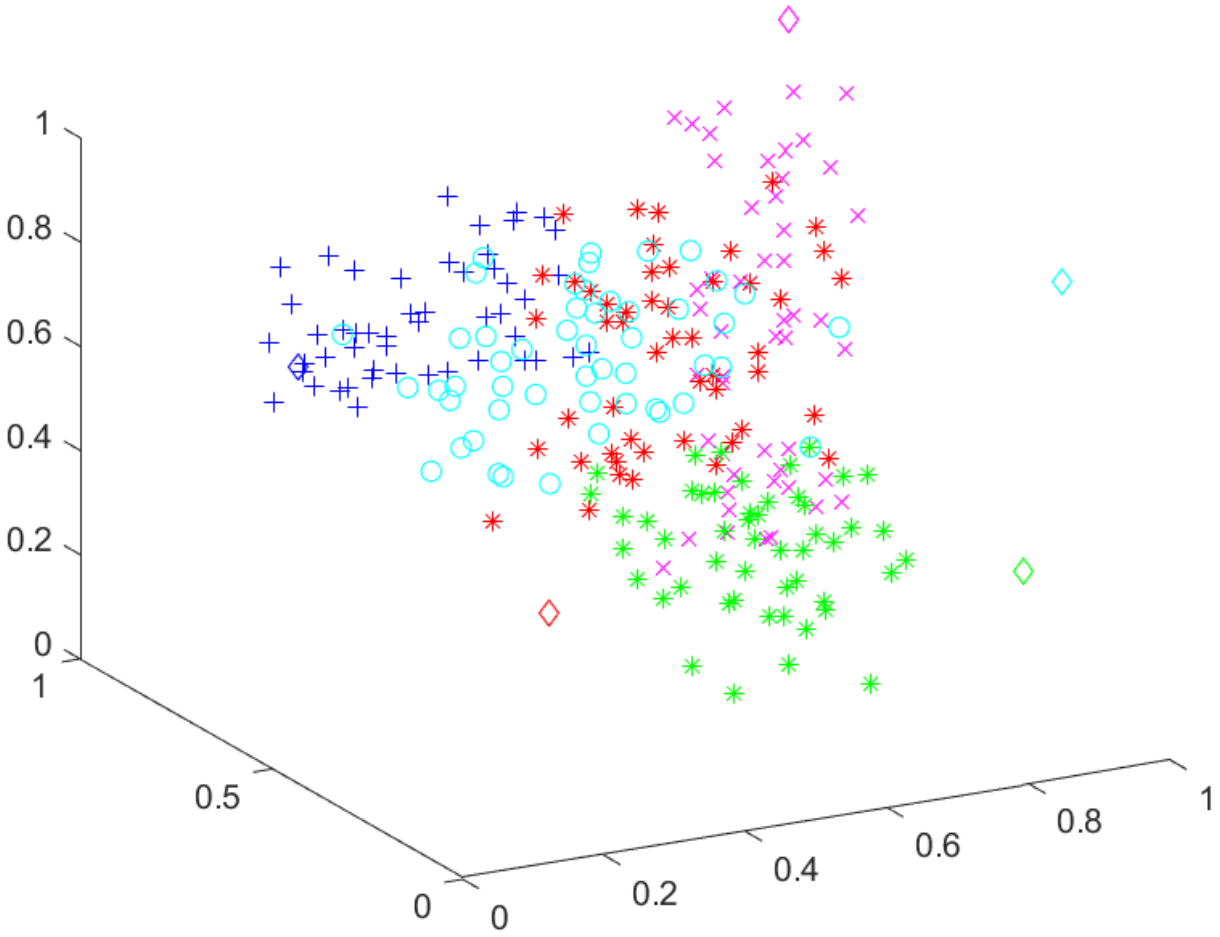
Trénovanie sme riešili takým spôsobom, že najprv sme vytvorili maticu s veľkosťou 5x3, a hodnoty v matici vyzerali ako vo vyššie uvedenej tabuľke. Tieto body sme museli transponovať, lebo vstupom sú tri parametre (x , y , z). Potom tieto transponované body sme dali do neurónovej sieti ako vstupné dáta.

0.0000	0.0000	1.0000	0.0000	0.0000
1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	1.0000	0.0000
0.0000	1.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	1.0000

Na výstup sme dostali takéto riadky, každý riadok reprezentuje jednu triedu, do ktorej môžu body patriť. Každý stĺpec reprezentuje zatriedenie jedného vstupného bodu do triedy.
V každom prípade máme tam hodnoty z intervalu (0,1) ktoré budú reprezentovať percentá nakoľko bola sieť schopná zaradiť do správnej triedy. Pomocou funkcie **vec2ind** môžeme tieto výstupné dáta meniť na triedy.

2	4	1	3	5

Zadefinovali sme formu a farby , ktorými by sme chceli zobraziť definované body. Vybrali sme diamantovú formu (**d**). Po zobrazení vstupných dát, ktoré sme načítali z **databody2**, sme zobrazili ešte 5 definovaných bodov. Celý graf vyzeral nasledovne:



Figures – Uloha1 – Uloha1_Testovanie

2. Úloha

Úlohou bolo vytvoriť MLP sieť na aproximáciu nelineárnej funkcie f so vstupom x a výstupom y. Po trénovaní sme mali v grafe farebne označiť trénovacie, testovacie dáta a výstup neurónovej siete.

Vstupné a výstupné dáta:

Načítali sme dáta, ktoré boli uložené v súbore **datafun2**. Tento súbor obsahoval súradnice **x**, **y** okrem nich sme tam mali aj **indx_train** a **indx_test**, ktoré budeme používať pri indexovom rozdelení. Hodnoty uložené v **x** budeme používať, ako vstupné dáta neurónovej siete, a hodnoty v **y** chceme dosiahnuť. Používali sme indexové rozdelenie, ktorú sme nastavili nasledovne:

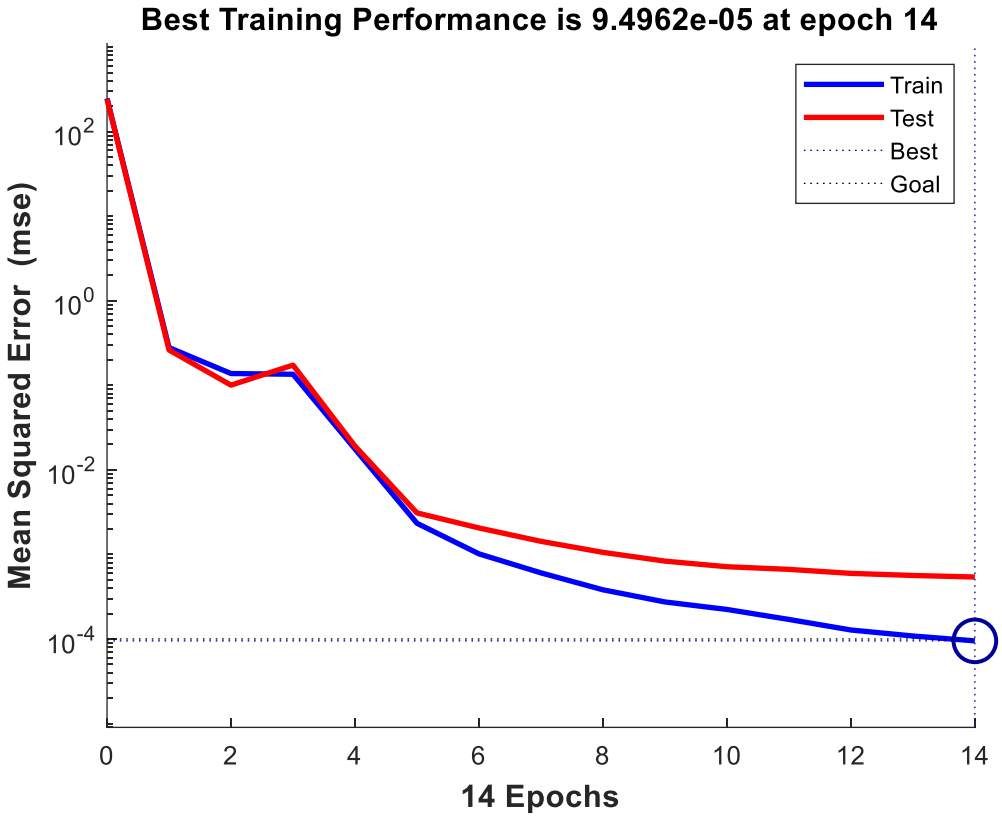
```
net.divideFcn='divideind';
net.divideParam.trainInd=indx_train;
net.divideParam.testInd=indx_test;
```

Štruktúra MLP siete a parametre trénovania:

Používali sme štruktúru **fitnet.** Mali sme len jednu skrytú vrstvu, ktorá obsahovala **15** neurónov, ktoré majú aktivačnú funkciu **tansig** (**hyperbolický tangens**). Mali sme výstupný neurón, ktorý mal aktivačnú funkciu **purelin**. Sieť používala trénovaciu metódu **Levenberg-Marquardt**. Počet neurónov sme hľadali tak isto experimentálne ako v 1. úlohe. Jediná ukončovacia podmienka bola dosiahnutie minimálnej chyby, ktorú sme nastavili na **0.0001** teda **1*10^-4** (alebo **1e-4**).

Trénovanie a výsledky :

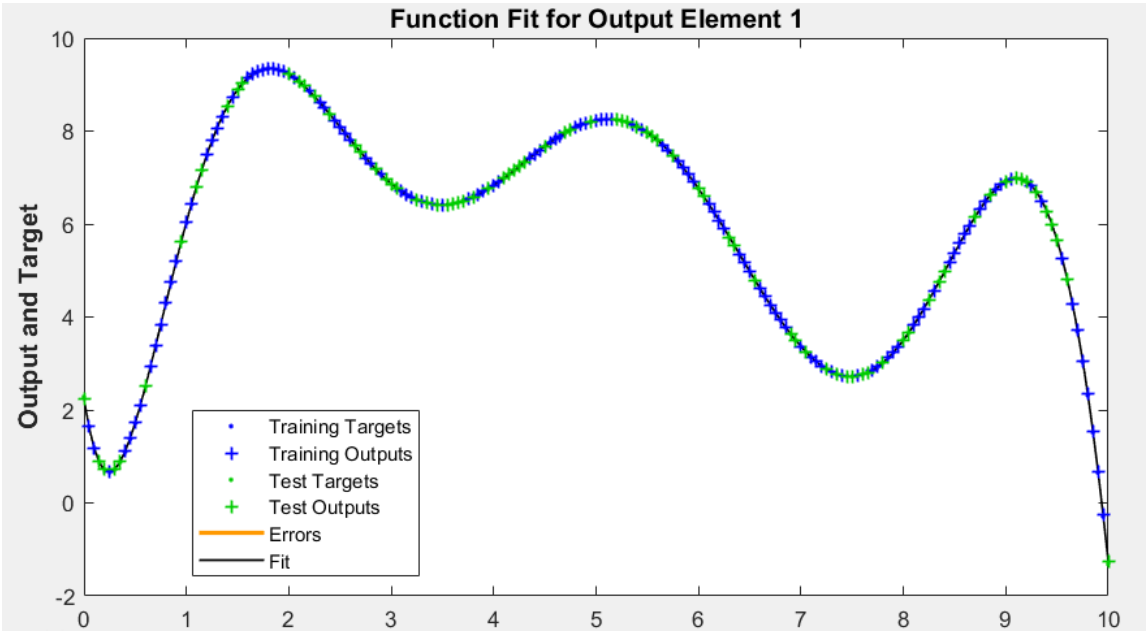
Pri používaní 15 neurónov v skrytej vrstve sme dosiahli nasledujúce výsledky:

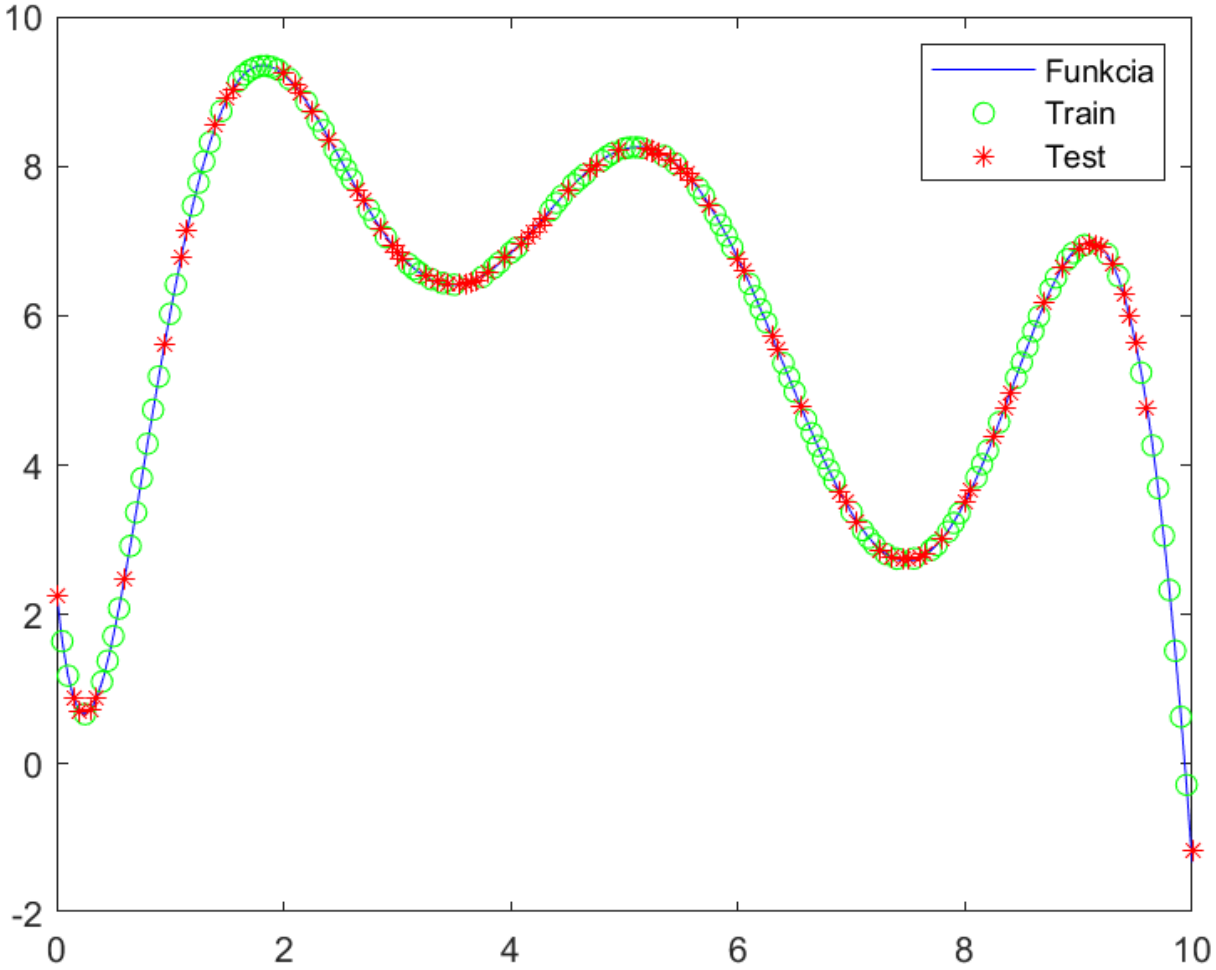


Figures – Uloha2 – 15Neuron – Uloha2_Performance_15N

Minimálnu chybu (10^-4) sme dosiahli po 14 trénovacích cykloch .

Keďže teraz nešlo o klasifikačnej úlohe, kontingenčnú maticu sme tu nemali, ale namiesto toho sme našli **fitgraf** a okrem toho sme vytvorili aj vlastný graf.





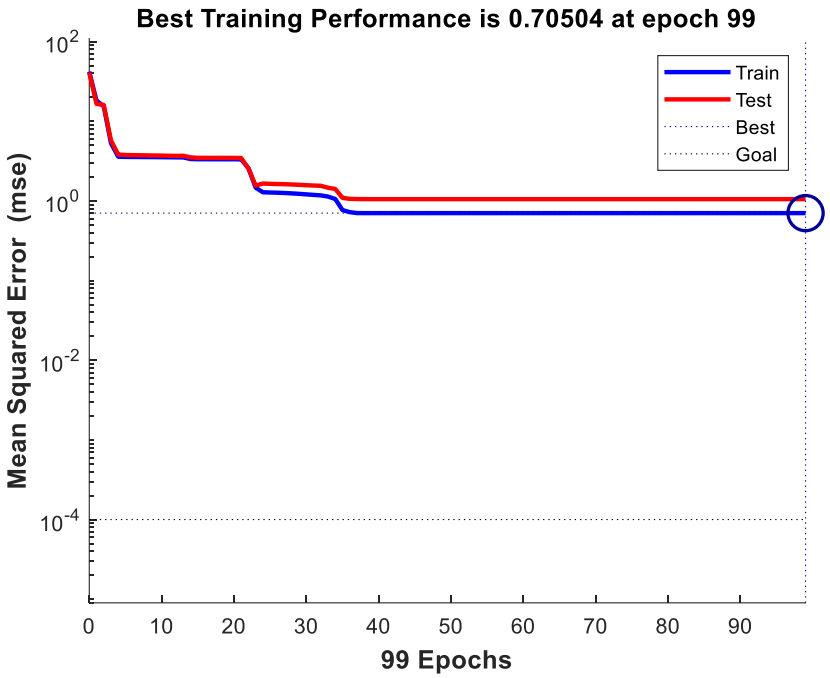
Figures – Uloha2 – 15Neuron – Uloha2_Graf_15N

Po zobrazení grafov sme vyčísli chyby **SSE** (suma kvadrátov odchýliek medzi meraným výstupom a výstupom siete), **MSE** (priemer z SSE), **MAE** (maximálna absolútna odchýlka medzi meraným výstupom a výstupom siete) na trénovacích a testovacích dátach. Ktoré vyzerali nasledovne:

```
==============SSE==============
SSE for network:
    0.0552

SSE for training data:
    0.0114

SSE for testing data:
    0.0438
==============MSE==============
MSE for network:
   2.7448e-04

MSE for training data:
   9.4962e-05

MSE for testing data:
   5.4043e-04

==============MAE==============
MAE for network:
    0.1551

MAE for training data:
    0.0289

MAE for testing data:
    0.1551
```

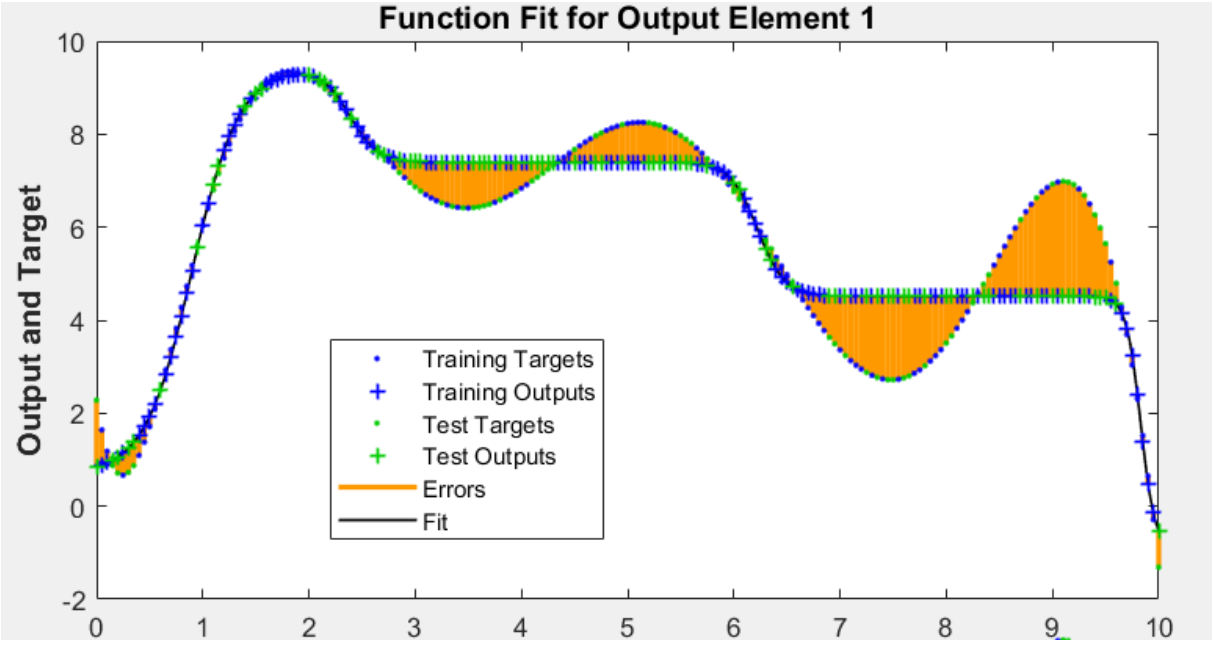
Figures – Uloha2 – MSE_MAE_SSE – 15Neuron – SSE_15,MSE_15,MAE_15

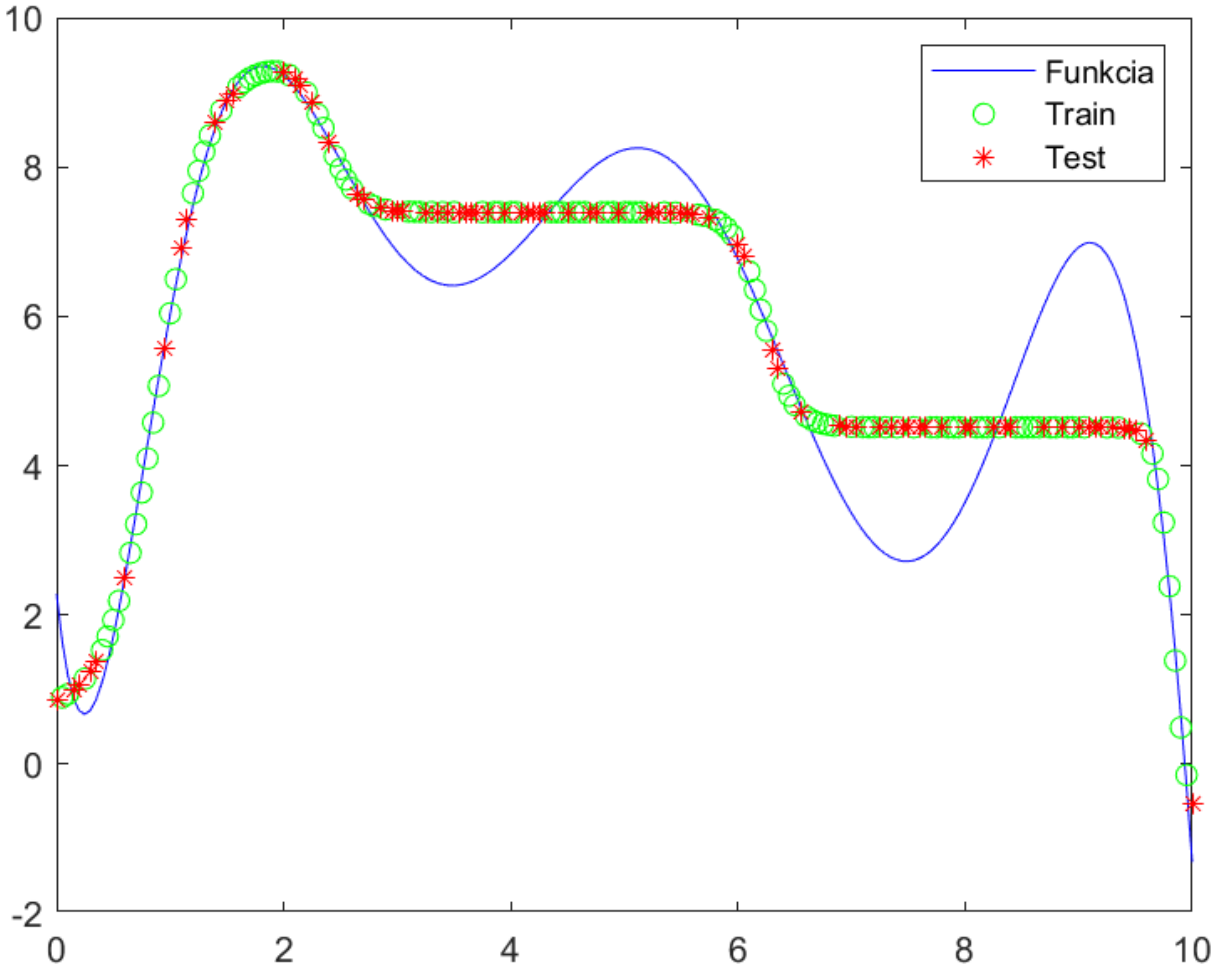
Pri používaní 4 neurónov v skrytej vrstve sme dosiahli nasledujúce výsledky:



Figures – Uloha2 – 4Neuron – Uloha2_Performance_4N

Minimálnu chybu (10^-4) sme nedosiahli ani po 99 trénovacích cykloch . Toto znamená že na trénovachích a testovacích dátach budeme mať dosť veľa chýb.





Figures – Uloha2 - 4Neuron - Uloha2_Graf_4N

Ako sme už videli na **performance** grafe na trénovacích a testovacích dátach naša sieť má dosť veľkú chybovosť.

Dosiahli sme tieto chybové hodnoty:

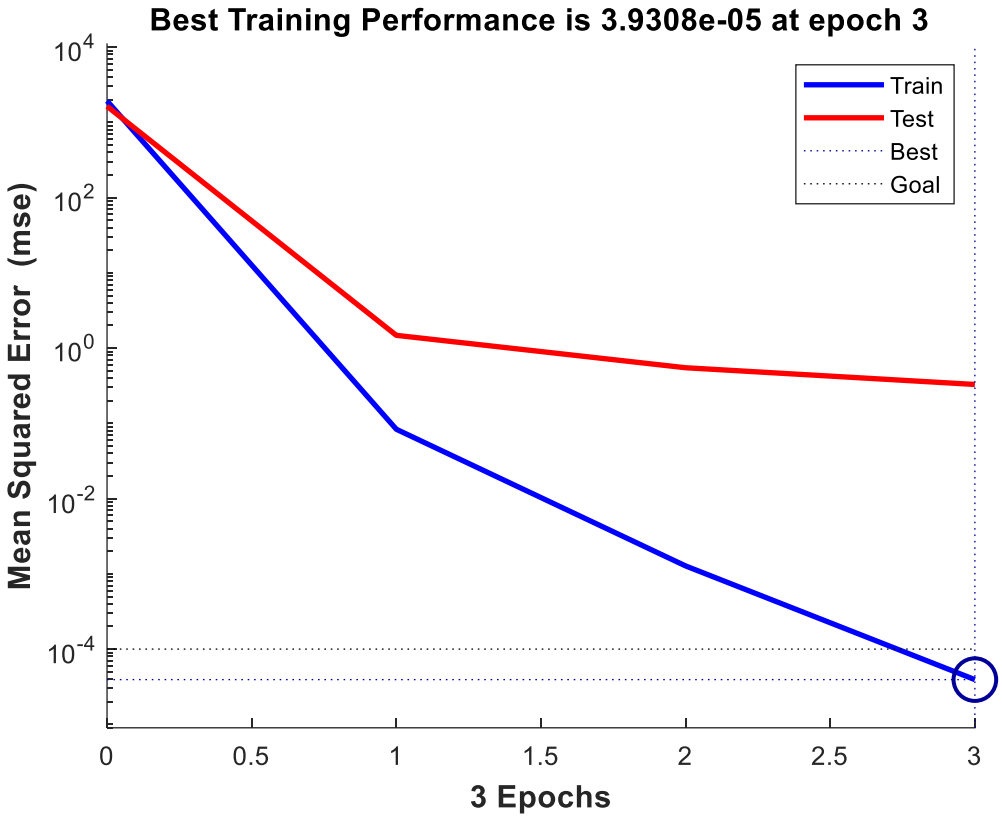
```
===============SSE===============
SSE for network:
  170.1498

SSE for training data:
   84.6044

SSE for testing data:
   85.5454
```

```
===============MSE===============
MSE for network:
    0.8465

MSE for training data:
    0.7050

MSE for testing data:
    1.0561

===============MAE===============
MAE for network:
    2.4751

MAE for training data:
    2.4594

MAE for testing data:
    2.4751
```

Figures – Uloha2 – MSE_MAE_SSE – 4Neuron – SSE_4,MSE_4,MAE_4

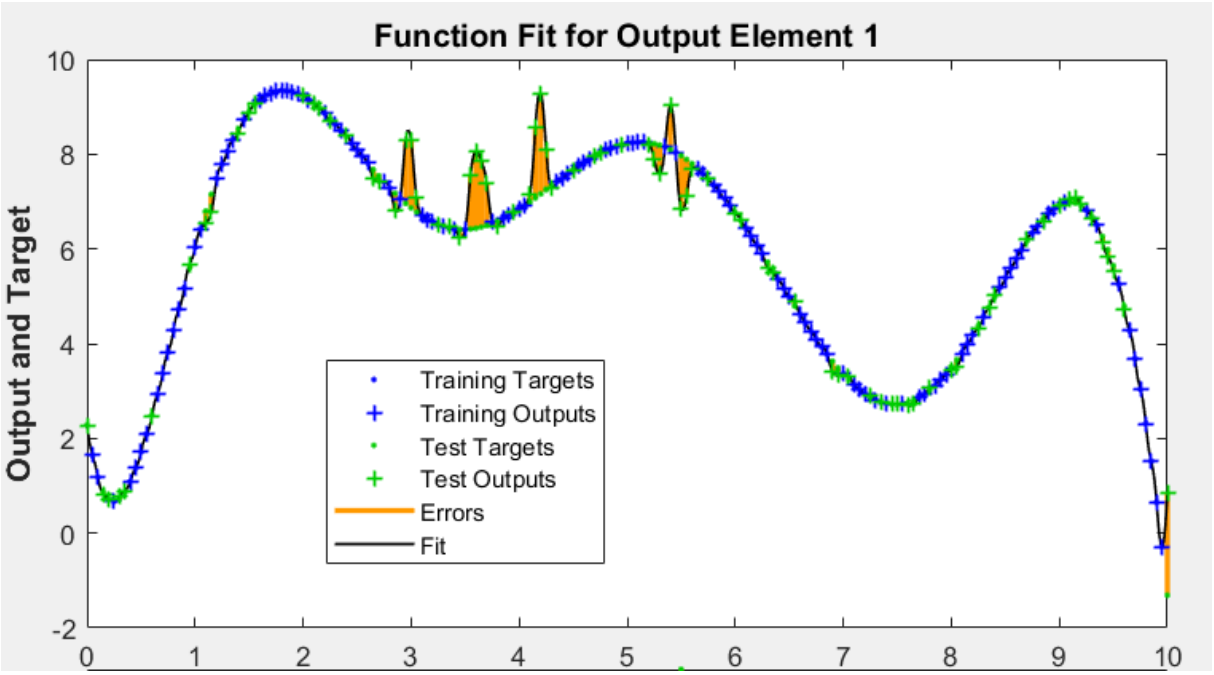
Tieto hodnoty sú oveľa horšie ako v prípade 15 neurónov.

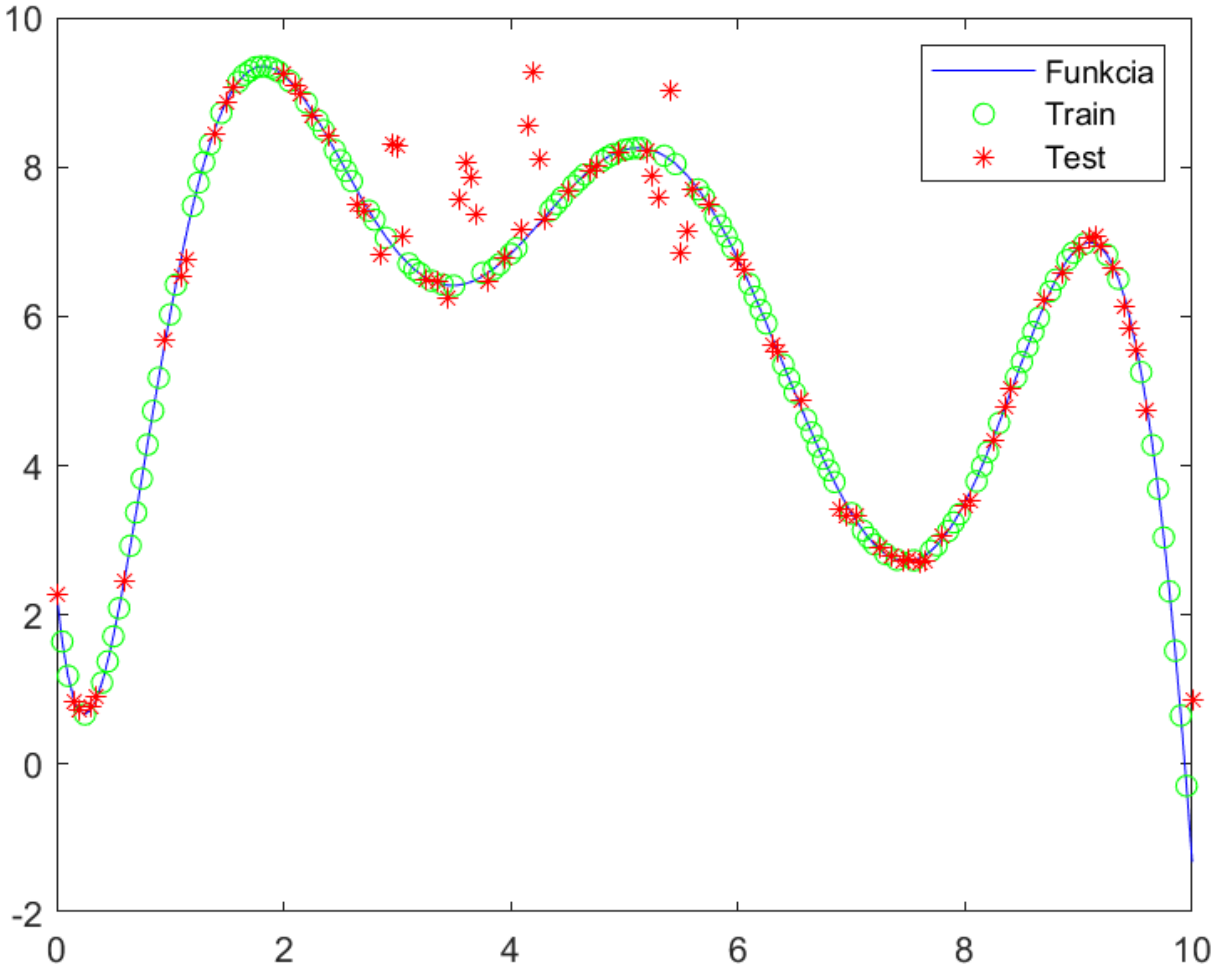
Potom sme vyskúšali čo sa stane, ak máme príliš veľa neurónov. Dosiahli sme nasledujúce výsledky:



Figures – Uloha2 – 100Neuron – Uloha2_Performance_100N

Z grafu je vidno, že na trénovacích dátach budeme mať menej chýb, a na testovacích bude z toho viacej.





Figures – Uloha2 – 100Neuron – Uloha2_Graf_100N

Na trénovacích dátach sieť dosiahla celkom dobré výsledky, ale na miestach, kde neurónová sieť počas trénovania nemala informácie (testovacie dáta) dochádzalo k veľkej chybovosti. Chybové hodnoty boli nasledovné:

```
===============SSE===============
SSE for network:
   26.6438

SSE for training data:
    0.0047

SSE for testing data:
   26.6391
```

```
===============MSE===============
MSE for network:
    0.1326

MSE for training data:
   3.9308e-05

MSE for testing data:
    0.3289
===============MAE===============
MAE for network:
    2.1687

MAE for training data:
    0.0482

MAE for testing data:
    2.1687
```

Figures – Uloha2 - MSE_MAE_SSE - 100Neuron - SSE_100,MSE_100,MAE_100

Aj tu je vidno, že na trénovacích dátach sme dosiahli celkom nízke chyby, ale na testovacích dátach sme dostali oveľa väčšie.

3. Úloha

Mali sme vytvoriť MLP sieť, ktorá bude schopná na základe 19 parametrov rozpoznať (klasifikovať) ochorenie očného pozadia Diabetickú retinopatiu. Mali sme ich klasifikovať do dvoch skupín 0 – žiadne ochorenie, 1 – diagnostikované ochorenie diabetickej retinopatie.

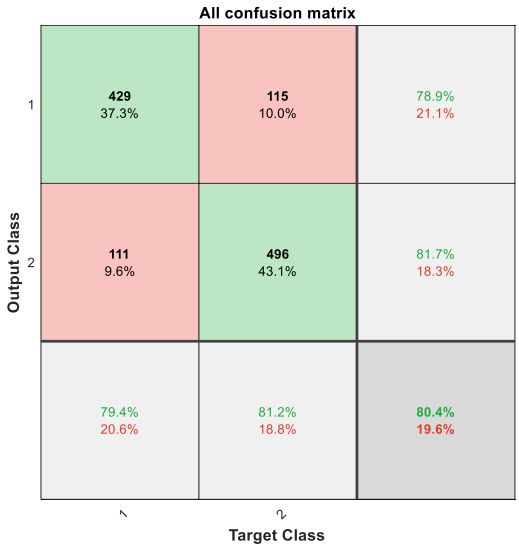
Vstupné a výstupné dáta:

Načítali sme dáta zo súboru **dataonly.txt**. Rozdelili sme ich do dvoch skupín. Prvých 19 transponovaných stĺpcov sme nazvali **inputs**, z 20. stĺpca sme vytvorili dvojriadkovú maticu takým spôsobom , že ak je v poslednom stĺpci 0 ide o zdravú vzorku, na 1. výstup siete sa dá jednotka, ak sme tam mali 1 ide o chorú vzorku, na 2. výstup siete sa dá jednotka. Po transponovaní sme túto maticu nazvali **targets**, teda hodnoty, ktorých chceme dosiahnuť. Po testovaní na výstup dostaneme dvojriadkovú maticu, presnejšie 2x1151.

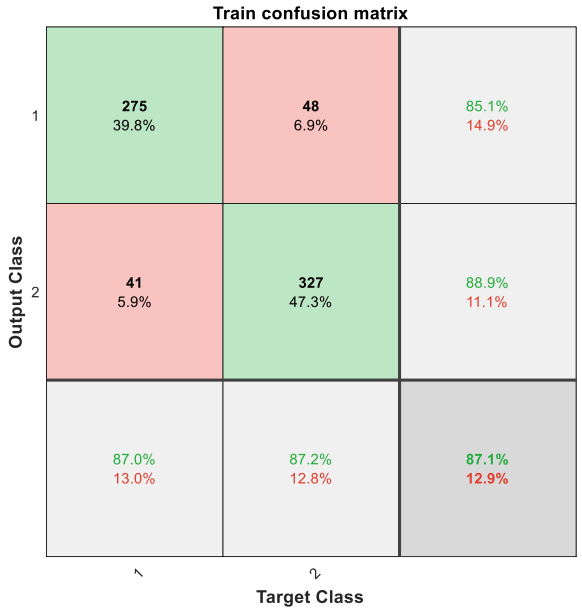
Štruktúra MLP siete a parametre trénovania:

Keďže ide o klasifikačnú úlohu, podobne ako v 1. úlohe sme používali štruktúru **patternet**. Mala 1 skrytú vrstvu v ktorej sa nachádzalo **40** neurónov, ktoré mali aktivačnú funkciu **tansig** (**hyperbolický tangens**). Výstupné neuróny, 2 neuróny, mali aktivačnú funkciu **softmax**. Sieť používala kriteriálnu funkciu **cross-entropy**. Prvá ukončovacia podmienka bola dosiahnutie minimálnej chyby, ktorú sme nastavili na **0.000001 . resp.= 1*10-6.** Druhá slúži na ukončenie trénovania ak zmena chyby bude malá - **net.trainParam.min_grad = 1e-10; . resp . =1*10^-10**. Ďalej sme nastavili ešte max. počet trénovacích cyklov na hodnotu **600**. Na trénovanie sme vybrali náhodný výber dát (**net.divideFcn='dividerand'**) a to 60% z celkového počtu našich dát. Zvyšných 40% sme použili na trénovanie.

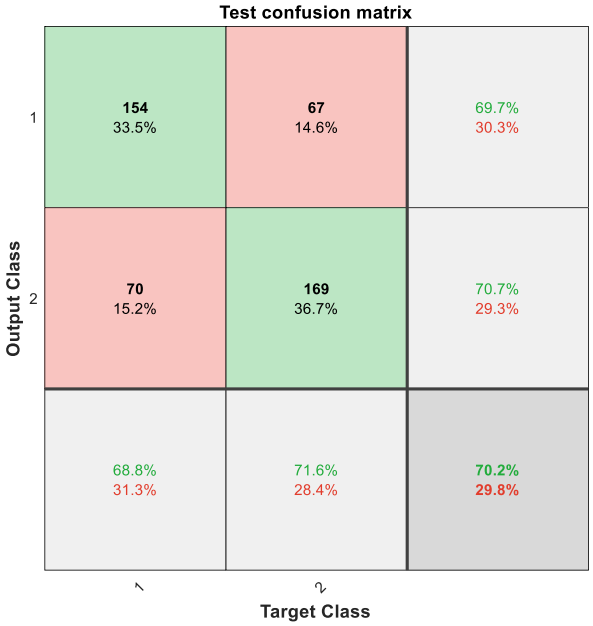
Trénovanie a výsledky :



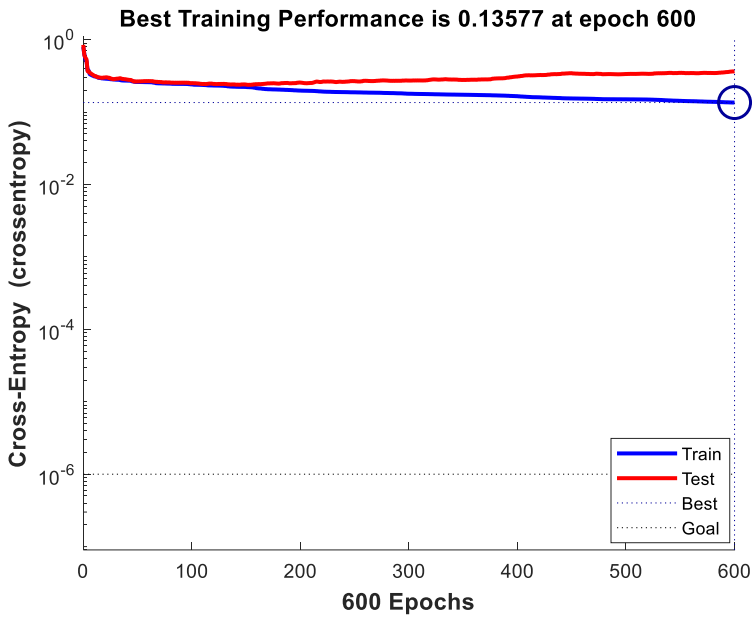
Figures – Uloha3 – OneRun – OneRun40N – AllCM40



Figures – Uloha3 – OneRun – OneRun40N – TrainCM40



Figures – Uloha3 – OneRun – OneRun40N – TestCM40



Figures – Uloha3 – OneRun – OneRun40N – Performance40

Na **Performance grafe** je vidno, že minimálnu chybu sme nedosiahli po 600 trénovacích cykloch . Toto znamená, že aj na trénovacích a testovacích dátach budeme mať príliš vysokú chybovosť (na testovacích väčšiu) , a preto ako je vidno vyššie na kontingenčnej matici (**All confusion matrix**) sme dosiahli len 80.4%, čo je len tesne nad požadovanou 80%.

Po 10 spustení sme dosiahli nasledujúce výsledky:

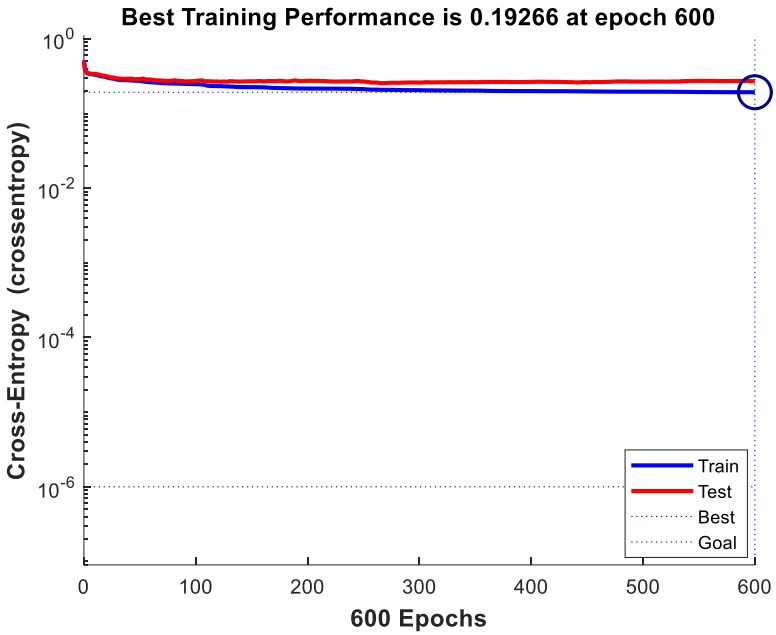
```
==============DOSIAHNUTÉ HODNOTY==============
Na trénovacích dátach:
   88.5673   87.9884   90.3039   88.5673   88.1331   90.8828   89.4356   89.0014   89.2909   90.1592

Na testovacích dátach:
   72.6087   69.1304   70.4348   72.3913   71.9565   69.1304   71.0870   74.1304   70.4348   71.0870

======================TRAIN=====================
Minimálna úspešnosť na trénovacích dátach :
   87.9884

Maximálna úspešnosť na trénovacích dátach :
   90.8828

Priemerná úspešnosť na trénovacích dátach :
   89.2330
```

```
====================TEST====================
Minimálna úspešnosť na testovacích dátach :
   69.1304

Maximálna úspešnosť na testovacích dátach :
   74.1304

Priemerná úspešnosť na testovacích dátach :
   71.2391
```

Figures – Uloha3 – TenRun40N – Dosiahnute_hodnoty, Train, Test

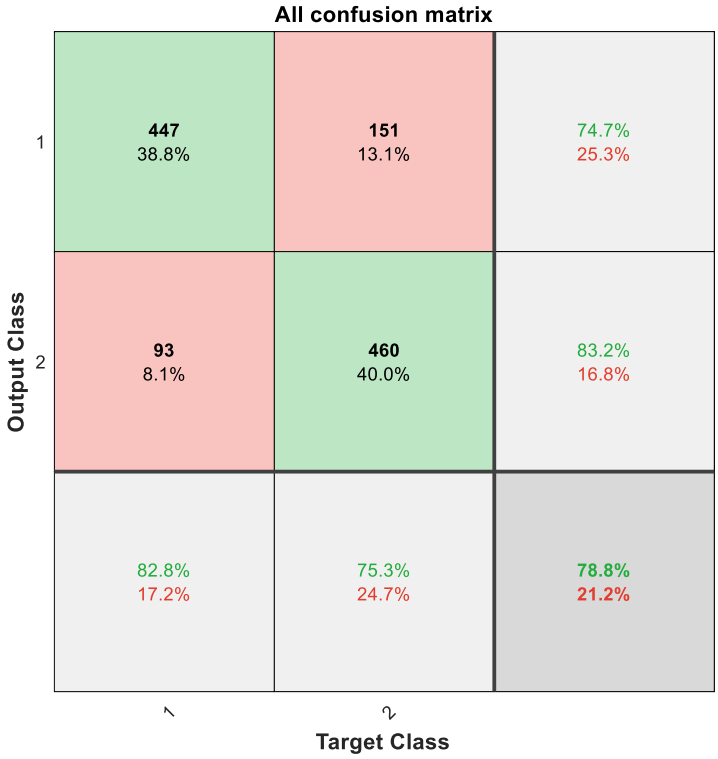
V priečinku ***Figures/Uloha3/TenRun40N*** sme nechali grafy proces trénovania a kontingenčné matice pre overenie.

Vyskúšali sme aké hodnoty dosiahneme pri používaní **5** neurónov:

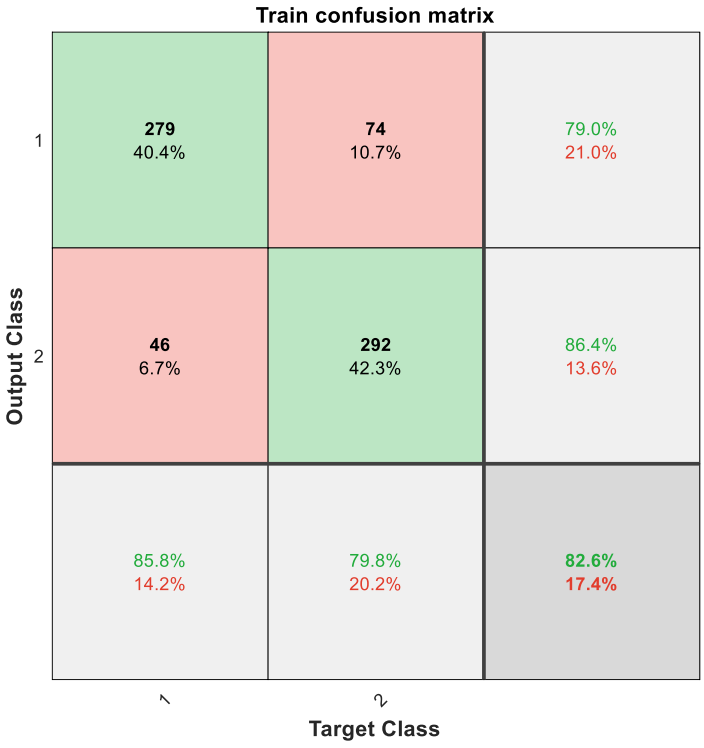


Figures – Uloha3 – OneRun – OneRun5N – Performance5

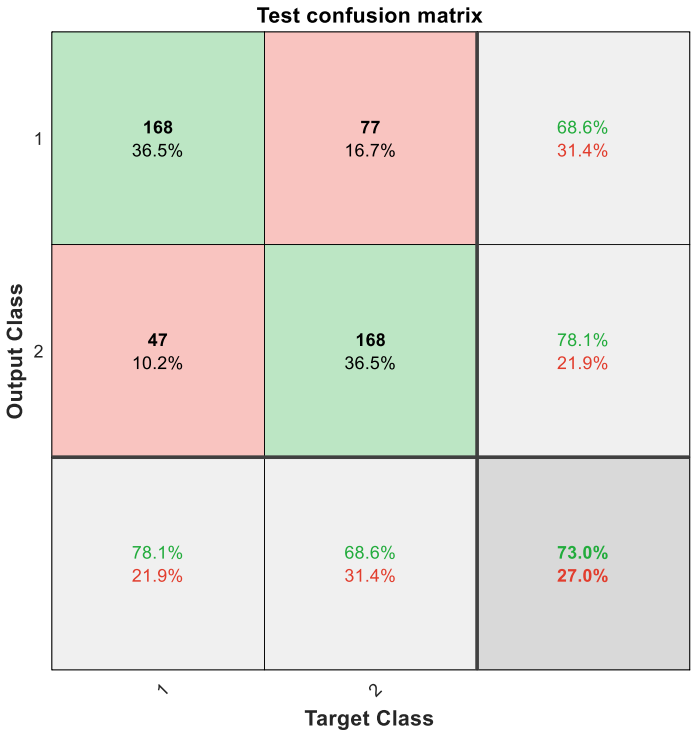
Minimálnu chybu sme nedosiahli po 600 trénovacích cykloch . Je vidno, že veľkú úspešnosť nedosiahneme, lebo máme veľkú chybovosť na oboch dátach. Na testovacích bude to väčšie.



Figures – Uloha3 – OneRun – OneRun5N – AllCM5



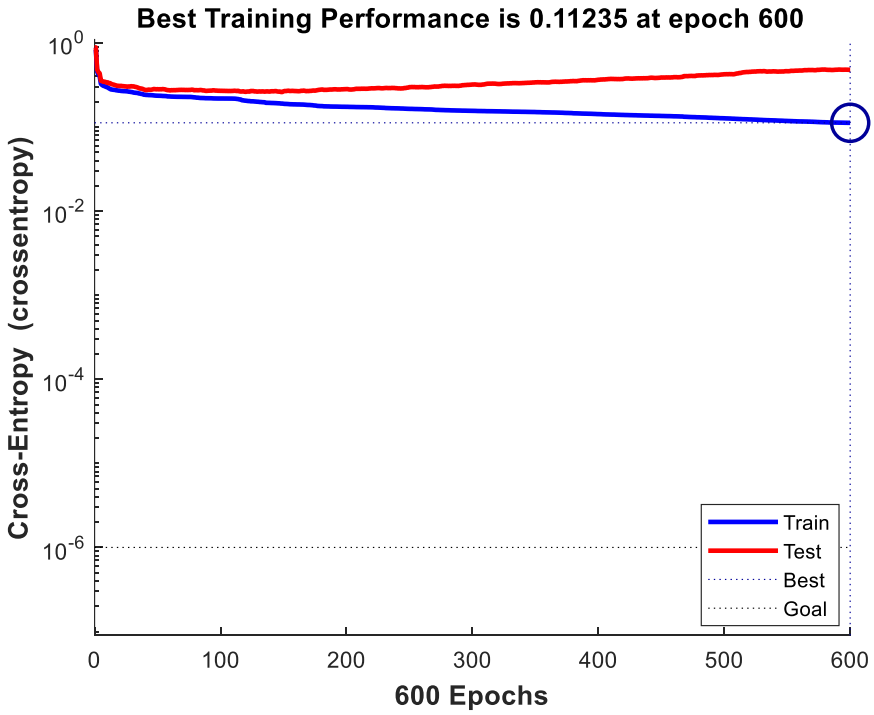
Figures – Uloha3 – OneRun – OneRun5N – TrainCM5



Figures – Uloha3 – OneRun – OneRun5N – TestCM5

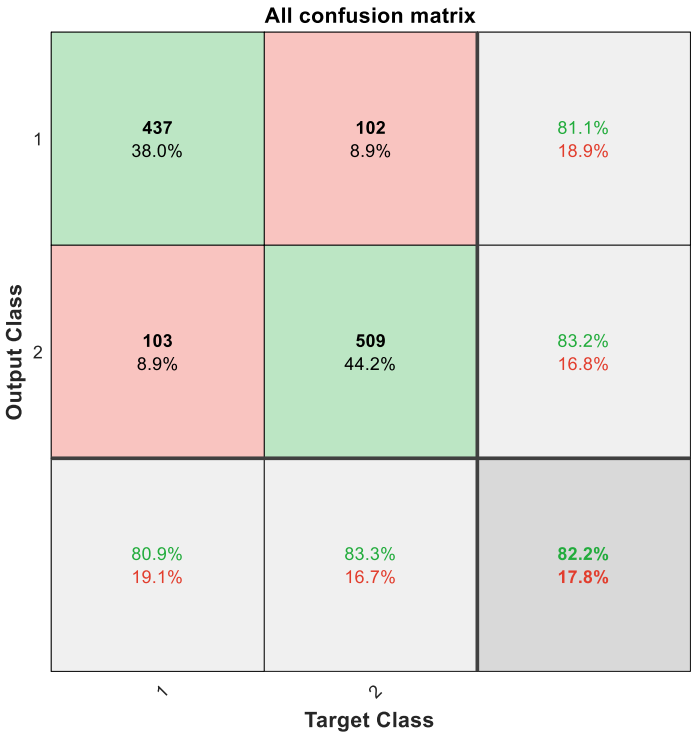
Požadovanú 80% úspešnosť, sme nedosiahli, lebo sme mali príliš málo neurónov.

Potom sme vyskúšali čo sa stane ak používame príliš veľa neurónov. Nasledujúce hodnoty sme dosiahli pri používaní **150** neurónov:

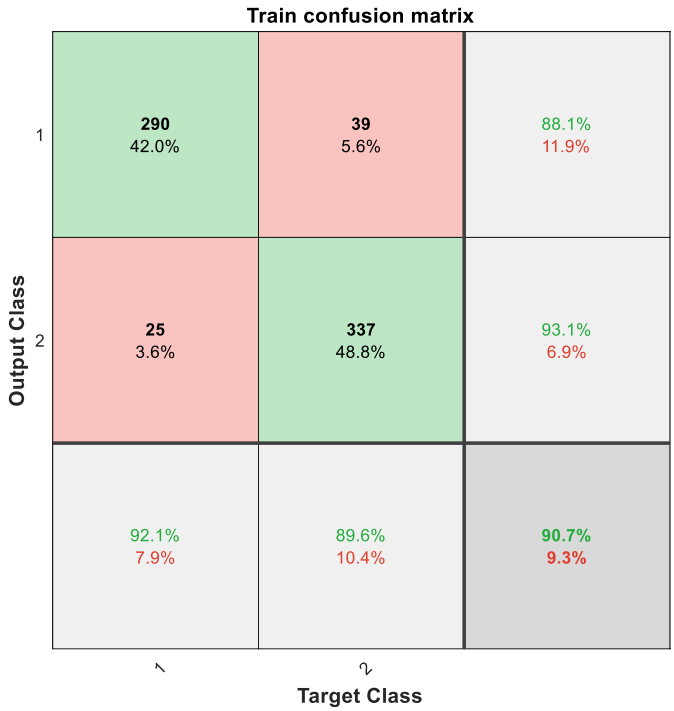


Figures – Uloha3 – OneRun – OneRun150N – Performance150

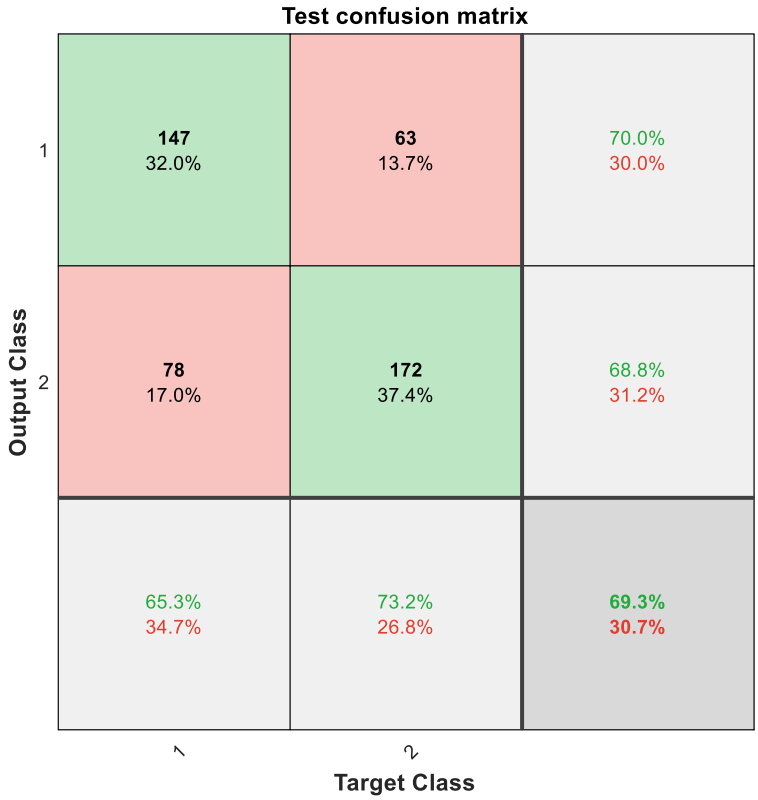
Ani v tomto prípade sme nedosiahli minimálnu chybu po 600 trénovacích cykloch.



Figures – Uloha3 – OneRun – OneRun150N – AllCM150



Figures – Uloha3 – OneRun – OneRun150N – TrainCM150



Figures – Uloha3 – OneRun – OneRun150N – TestCM150

Napriek tomu, že sme nedosiahli minimálnu chybu splnili sme 80% celkovú úspešnosť.

Testovanie:

Na koniec sme náhodne vybrali 10 pacientov. Chceli by sme ich klasifikovať, či majú diabetickú retinopatiu alebo nie. Vyzeralo to nasledovne:

```
==================TESTOVANIE==================
================= 1. pacient =================

prediction =

    0.0000
    1.0000


predicted_output =

     1


real_output =

     1
```

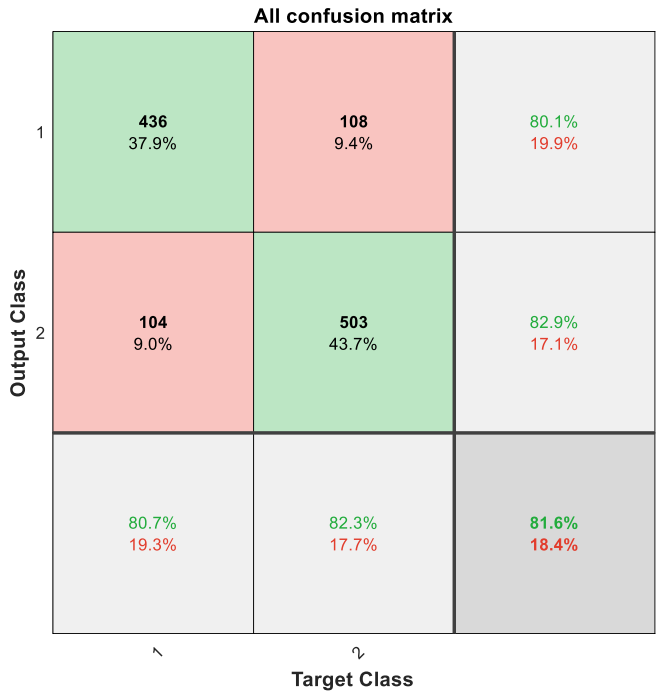
```
================= 2. pacient =================

prediction =

    0.0000
    1.0000


predicted_output =

     1


real_output =

     1

 ================= 3. pacient =================

 prediction =

     0.0844
     0.9156


 predicted_output =

      1


 real_output =

      0
================= 4. pacient =================

prediction =

    0.5415
    0.4585


predicted_output =

     0


real_output =

     1
```

```
================= 5. pacient =================

prediction =

    0.7050
    0.2950


predicted_output =

     0


real_output =

     1

================= 6. pacient =================

prediction =

    0.9925
    0.0075


predicted_output =

     0


real_output =

     0

================= 7. pacient =================

prediction =

    0.0017
    0.9983


predicted_output =

     1


real_output =

     1
```

```
================= 8. pacient =================

prediction =

    0.9788
    0.0212


predicted_output =

     0


real_output =

     0

================= 9. pacient =================

prediction =

    0.7535
    0.2465


predicted_output =

     0


real_output =

     0

================= 10. pacient =================

prediction =

    0.8677
    0.1323


predicted_output =

     0


real_output =

     0
```

Figures – Uloha3 – Testovanie – Testovanie1 ... Testovanie10

Sieť na ktorom sme testovali pacientov mala celkovú úspešnosť:



Z 10 pacientov nám správne klasifikoval 7. Podobne ako v 1. úlohe sme dostali na výstup toľko riadkov koľko tried sme mali , teda pacient je zdravý (prvý riadok) alebo má diabetickú retinopatiu (druhý riadok). Hodnoty sú z intervalu (0,1), ktoré budú reprezentovať percentá nakoľko bolo sieť schopná zaradiť do jednej skupiny. Pomocou funkcie **vec2ind** sme zobrazili triedy do ktorej sme zaradili pacienta. Potom sme ešte zobrazili aj správne riešenie.

Bonus

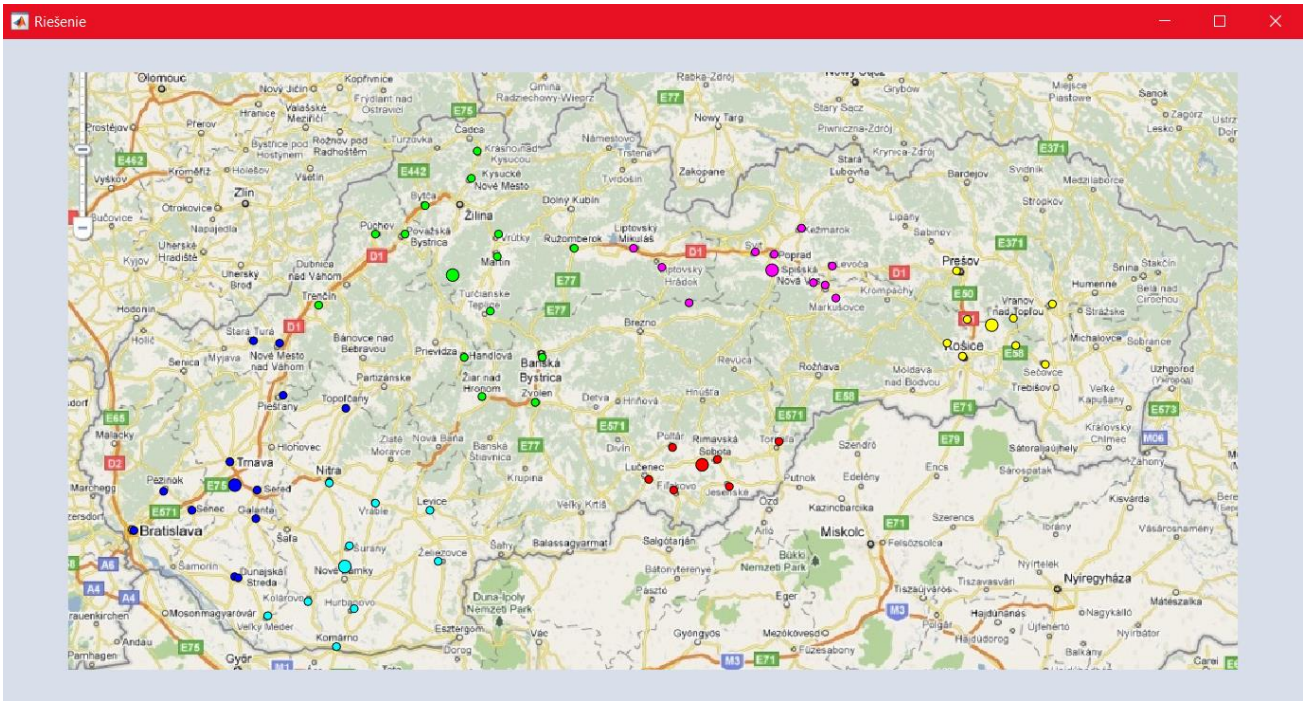
Na základe rozmiestnenia odberateľov na mape Slovenska sme mali nájsť optimálne rozmiestnenie distribučných skladov pre počet 6,8,9,10 pomocou Kohonenovej siete. Na koniec sme mali farebne znázorniť rozložené sklady a k nim priradených odberateľov. Mali sme aj vyčísliť počet priradených odberateľov, sumu vzdialeností odberateľom od priradeného skladu a priemernú vzdialenosť odberateľa.

Túto úlohu sem riešili implementovaním Kohonenovej siete, v algoritmu učenia sme mali nastavené tieto počiatočné hodnoty:

```
counter = 10;

gama1 = 0.8;        % zaciatocny krok ucenia

r1 = 4;             % polomer susedtsva

h1 = 1;             % vyska funkcie susedstva
```

Po učení sme dosiahli nasledujúce výsledky:

1) 6 skladov

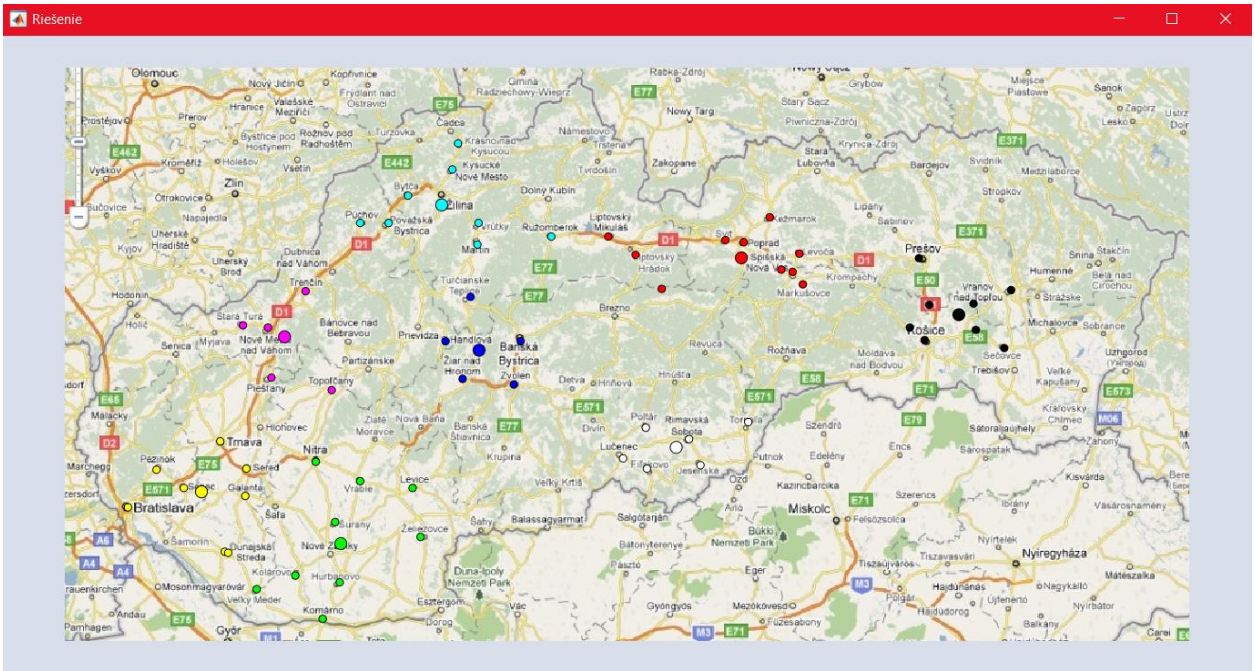



```
Sklad 1
     8

Suma vzdial. 1
  312.4221

Priemerná vzdial. 1
   39.0528

Sklad 2
    10

Suma vzdial. 2
  564.3883

Priemerná vzdial. 2
   56.4388

Sklad 3
     9

Suma vzdial. 3
  544.9909

Priemerná vzdial. 3
   60.5545
```

```
Sklad 4
     6

Suma vzdial. 4
  221.4991

Priemerná vzdial. 4
   36.9165

Sklad 5
    14

Suma vzdial. 5
   1.1577e+03

Priemerná vzdial. 5
   82.6899

Sklad 6
    12

Suma vzdial. 6
  882.9651

Priemerná vzdial. 6
   73.5804
```

Figures – Bonus – 6 – Rozlozenie6,Sklad6_1,Sklad6_2

2) 8 skladov



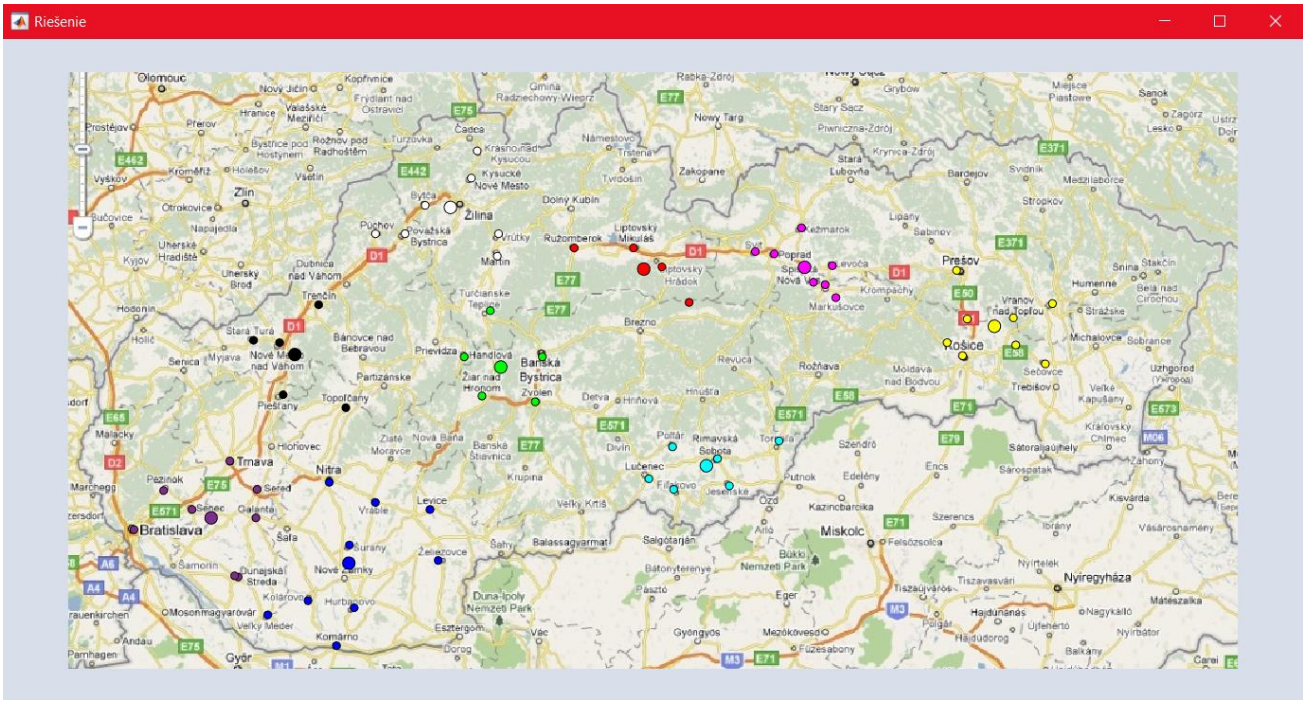

```
Sklad 1
      8

Suma vzdial. 1
  375.6872

Priemerná vzdial. 1
    46.9609

Sklad 2
      5

Suma vzdial. 2
  201.3544

Priemerná vzdial. 2
    40.2709

Sklad 3
      8

Suma vzdial. 3
  430.2069

Priemerná vzdial. 3
    53.7759
```

```
Sklad 4
     10

Suma vzdial. 4
  564.8048

Priemerná vzdial. 4
    56.4805

Sklad 5
      9

Suma vzdial. 5
  543.7715

Priemerná vzdial. 5
    60.4191

Sklad 6
      5

Suma vzdial. 6
  189.4491

Priemerná vzdial. 6
    37.8898
```

```
Sklad 7
      6

Suma vzdial. 7
  220.9199

Priemerná vzdial. 7
    36.8200

Sklad 8
      8

Suma vzdial. 8
  310.4075

Priemerná vzdial. 8
    38.8009
```

Figures – Bonus – 8 – Rozlozenie8,Sklad8_1,Sklad8_2,Sklad8_3

3) 9 skladov



Sklad 1
8

Suma vzdial. 1
311.1090

Priemerná vzdial. 1
38.8886

Sklad 2
7

Suma vzdial. 2
205.2549

Priemerná vzdial. 2
29.3221

Sklad 3
6

Suma vzdial. 3
221.1260

Priemerná vzdial. 3
36.8543

Sklad 4
4

Suma vzdial. 4
145.8737

Priemerná vzdial. 4
36.4684

Sklad 5
5

Suma vzdial. 5
189.3519

Priemerná vzdial. 5
37.8704

Sklad 6
9

Suma vzdial. 6
542.1669

Priemerná vzdial. 6
60.2408

Sklad 7
7

Suma vzdial. 7
323.5107

Priemerná vzdial. 7
46.2158

Sklad 8
5

Suma vzdial. 8
199.8509

Priemerná vzdial. 8
39.9702

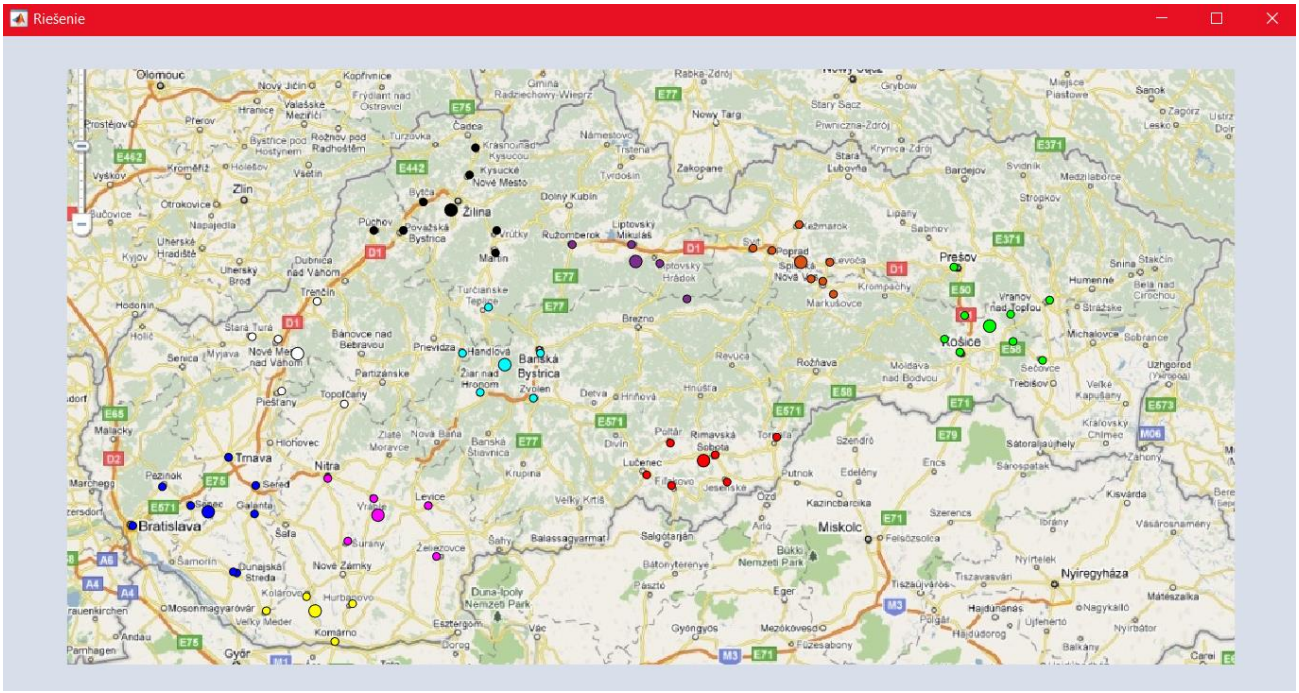
Sklad 9
8

Suma vzdial. 9
376.1343

Priemerná vzdial. 9
47.0168

Figures – Bonus – 9 – Rozlozenie9,Sklad9_1,Sklad9_2,Sklad9_3

4) 10 skladov



Sklad 1
4

Suma vzdial. 1
119.9600

Priemerná vzdial. 1
29.9900

Sklad 2
5

Suma vzdial. 2
207.0446

Priemerná vzdial. 2
41.4089

Sklad 3
5

Suma vzdial. 3
190.1329

Priemerná vzdial. 3
38.0266

Sklad 4
6

Suma vzdial. 4
221.3169

Priemerná vzdial. 4
36.8861

Sklad 5
8

Suma vzdial. 5
311.1228

Priemerná vzdial. 5
38.8903

Sklad 6
8

Suma vzdial. 6
375.8786

Priemerná vzdial. 6
46.9848

Sklad 7
5

Suma vzdial. 7
203.8315

Priemerná vzdial. 7
40.7663

Sklad 8
7

Suma vzdial. 8
324.4070

Priemerná vzdial. 8
46.3439

Sklad 9
4

Suma vzdial. 9
146.1319

Priemerná vzdial. 9
36.5330

Sklad 10
7

Suma vzdial. 10
207.3018

Priemerná vzdial. 10
29.6145

Figures – Bonus – 10 – Rozlozenie10,Sklad10_1,Sklad10_2,Sklad10_3,Sklad10_4